Jump to: Account info Compute Full Site Delivery Getting started Integrations Security

Account info guides

Category: Account management

These articles describe how to manage account access.

Account lockouts

🔗 /en/guides/account-lockouts

Why is my account locked?

For security reasons, Fastly limits the number of times someone can try logging in to an account. We don't want to give people unlimited attempts at guessing your password, so we stop them from trying after a limited number of failed attempts to sign in. You can change your password at any time when you're logged in to your account.

I am not using two-factor authentication. How can I access my account?

Once locked, you will not be able to sign in to your account, even with the correct password. To unlock your account because you exceeded the number of guesses you were allowed, you can either:

- reset the password yourself.
- ask a superuser associated with your account to send you a password reset email.

I am using two-factor authentication. How can I access my account?

I don't have my mobile device.

If you do not have access to your mobile device, you can complete the login process using one of your recovery codes. These were the recovery codes you saved in a secured location outside of your Fastly account when two-factor authentication was first enabled. You can continue to use your recovery codes until your device is once again accessible. Recovery codes can only be used once, however, so remember to regenerate a new set to avoid running out before you recover your mobile device.

If you don't believe you will be able to recover your lost mobile device and you still have at least two recovery codes left, you can log in with one recovery code and disable two-factor authentication with a second code. Once two-factor authentication is disabled, you can re-enable it with a new mobile device at a later time and regenerate a new set of codes.

I don't have my mobile device and I don't have my recovery codes.

If you don't have your mobile device and didn't save any recovery codes, have another user at your company with the superuser role follow the steps to reset a user's two-factor authentication. If you don't know who a superuser is at your company, contact support.

I don't have my phone, I didn't save my recovery codes, and I am the only superuser for the account.

Contact Customer Support. We will verify that you are associated with the company by phone. We will use the contact information located on the company website or under the Fastly account tab. Upon verification, we will send you a recovery code and reset your password.

Was my account compromised?

If a user's account appears to be hacked or phished, we may proactively reset the passwords for the affected accounts to revoke access to the hacker. In these cases, we send an email to the account's real owner (you) with additional information on how to reset the password. If you received one of these emails, follow the instructions in the email.

If you think your account has been hacked or phished, contact Customer Support immediately.

How is a locked account different from a blocked account?

Fastly allows you to restrict who can access your Fastly account based on the IP address of the person attempting to log in. This means that even with the correct login name and password, access to your Fastly account may be blocked if the IP doesn't match your company's list of allowed addresses.

If your company enables this optional IP allowlisting, they must keep the list of restricted IP addresses up to date. Only users with the role of superuser can make changes to the IP allowlist settings (your account owner is always a superuser), and your account owner must have a valid telephone number on file to do so.

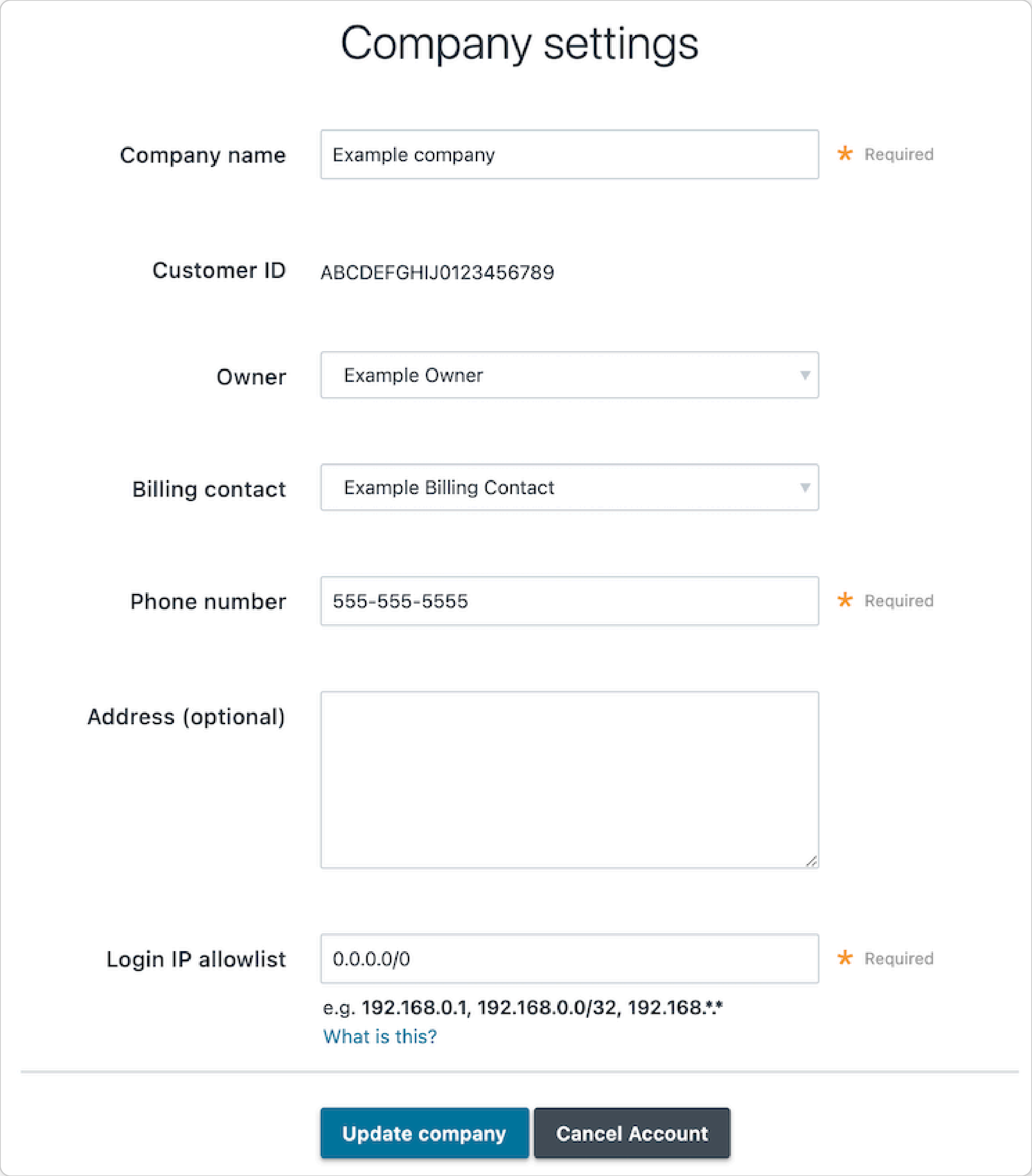
If your IP addresses change after allowlisting is enabled and you forget to update your allowlist configuration, you will be locked out of your account. You will need to contact support to request that a Customer Support representative contact your account's owner via telephone during Fastly's regular business hours. To protect your account's security, we will not unlock your account based on an email request alone.

* * *

Changing company profile details

/en/guides/changing-company-profile-details

Fastly allows you change most of the details about your company after your account has been created, including the company name and address, the company's listed account owner, and any of the contacts you've created for specific company communications.



Company settings

Company name	Example company	* Required
Customer ID	ABCDEFGHIJ0123456789	
Owner	Example Owner	
Billing contact	Example Billing Contact	
Phone number	555-555-5555	* Required
Address (optional)		
Login IP allowlist	0.0.0.0/0	* Required

e.g. 192.168.0.1, 192.168.0.0/32, 192.168.*.*

NOTE

Customer IDs are automatically generated upon account creation and cannot be edited.

Who can view and change company profile details

Users assigned the role of engineer can view company profile details. You must be the account owner or be assigned the role of superuser to change company profile details.

Changing an account's company settings

Fastly allows you to change an account's company settings like company name, owner, and company address at any time after your account has been created.

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Company name** field of the **Company settings** area, replace the current company name with the new one. This is the company name that will be displayed on your monthly invoice.
3. From the **Owner** menu, select a new company owner from the available names.
4. In the **Address (optional)** field, replace the existing company address with the new one.
5. To define the specific range of IP addresses authorized to access your account, follow the instructions for enabling or disabling an IP allowlist.
6. Click **Update Company**.

Creating company contacts

We allow you to set up a variety of company contacts to help Fastly route conversations and requests to the correct humans in your organization. To create a new company contact:

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Company contacts** area, click **Add contact**.
3. From the **Select contact type** menu, select the type of company contact to create:
    - **Primary:** Select this option to specify who Fastly should contact when account renewals and decisions (e.g., new product adoption) must be discussed with someone in your organization.
    - **Technical:** Select this option to specify who Fastly should contact when product changes (e.g., retirements and deprecations) must be communicated to someone in your organization who can make changes to your services.
    - **Security:** Select this option to specify who Fastly should contact when security-related vulnerabilities or incidents must be communicated to someone in your organization.
    - **Emergency:** Select this option to specify who Fastly should contact when urgent, high-severity issues must be communicated to someone in your organization.
    - **Billing:** Select this option to specify who Fastly should contact when sending out invoices to someone in your organization or for confirmation of cost-based training that will be billed at the account level.
    - **Third-Party Reports:** Select this option to specify who Fastly should contact when compliance-related issues (e.g., DMCA, GDPR) need to be addressed by someone in your organization.
4. From the **Name** menu, select the name of the human contact to specify for this contact type.

    > TIP
    >
    > Some contacts for your organization may not have nor require Fastly accounts and, thus, can't be selected from the controls. You can enter contact information manually by instead clicking **Enter contact manually** to enable manual entry of information in all Company contacts fields.

5. *(Optional)* In the **Phone number** field, enter a telephone number at which to contact the selected user.
6. Click **Add**. The contact's information appears.

> IMPORTANT

Company contact information can only be added or removed, not edited. Once you've added at least one company contact of a particular type, you cannot remove that contact until you've added another of the same type to replace it. To update a contact's information, remove their entry via the **Remove** link and re-add them to the company contacts list instead. Contacts are automatically removed from the system if the assigned contact is deleted as an account user.

---

* * *

Enabling an IP allowlist for account logins through the web interface

/en/guides/enabling-an-ip-allowlist-for-account-logins-through-the-web-interface

---

Fastly allows you to define the range of IP addresses authorized on your Fastly account from which users are able to login to the Fastly web interface. This optional IP allowlisting functionality is not enabled by default. It can restrict access to most of Fastly's API endpoints.

Limitations and considerations

If you decide to use optional IP allowlisting, keep the following things in mind:

- **Metrics and stats API endpoints are exempt.** Metrics and stats API endpoints, such as historical stats or real-time analytics, do no support allowlisting.
- **An account owner telephone number is required.** During setup, Fastly checks your current IP address against the list you provide to ensure you don't lock yourself out of your account. If your IP addresses change at a later date (for example, because you move offices) and you forget to update your allowlist configuration, you will be locked out of your account.

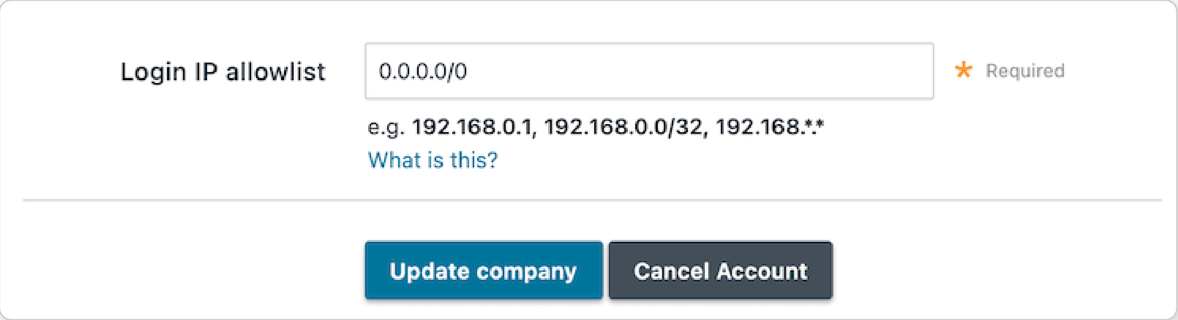
WARNING

If you are locked out of your account, you will need to contact support and request that a Customer Support representative contact your account's owner via telephone during Fastly's regular business hours. To protect your account's security, ***we will not unlock your account based on an email request alone***.

Enabling an IP allowlist for account logins through the web interface

To restrict logins to the Fastly web interface based on a specific list or range of IP addresses, follow these steps.

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Login IP allowlist** field of the **Company settings** area, replace `0.0.0.0/0` (the default IP range indicating no allowlisting) with the IP addresses for which web interface access to your account should be restricted.



You can include single or multiple IP addresses or IP ranges (separated by commas) as follows:

- a single IPv4 address (e.g., replace the default with `192.168.0.1`)
- an IPv4 CIDR range (e.g., replace the default with `192.168.0.0/24`, `192.168.0.1/32`)
- an IPv4 Wildcard range (e.g., replace the default with `192.168.0.*`, `192.168.*.1`, `192.168.*.*`)

3. Click **Update Company**.

Disabling an IP allowlist for account logins through the web interface

To disable IP allowlisting on your Fastly account and allow web interface access to any IP range, follow these steps.

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Login IP allowlist** field of the **Company settings** area, enter `0.0.0.0/0` (the default IP range indicating no allowlisting).
3. Click **Update Company**.

---

* * *

Enabling and disabling two-factor authentication

/en/guides/enabling-and-disabling-two-factor-authentication

IMPORTANT

This guide only applies to customers with Fastly accounts or with Signal Sciences accounts linked to Fastly accounts. If you have a Signal Sciences account that isn't linked to a Fastly account, check out our guide to enabling and disabling two-factor authentication for Signal Sciences.

Fastly supports two-factor authentication, a two-step verification system, for logging in to the web interface. In a two-factor authentication security process, users provide two means of identifying themselves to the system, typically by providing the system with something they know (for example, their login ID and password combination) and something they have (such as an authentication code). Organizations can enable company-wide two-factor authentication to require all users within the organization to use two-factor authentication.

Before you begin

You'll need to enter an authentication code regularly. Once two-factor authentication has been enabled, an authentication code will be requested upon login at least every 14 days for each computer and browser you use to access the Fastly web interface.

A mobile device is required. Using this security feature with a Fastly account requires a mobile device capable of scanning a barcode or QR code using a downloadable authenticator application. We recommend the following:

- For Android, iOS, and Blackberry: Google Authenticator
- For Android and iOS: Duo Mobile
- For Windows Phone: Authenticator

There are special requirements for using this feature with API tokens. Check out the API token documentation for more information.

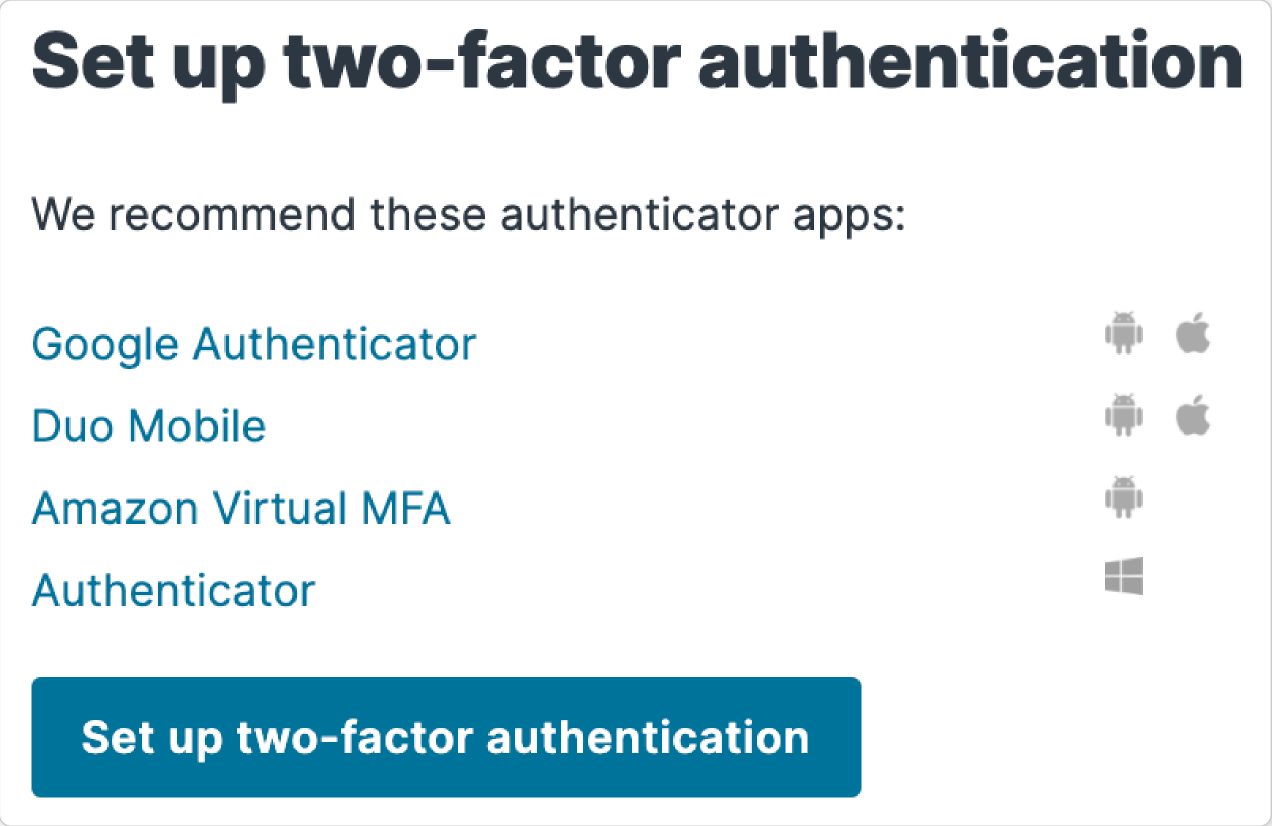
Managing two-factor authentication as a user

Depending on whether or not your organization has enabled company-wide two-factor authentication, you may be able to enable and disable two-factor authentication for your personal account. We also have instructions for recovering access to your account if you lose your mobile device.

Enabling two-factor authentication

Follow these steps to enable two-factor authentication for your account.

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **Two-factor authentication**.



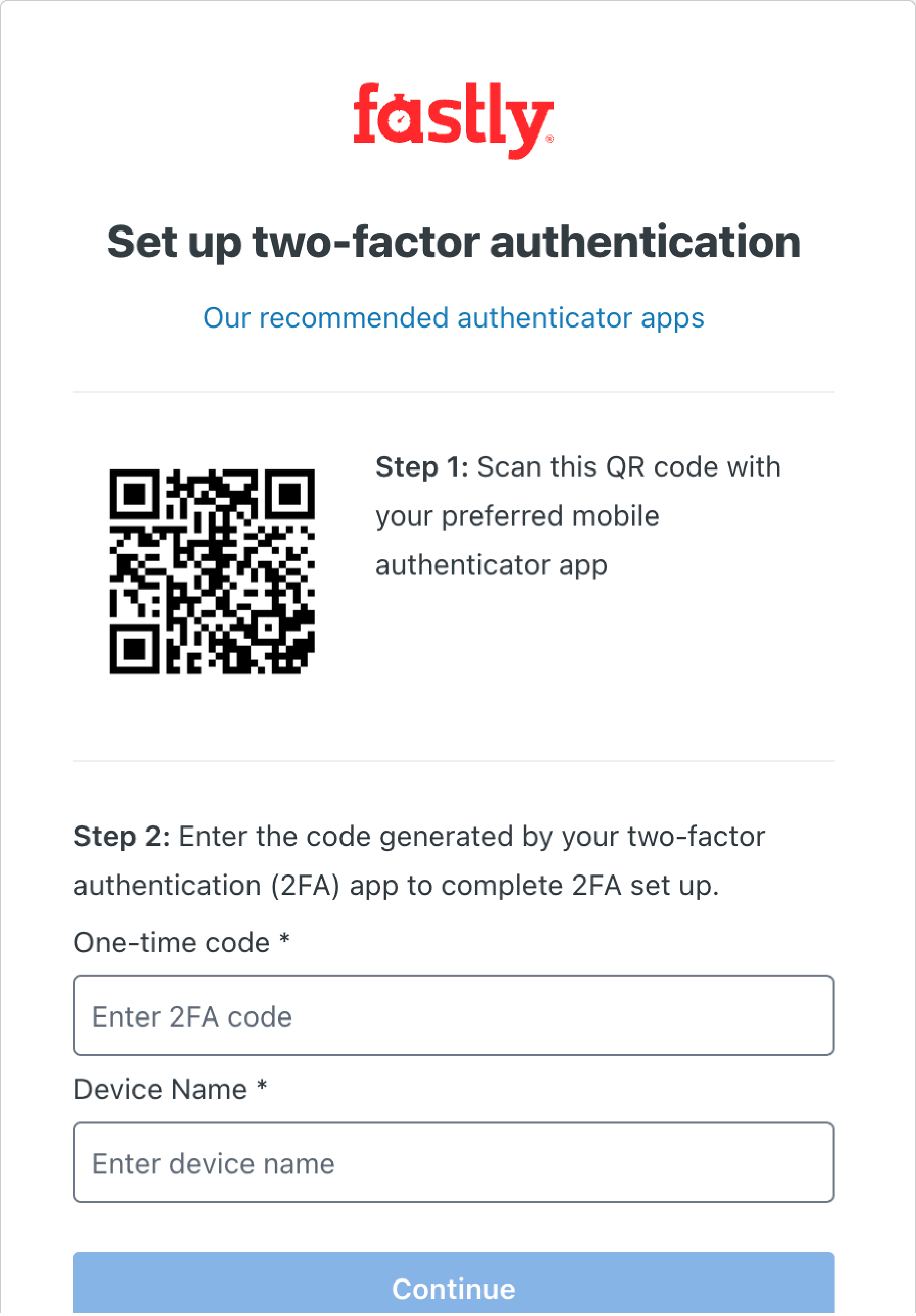
Set up two-factor authentication

We recommend these authenticator apps:

Google Authenticator
Duo Mobile
Amazon Virtual MFA
Authenticator

3. Click **Set up two-factor authentication**.

4. Re-enter your login credentials and then click **Continue**. The setup page with the authentication QR code appears.



Set up two-factor authentication

Our recommended authenticator apps

Step 1: Scan this QR code with your preferred mobile authenticator app

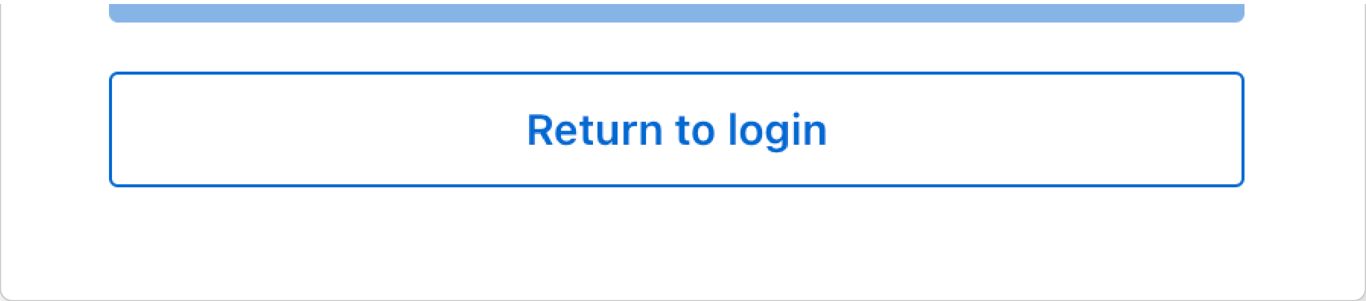
Step 2: Enter the code generated by your two-factor authentication (2FA) app to complete 2FA set up.

One-time code *

Enter 2FA code

Device Name *

Enter device name



IMPORTANT

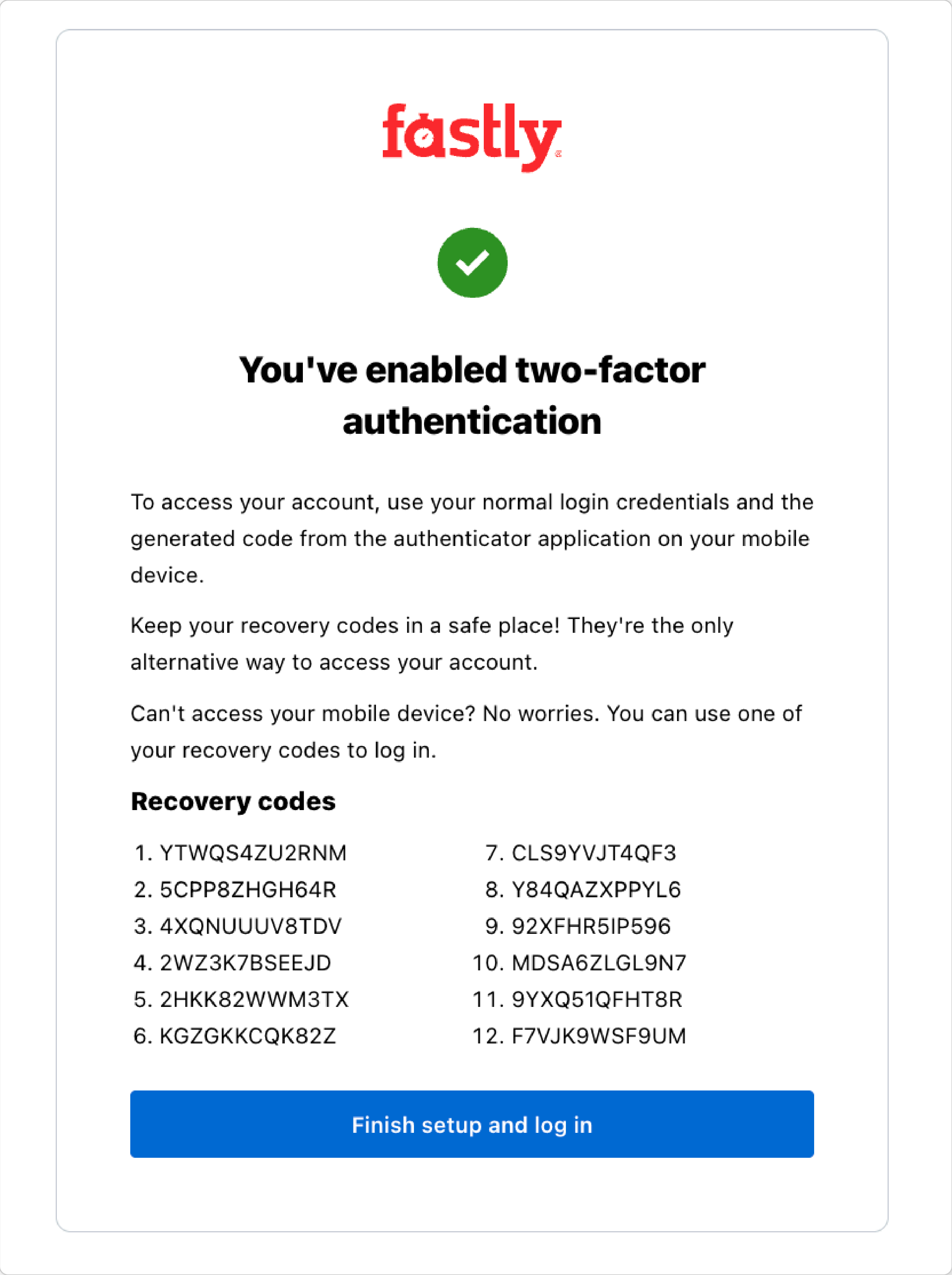
The QR code above is an example. Scan the one that appears in the Fastly application, not in this guide.

5. Launch the authenticator application installed on your mobile device and scan the displayed QR code or manually enter the key displayed in the setup window. A time-based authentication code appears on your mobile device. Depending on your device, however, a browser link may first appear. You need to click this link to save it. When you do, the words `Secret saved` appear briefly.
6. In the **One-time code** field in the Fastly application, enter the time-based authentication code displayed by your mobile device.

IMPORTANT

A common time syncing issue may cause your authenticator codes to fail. You can correct this using Google's instructions for your authenticator application.

7. In the **Device Name** field in the Fastly application, enter a name to help you identify your device.
8. Click **Continue**. The confirmation screen appears along with your recovery codes.



You've enabled two-factor authentication

To access your account, use your normal login credentials and the generated code from the authenticator application on your mobile device.

Keep your recovery codes in a safe place! They're the only alternative way to access your account.

Can't access your mobile device? No worries. You can use one of your recovery codes to log in.

Recovery codes

1. YTWQS4ZU2RNM
2. 5CPP8ZHGH64R
3. 4XQNUUUV8TDV
4. 2WZ3K7BSEEJD
5. 2HKK82WWM3TX
6. KGZGKKCQK82Z
7. CLS9YVJT4QF3
8. Y84QAZXPPYL6
9. 92XFHR5IP596
10. MDSA6ZLGL9N7
11. 9YXQ51QFHT8R
12. F7VJK9WSF9UM

IMPORTANT

Recovery codes are only displayed once. Be sure to store a copy of them in a safe place in the same order they appear on the confirmation screen. If you're ever unable to access your mobile device, the recovery codes can be used to log in when your account has two-factor authentication enabled. Each of these recovery codes can only be used once, but you can regenerate a new set of 12 at any time and any previously generated codes that are still unused will be invalidated).

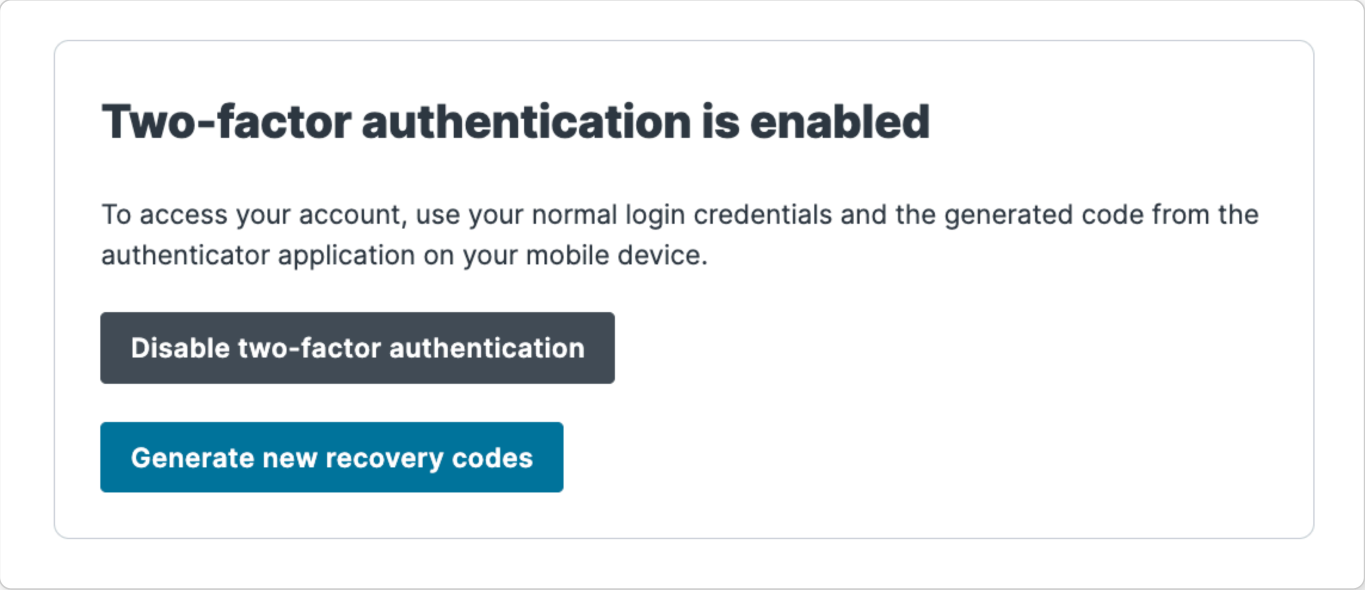
Once you enable two-factor authentication for your account, any other open sessions will require reauthentication. For example, if you enable two-factor authentication in one browser window and you're viewing various aspects of your service through multiple additional browser windows, you will be required to reauthenticate in those additional windows, this time using an authentication code generated by the authenticator application installed on your mobile device (in addition to your email and password). Future logins will also require an authenticator code. By default, the system requires you to authenticate your login using an authentication code at least every two weeks for each computer and browser you use to access the Fastly web interface.

Disabling two-factor authentication

Once two-factor authentication is enabled for your account, you can disable it at any time by following the steps below.

> IMPORTANT
>
> If your organization has enabled company-wide two-factor authentication, you cannot disable two-factor authentication for your account.

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **Two-factor authentication**.



Two-factor authentication is enabled

To access your account, use your normal login credentials and the generated code from the authenticator application on your mobile device.

Disable two-factor authentication
Generate new recovery codes

3. Click **Disable two-factor authentication**.
4. In the **Authentication Code** field, enter the time-based authentication code displayed in the authenticator application on your mobile device, then click **Confirm and Disable**.

> NOTE
>
> If you have lost your mobile device, you can enter a recovery code in the **Authentication Code** field. For more information, check out the section on what to do if you lose your mobile device.

What to do if you lose your mobile device

If you lose your mobile device after enabling two-factor authentication, use a recovery code to log in to your Fastly account.

> ⚠ WARNING
>
> You must use the recovery codes for your account in the order they were supplied to you. If you don't, you'll be asked to enter a valid code.

You can continue to use recovery codes to log in until you get your mobile device back. Recovery codes can only be used once, however, so remember to regenerate a new list of codes to avoid running out before you recover your mobile device.

If you do not believe you will be able to recover your lost mobile device and you still have at least two recovery codes left, you can log in with one recovery code and disable two-factor authentication with a second code. Once two-factor authentication is disabled, you can re-enable it with a new mobile device at a later time and regenerate a new set of codes.

If your organization has enabled company-wide two-factor authentication, you can contact a superuser for your organization and ask them to reset your two-factor authentication.

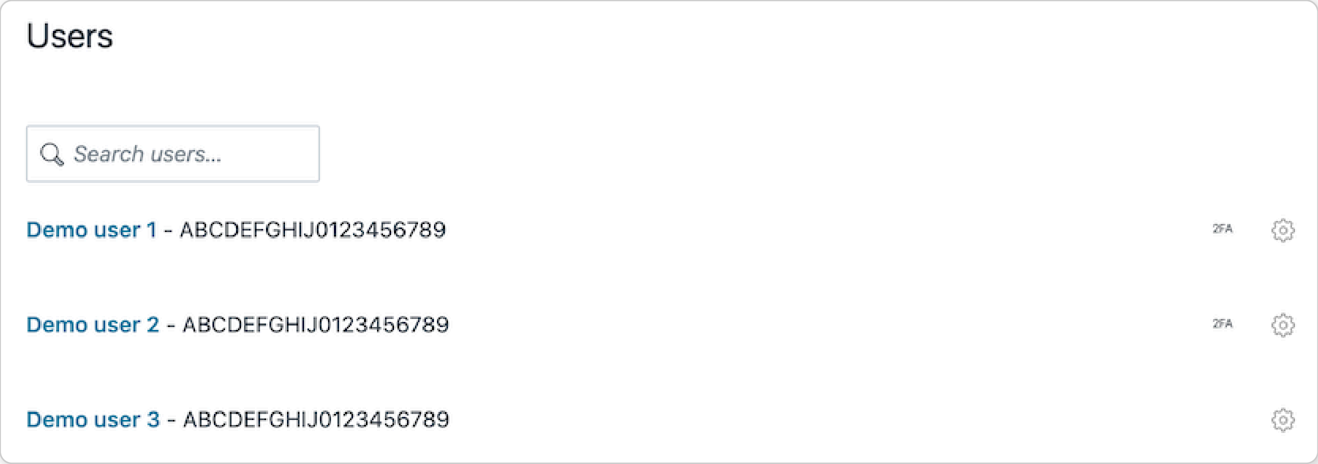
Locked out of your account? See our article on what you can do about it.

Managing two-factor authentication as a superuser

Organizations can enable two-factor authentication users one at a time or all at once for all of their users. When the company-wide two-factor authentication feature is enabled, all users within the organization are required to use two-factor authentication to log in to the Fastly web interface, and they cannot disable two-factor authentication for their accounts.

Enabling two-factor authentication for a single user

If you are assigned the superuser role for your organization, you can view who has two-factor authentication enabled the User management settings for your Account. Users with this feature enabled have 2FA displayed next to their names.


Users

Search users...

Demo user 1 - ABCDEFGHIJ0123456789 2FA

Demo user 2 - ABCDEFGHIJ0123456789 2FA

Demo user 3 - ABCDEFGHIJ0123456789

To disable two-factor authentication for any user within your organization, select **Disable 2FA** from the menu that appears when you click the gear icon next to that user's name.

Resetting a user's two-factor authentication

If company-wide two-factor authentication is enabled, and a user within the organization gets locked out of their account or needs to enable a new device, an account superuser can reset their two-factor authentication. To reset a user's two-factor authentication, follow the steps below.

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Users** area, click the gear next to a user and then select **Reset 2FA**. A warning message appears.
4. Click **Reset**. The user will need to set up two-factor authentication for their account the next time they log in.

Disabling two-factor authentication for a single user's account

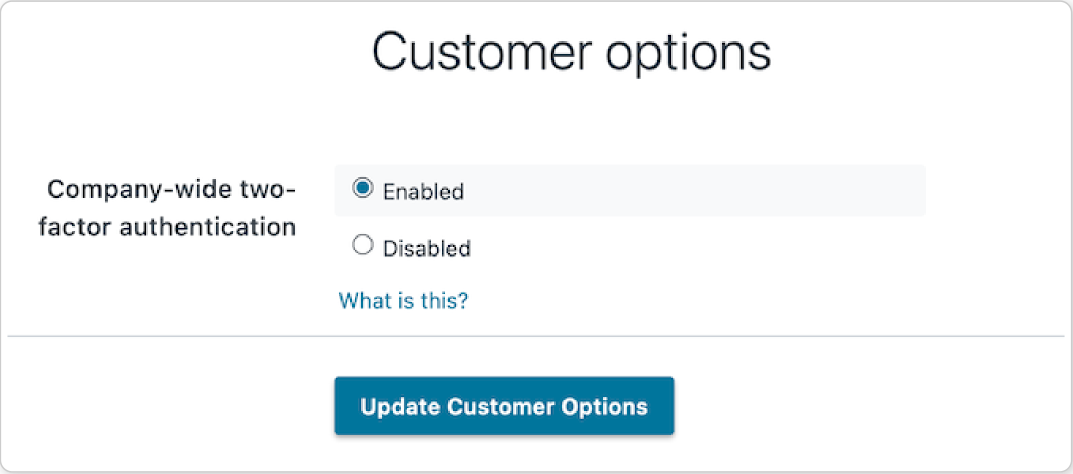
If company-wide two-factor authentication is enabled, a superuser can disable two-factor authentication for a single user's account. This is typically done for user accounts being used for scripts and session authentication. To disable two-factor authentication for a single user's account, follow the steps below.

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Users** area, click the gear next to a user and then select **Ignore 2FA**. A warning message appears.
4. Click **Ignore**. Two-factor authentication will no longer be required for the selected user.

Enabling company-wide two-factor authentication

Users assigned the superuser role can enable this feature on the Account page. To enable company-wide two-factor authentication for all users within your organization, follow the steps below.

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Customer options** area, select **Enabled** from the **Company-wide two-factor authentication** controls.



Customer options

Company-wide two-factor authentication
● Enabled
○ Disabled

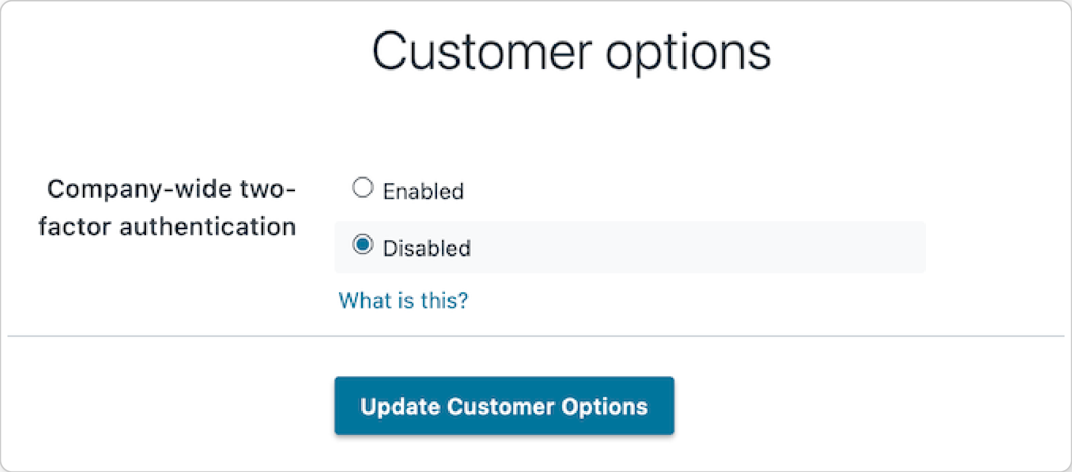
3. Click **Update Customer Options**. A warning message appears stating that login sessions from non-2FA users in your company will immediately expire.
4. Click **Continue**. Two-factor authentication becomes required for all users in your company. Anyone currently logged in and not previously using 2FA on their account will be logged out of the Fastly web interface. Anyone

who has not already enabled two-factor authentication for their account will be prompted to do so the next time they log in to the Fastly web interface.

Disabling company-wide two-factor authentication

A superuser can disable company-wide two-factor authentication. Once this feature is disabled, existing users within the organization will be able to manage their own two-factor authentication settings, and new users will not be required to set up two-factor authentication to log in to the Fastly web interface. To disable company-wide two-factor authentication, follow the steps below:

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. In the **Customer options** area, select **Disabled** from the **Company-wide two-factor authentication** controls.



Customer options

Company-wide two-factor authentication ○ Enabled ● Disabled

3. Click **Update Customer Options**. A warning message appears.
4. Click **Continue**. Company-wide two-factor authentication becomes disabled.

* * *

Monitoring account activity with the audit log

/en/guides/monitoring-account-activity-with-the-audit-log

The audit log keeps track of events related to your Fastly account, users, and services. You can use the audit log to determine which changes were made and by whom. For example, you can use the audit log to review:

- when API tokens were created
- who logged in to your account via the web interface
- what types of changes may have been made to your service configuration settings and by whom (e.g., a user assigned to your account, a Fastly user acting on your behalf)
- when users make changes to their account security settings

You can use the Fastly web interface and API to view the audit log.

TIP

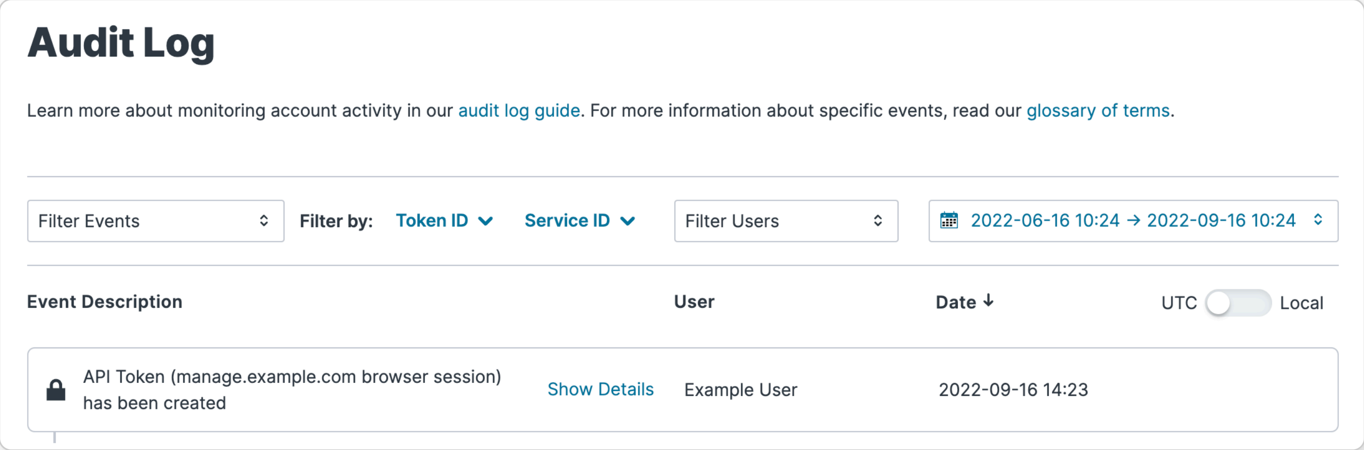
For information on monitoring events related to a service, check out our guide on the event log. For information on monitoring the data that passes through Fastly, check out our guide on Fastly's real-time log streaming features.

Accessing the audit log via the Fastly web interface

You must be assigned the role of superuser to view the audit log. You can filter audit log events by event type, token ID, service ID, user ID, and date directly in the Fastly web interface.

Follow these instructions to access the audit log for your account:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Audit log**.



Audit Log

Learn more about monitoring account activity in our audit log guide. For more information about specific events, read our glossary of terms.

Event Description		User	Date ↓
API Token (manage.example.com browser session) has been created	Show Details	Example User	2022-09-16 14:23

NOTE

As of June 22, 2021, audit log data is retained for a period of one year (365 days).

Accessing the audit log via the Fastly API

The `/events` API endpoint can be used to retrieve the audit log for a Fastly account. For example, you could make the following API call in a terminal application to retrieve a filtered view of all recent audit log events:

```
$ curl -g -H "Fastly-Key: FASTLY_API_TOKEN" "https://api.fastly.com/events?filter[custom
```

The response will look like this:

```
{
  "data": [
      {
          "attributes": {
              "admin": false,
              "created_at": "2016-06-06T20:05:10Z",
              "customer_id": "x4xCwxxJxGCx123Rx5xTx",
              "description": "Version 2 was activated",
              "event_type": "version.activate",
              "ip": "127.0.0.0",
              "metadata": {
                  "version_number": 2
              },
              "service_id": "SU1Z0isxPaozGVKXdv0eY",
              "user_id": "x9KzsrACXZv8tPwlEDsKb6"
          },
          "id": "5IH1QmNSV1Qi7jXc4oIZlZc",
          "type": "event"
      }
  ],
  "links": {
    "last": "https://api.fastly.com/events?filter[customer_id]=x4xCwxxJxGCx123Rx5xTx&pa
  }
}
```

Check out the Fastly API documentation for more information.

* * *

Reviewing service activity with the event log

/en/guides/reviewing-service-activity-with-the-event-log

Event logs keep track of events related to a service. With the appropriate user role and permissions you can use event logs to determine which service-level changes were made and by whom. For example, you can use them to see who activated the most recent version of a service and view recent service configuration setting changes.

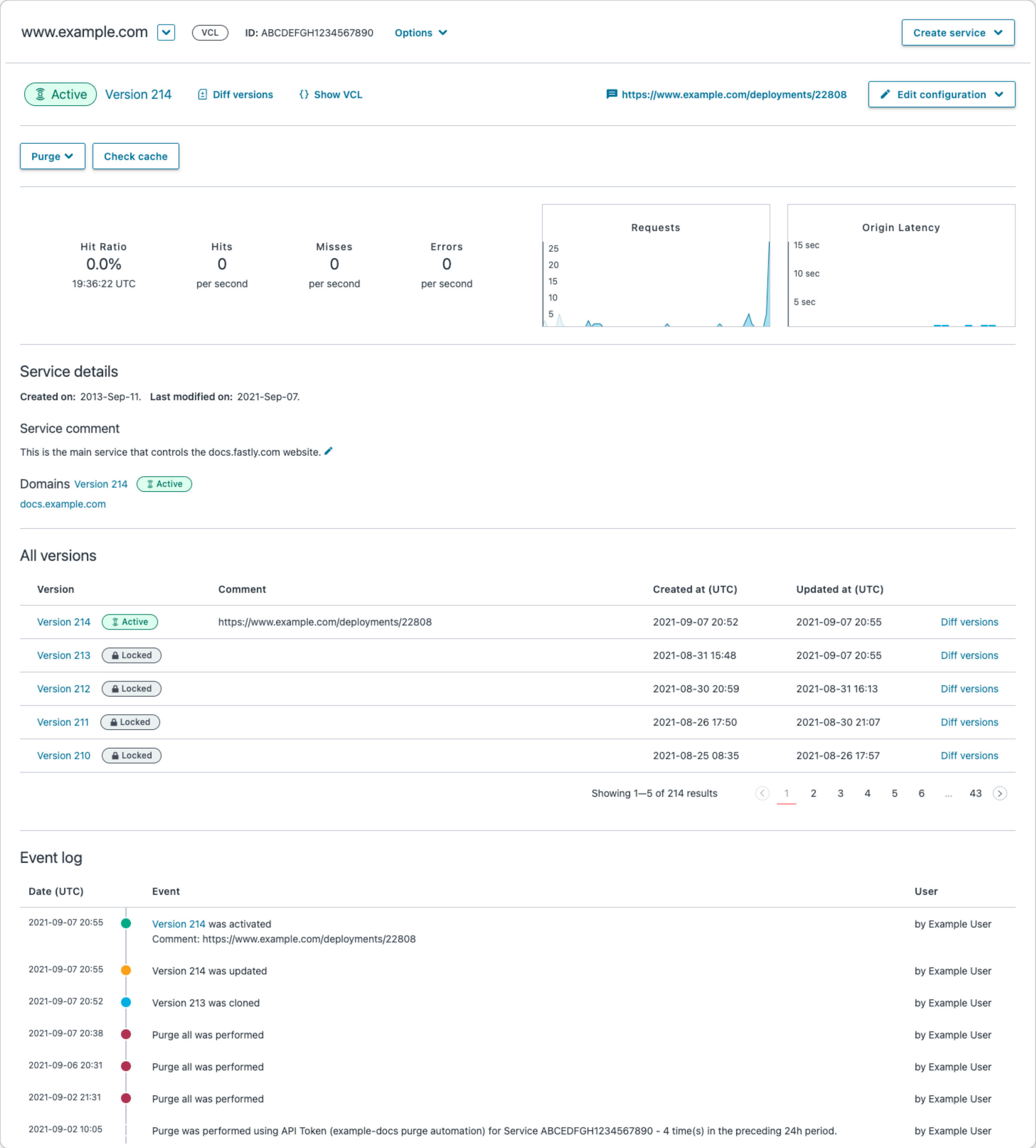
TIP

For information on monitoring Fastly account activity, check out our guide on the audit log. For information on monitoring the data that passes through Fastly, read our guide on Fastly's real-time log streaming features.

Follow these instructions to access the event logs for a service:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

The last 20 service-related events are displayed near the bottom of the page, in the Event log area.


www.example.com
VCL
ID: ABCDEFGH1234567890
Options
Create service
Active
Version 214
Diff versions
Show VCL
https://www.example.com/deployments/22808
Edit configuration
Purge
Check cache
Hit Ratio
0.0%
19:36:22 UTC
Hits
0
per second
Misses
0
per second
Errors
0
per second
Requests
25
20
15
10
5
Origin Latency
15 sec
10 sec
5 sec

Service details

Created on: 2013-Sep-11. **Last modified on:** 2021-Sep-07.

Service comment

This is the main service that controls the docs.fastly.com website.

Domains Version 214 Active

docs.example.com

All versions

Version		Comment	Created at (UTC)	Updated at (UTC)	
Version 214	Active	https://www.example.com/deployments/22808	2021-09-07 20:52	2021-09-07 20:55	Diff versions
Version 213	Locked		2021-08-31 15:48	2021-09-07 20:55	Diff versions
Version 212	Locked		2021-08-30 20:59	2021-08-31 16:13	Diff versions
Version 211	Locked		2021-08-26 17:50	2021-08-30 21:07	Diff versions
Version 210	Locked		2021-08-25 08:35	2021-08-26 17:57	Diff versions

Showing 1—5 of 214 results 1 2 3 4 5 6 ... 43

Event log

Date (UTC)	Event	User
2021-09-07 20:55	Version 214 was activated Comment: https://www.example.com/deployments/22808	by Example User
2021-09-07 20:55	Version 214 was updated	by Example User
2021-09-07 20:52	Version 213 was cloned	by Example User
2021-09-07 20:38	Purge all was performed	by Example User
2021-09-06 20:31	Purge all was performed	by Example User
2021-09-02 21:31	Purge all was performed	by Example User
2021-09-02 10:05	Purge was performed using API Token (example-docs purge automation) for Service ABCEDFGH1234567890 - 4 time(s) in the preceding 24h period.	by Example User

NOTE

As of June 22, 2021, event log data is retained for a period of one year (365 days).

* * *

Troubleshooting your login

/en/guides/troubleshooting-your-login

If you're having trouble logging in to https://manage.fastly.com/ or viewing features or data, refer to this page.

I can't log in

If you're having trouble logging in to Fastly, check if any of the following are true:

- **There are outages or scheduled maintenance.** Check the Fastly status page to make sure there are no outages or scheduled maintenance currently affecting the Fastly app.
- **Your email and password are incorrect.** Make sure you're using the correct email for your account, especially if you have multiple accounts. If you've forgotten your password, learn how to reset it.
- **You're logging in at the wrong URL.** If you have a Fastly account or both Fastly and Signal Sciences accounts, use https://manage.fastly.com/ to log in. If you only have a Signal Sciences account use https://dashboard.signalsciences.net/ to log in.
- **Your account may be locked.** For security reasons, Fastly limits the number of times someone can try logging in to an account. Refer to Account lockouts for more information.

I don't see the features or data I'm looking for

Information that appears in the Fastly web interface depends on several things, including the access you've been granted. If you don't have access to what you're looking for or can't see something you're expecting to see, check if any of the following are true:

- **Your account ID has incorrect permissions assigned to it.** The user role and permissions assigned to your account ID for your Fastly app access or your Next-Gen WAF access affect what you can do in the web interface. Check with a superuser on your account if your access appears to be incorrect.
- **Your login session has timed out.** Once logged in to your account, your session can potentially remain active for up to 12 hours without requiring reauthentication. This depends on the session timeout settings for your Signal Sciences corp and whether or not SSO is enabled for either your Signal Sciences account or Fastly account. By design, session timeouts cannot exceed 12 hours. All users will be logged out automatically after 3 hours of inactivity.
- **You're signed in to a different account or control panel.** If you have both Fastly and Signal Sciences accounts you may need to [switch application control panels](/en/guides/logging-in-and-using-fastly#switching-between-application-control panels) to see the correct information. Likewise, if you have multiple Fastly accounts, make sure you sign in using the account that has the data you're looking for.
- **You're accessing a different Fastly service.** Every service tied to your Fastly account contains a unique ID that can help you determine if you're looking at the right information. Our guide on services provides more information that can help you switch services or search for the correct one.
- **You're looking at a different site in your Signal Sciences corp.** Be sure you're looking at the right site within your corp. Our web interface controls can help you select different sites from the corp navigation bar.

- **The product or feature isn't enabled for your account.** Some Fastly products and features (e.g., Image Optimizer) may be disabled by default but can be enabled directly in the web interface by anyone with the appropriate roles and permissions. If you can't access a product or feature you think you should have, check with a superuser on your account to make sure your permissions are properly set. If you still can't access a product or feature, contact support for assistance or contact sales@fastly.com to start a conversation about which features may be best for your account.

I still can't log in or use Fastly the way I expect

If you continue to have issues logging in or using Fastly, contact Fastly Support.

* * *

Using API tokens

🔗 /en/guides/using-api-tokens

API tokens are unique security credentials that allow human users and automated systems to prove their identity to Fastly, thereby indicating they can be trusted to access restricted resources and perform specific, restricted operations via the Fastly API. Use the API tokens page to add, view, and delete API tokens.

About API tokens

There are two types of API tokens: user tokens and automation tokens. Each type can optionally have their capabilities limited by controlling the specific scope of their activities.

Token types

There are two types of API tokens. Your ability to manage them is based on the roles and permissions you have been assigned.

- **Automation tokens.** Automation tokens provide security credentials for non-human clients (e.g., continuous integration and build systems) that need to conduct automated API activities like continuous integrations, deployment pipelines, and routinely scripted tasks. Automation tokens may sometimes be referred to as "account tokens" or "service account tokens."

  Only superusers can create automation tokens. These tokens are not tied to a specific human user and therefore can remain active indefinitely. They only appear as part of a company's profile settings and can only be managed by human users assigned the role of superuser.

- **User tokens.** User tokens provide security credentials for API activities initiated by human users. They are associated with a specific human and are only active for the lifetime of that user's account. User tokens may sometimes be referred to as "personal tokens" or "personal API tokens."

  Anyone can create and view their own user tokens. These tokens carry the same permissions as the user who would be performing account-based actions via the API. For example, if you are a billing user, then your user token will only allow you to perform the capabilities assigned to the billing role. When a user's account is deleted, active API tokens must be revoked.

Token scopes

You can limit the capabilities of API tokens by specifying the scope of their service-related activities. Specifically, you can allow or limit API tokens as follows:

- **Global API access (`global`)** allows the token full control over a service with access to all API endpoints, including purging.
- **Purge full cache (`purge_all`)** allows the token purging ability for an entire service via a `purge_all` API request.
- **Purge select content (`purge_select`)** allows the token purging ability via Surrogate-Key and URL but does not include the ability to purge all cache.
- **Read-only access (`global:read`)** allows the token read-only access to account information, configuration, and stats.

Limitations and best practices

When managing and using API tokens, keep in mind the following limitations:

- **API tokens can only be created, viewed, and deleted.** They cannot be edited or updated.
- **Each user is limited to 100 active API tokens.** Deleted and expired tokens don't count against the limit.
- **Unused tokens do not last forever.** API tokens that remain unused for two years are automatically deleted even if they have been set to never expire.
- **Your role and its permissions change what you see in the event log.** The user roles and permissions assigned to you control exactly what you can and can't view in the event log when API token work appears there. Specifically, if you've been assigned the `user` role, you can only see events from your own actions. If you've been assigned the `engineer` role, you can see all events related to your own actions plus all the actions on services to which you've been granted access.

When creating API tokens, also keep the following best practices in mind:

- **Keep it secret. Keep it safe.** When you generate a new token, you should store it in a protected place like a password manager to keep it secret and safe. For security reasons, you will only be able to copy tokens once, at the time of creation. You won't be able to retrieve token strings later.
- **Consider implementing minimal privileges.** Limiting a token's service access, controlling its scope, and setting an expiration date restricts that credential's access can minimize the risk of damage if security credentials are somehow compromised. For more information, review the principle of least privilege.

Creating API tokens

To create an API token, follow the steps below:

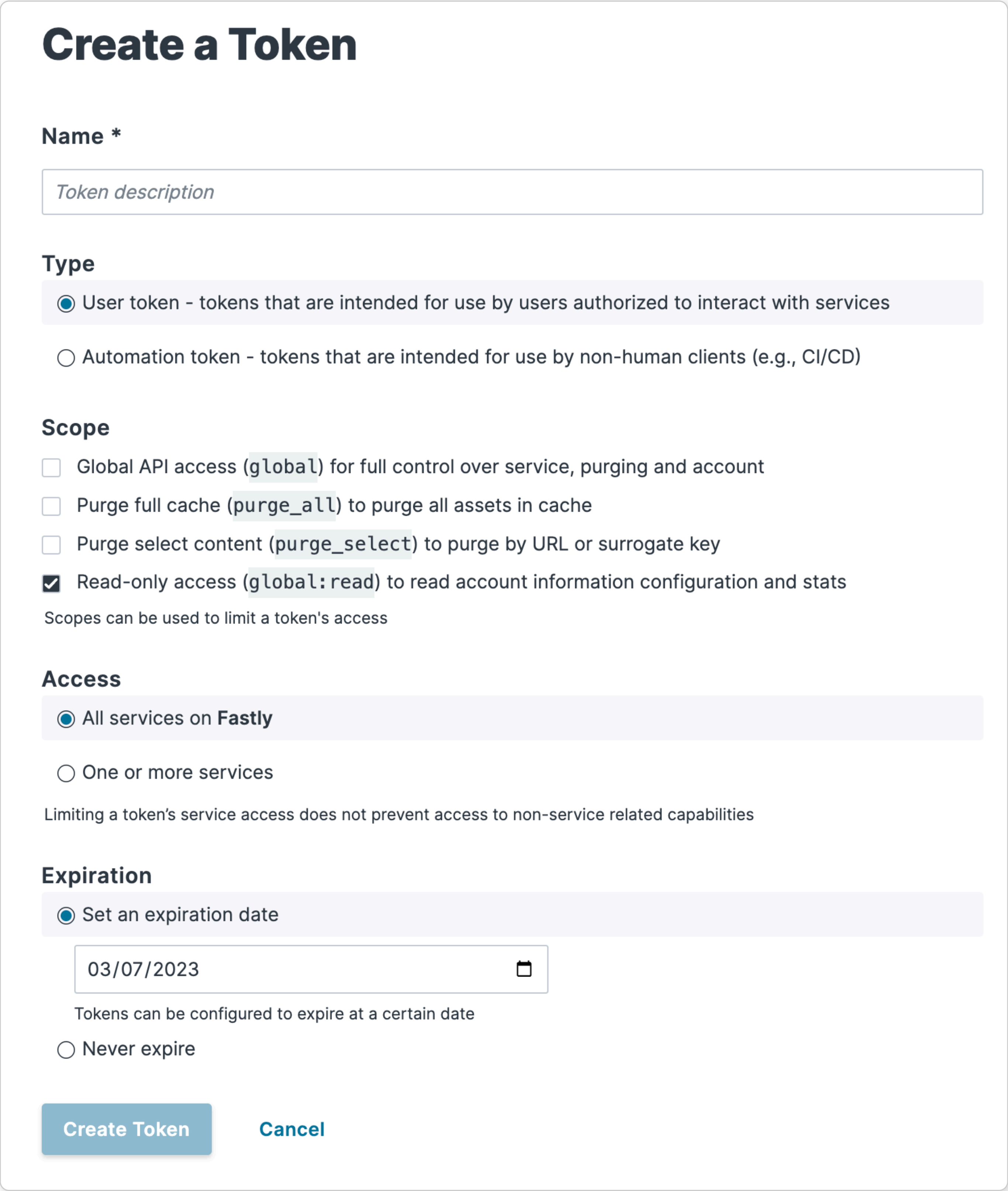
1. Log in to the Fastly web interface.

   > IMPORTANT
   >
   > If you're creating a user token, be sure you're creating tokens for the right account. If you've been invited as a user on multiple accounts, you'll need to switch to the appropriate account first.

2. Go to **Account** > **Personal profile** > **API tokens**.

3. Click **Create token**.

4. When prompted, enter your password to re-authenticate your permissions.



Create a Token

Name *

Token description

Type

● User token - tokens that are intended for use by users authorized to interact with services

○ Automation token - tokens that are intended for use by non-human clients (e.g., CI/CD)

Scope

☐ Global API access (`global`) for full control over service, purging and account

☐ Purge full cache (`purge_all`) to purge all assets in cache

☐ Purge select content (`purge_select`) to purge by URL or surrogate key

☑ Read-only access (`global:read`) to read account information configuration and stats

Scopes can be used to limit a token's access

Access

● All services on **Fastly**

○ One or more services

Limiting a token’s service access does not prevent access to non-service related capabilities

Expiration

● Set an expiration date

03/07/2023

Tokens can be configured to expire at a certain date

○ Never expire

5. Fill out the **Create a Token** fields as follows:

   - In the **Name** field, enter a descriptive name for the token that indicates how or where it will be used.

- From the **Type** options, optionally select the type of API token to create: `User token` or `Automation token`. Only superusers have the ability to create automation tokens. If you select `Automation token`, controls to specify the user role for that token appear.
- From the **Role** options, select the user role that will assign the appropriate access permissions to the API token. Available options are `Engineer`, `User`, and `Billing`. Our guide to configuring user roles and permissions provides more information.
- *(Optional) Select **TLS management** to grant the token the ability to modify TLS configurations across all services, including TLS certificates and domains.
- From the **Scope** options, select one or more checkboxes to limit the token's access to a specific scope. Only the Scope options applicable to the selected role will be selectable. Our guide to configuring user roles and permissions provides more information.
- From the **Access** options, select either all services or limit the token's access to a specific service or group of services by selecting them from the **Search or select service** menu.
- From the **Expiration** options, set the token expiration timeframe. By default the web interface will set the expiration date to 90 days from the date on which you create it. You can, however, set a token to never expire or you can select a specific date on which it expires.

> ✓ TIP
>
> After a token expires, using it for any request will return an HTTP 401 response.

6. Click **Create Token** to create the new token. A new token and its creation notification appears. This is the credential you'll use to authenticate via the Fastly API. You may use the same token for multiple applications.

7. Click the clipboard to copy the API token string so you can store it in a safe, secret location.

> ⚠ WARNING
>
> This is the only time your API token string will be visible. Be sure to immediately copy it and store it in a safe location. It will never be visible again.

8. Click **Okay**.

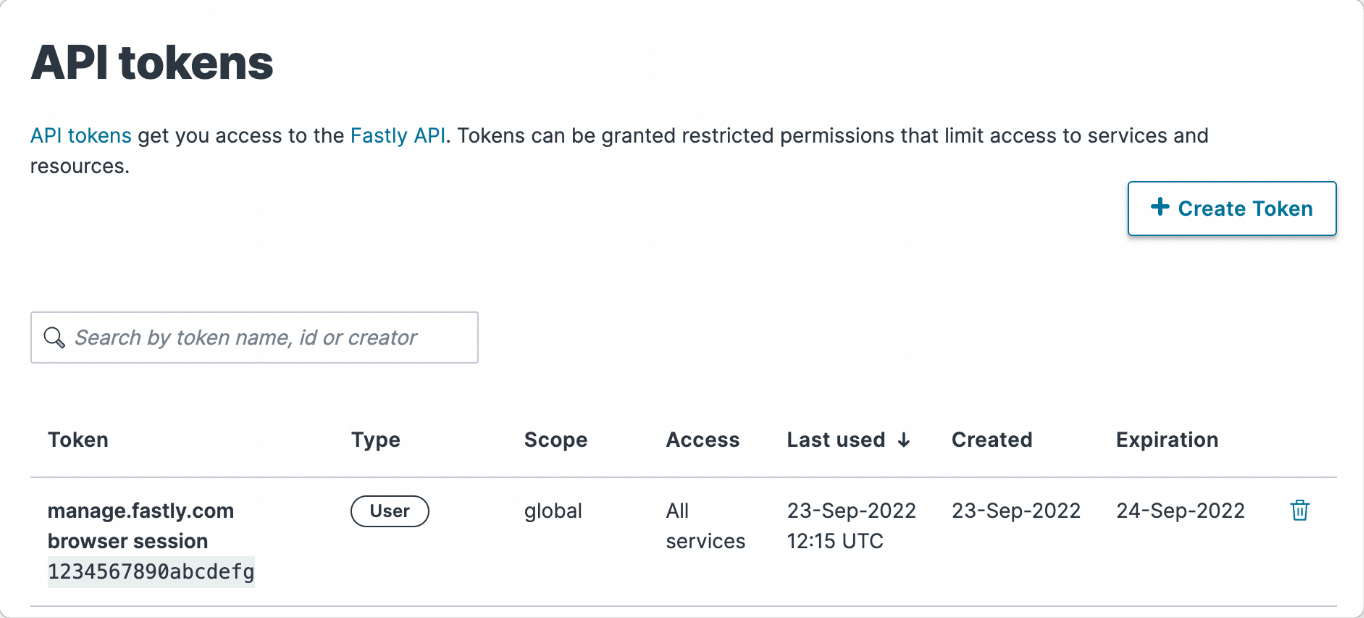
Viewing API tokens

To view API tokens, follow these steps.

Viewing user tokens

To view your personal user tokens, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **API tokens**. The API tokens page appears with a list of your personal tokens.



API tokens

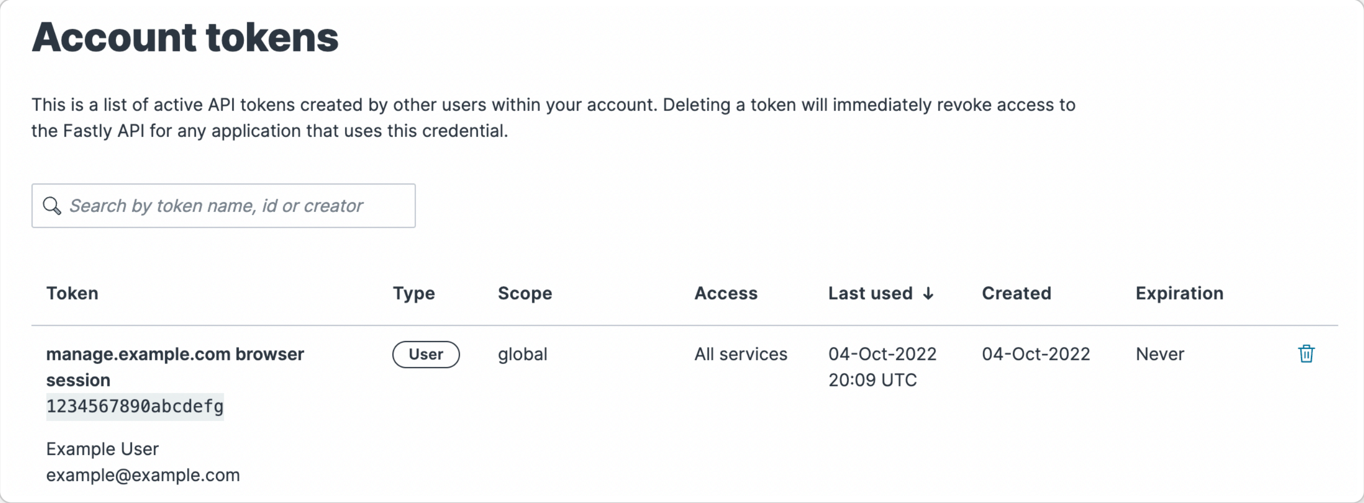
API tokens get you access to the Fastly API. Tokens can be granted restricted permissions that limit access to services and resources.

Token	Type	Scope	Access	Last used ↓	Created	Expiration
manage.fastly.com browser session `1234567890abcdefg`	User	global	All services	23-Sep-2022 12:15 UTC	23-Sep-2022	24-Sep-2022

Viewing account tokens

If you've been assigned the role of superuser, view account tokens using these steps:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Account tokens**. The Account Tokens page appears with a list of tokens associated with your organization's Fastly account.



Account tokens

This is a list of active API tokens created by other users within your account. Deleting a token will immediately revoke access to the Fastly API for any application that uses this credential.

Token	Type	Scope	Access	Last used ↓	Created	Expiration
manage.example.com browser session `1234567890abcdefg` Example User example@example.com	User	global	All services	04-Oct-2022 20:09 UTC	04-Oct-2022	Never

Deleting API tokens

> ⚠ WARNING
>
> Deleting an API token will break any integration actively using that credential.

To delete user or automation tokens, follow these steps.

Deleting user tokens

To delete a user token, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **API tokens**.
3. Find the token you want to delete and click the trash.
4. Click **Delete** to permanently delete the user token.

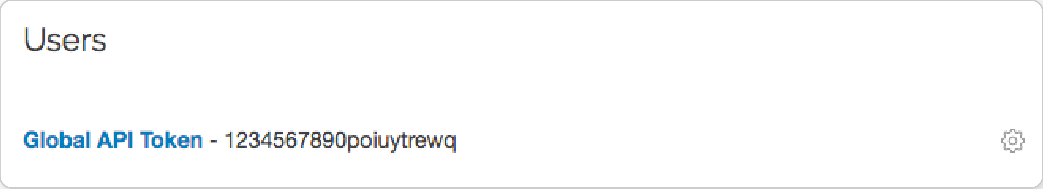
Deleting account tokens

To delete an account token or to revoke another user's token as a superuser, follow the steps:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Account tokens**.
3. Find the token you want to delete and click the trash.
4. Click **Delete** to permanently delete the token.

Legacy API credentials

If you created a Fastly account before May 15th, 2017, you may have used an API key (or multiple API keys) to authenticate API requests. This account-level credential was migrated to an API token with a `global` scope and access to all of your services. It was assigned to a newly created, synthetic user with the name `Global API Token`.



Users

Global API Token - 1234567890poiuytrewq

* * *

Category: Billing

These articles describe Fastly's billing and payment plans and how to make adjustments to your billing information.

About the charges on your bill

/en/guides/about-the-charges-on-your-bill

We bill you as specified in your applicable ordering document. Unless otherwise specified, we will bill you monthly according to that month's use of Fastly's services.

NOTE

Billing limits for this product may be different if you've purchased a package offering or are using a product or feature trial.

About the monthly minimum charges

We bill a minimum of $50 per month so we can fully support all of our customers. This is the minimum price you'll pay in any month once you've completed your testing trials.

For example, say that you're done testing Fastly's services and you've begun to push production-level traffic through Fastly. If most of your site's traffic for the current month is in North America and Europe and your site uses 10GB of traffic over 10 million requests, the combined bandwidth and request charges would be $8.70 for the month. Because this amount falls below the $50 monthly minimum (you'll see this appear as a "commit shortfall" on your bill), we would charge you $50 for that month, not $8.70.

Bandwidth and request prices for some billing regions are slightly higher. If most of your site's traffic were in these other regions instead, then at the above traffic levels your bandwidth and request usage charges would still fall below the monthly minimum and we would charge you $50 for that month.

NOTE

If you're using Fastly for content delivery via Heroku's cloud development services, read Heroku's add-ons pricing plan for additional details.

When we charge you for Fastly services

Fastly typically bills in arrears, not in advance, meaning that we bill you for services after you've used them, not before. For example, if you sign up for and start using Fastly services in January, the bill you receive in February reflects January's charges and services, your March bill reflects February's charges and services, and so forth.

How account cancellation affects your bill

If you ever cancel your account, you'll be billed for any outstanding charges accrued through the day you canceled, or at least the monthly minimum, whichever amount is greater.

Estimating your month-to-date bill

You can estimate your month-to-date (MTD) bill via the web interface or via the API.

Via the web interface

To view an estimated report of account usage for the current partial month, use any standard web browser to log in to your Fastly account and navigate to:

https://manage.fastly.com/account/invoices/month-to-date

NOTE

A small number of billing plans cannot be calculated month-to-date and only include an end-of-month generated invoice. If you have one of these billing plans, the web interface will clearly tell you that you can’t see the report due to your account’s status.

Via the API

As part of our API, a billing endpoint exists to generate a report of your usage for the current partial month (known as month-to-date, or MTD). Full details of this endpoint's output format can be found in our Billing API documentation. Generating a report via API usually takes only a few seconds, but can potentially take up to 60 seconds. During this time, the API call will return an HTTP `202 Accepted` response.

```
{
  "data" : {
     "attributes" : {
        "status" : "Pending: waiting for another process"
     },
     "id" : "MTD_2i0wWA8Zvo6uUpmATZYuQi",
     "type" : "mtd-invoice-pending"
  }
}
```

* * *

Account types

/en/guides/account-types

Fastly offers a variety of account types, which we detail below.

Trial accounts

We provide several ways to give Fastly a try before you purchase anything. Use a trial account to get hands-on experience exploring the capabilities of our platforms.

Trials for delivery services

We offer a development trial that allows you to test our delivery services by simply signing up. We allow you to test up to $50 of traffic per month for free to ensure everything fits your requirements. You can pay as you go from there by switching to a paid account directly in the web interface. Keep in mind that some add-on options (bringing your own TLS certificates, for example) require you to switch your account to a paid account before that functionality becomes available to you.

Trials for Compute services

We offer a trial of Compute that allows you to gain experience working with Fastly's serverless edge platform. We allow you to test up to $50 of Compute charges. Once you've signed up for a Fastly account, start your trial by navigating to the Compute tab in the web interface and then following the on-screen prompts. Use the trial to test how serverless application concepts created in your non-production environment can be deployed to the Fastly Edge.

Limitations on trials

Keep in mind the limitations that apply to all trials:

- Trial accounts are not designed for use with production traffic or workloads.
- Trials do not include customer support for Compute issues, nor are the service availability SLAs applicable. Only paid accounts include this support and the associated SLAs.
- Specific resource allocation limitations apply during your Compute trial. Compute has a lower resource allocation that can only be increased or removed if you upgrade to a paid account, at which point normal limitations and constraints apply. Contact sales@fastly.com for more information.

Paid accounts

You can easily sign up for a paid Fastly account via our web interface or by contacting us at sales@fastly.com.

Signing up for a paid account online

You can use Fastly's delivery services on a month-to-month basis across our global regions. Start by signing up for a trial, explore a bit, and then, when you're ready to start pushing production traffic our way, upgrade your account directly in the web interface.

Once you switch to a paid account, the developer account trial option disappears and we'll begin billing you automatically at the end of every month using your credit card information. You can estimate your monthly delivery charges by using the pricing estimator on our pricing page. For more detailed information on those monthly charges, including how we measure your usage and when we charge you for it, see our guide to how we calculate your bill.

Contact us at sales@fastly.com for more information if you plan to push at least 2TB of data per month for delivery and require one of our TLS service options, if you plan to push a minimum of 4TB of data per month, or if you want to purchase our Gold or Enterprise support offerings. We also offer solutions targeted to the needs of specific industries.

Other paid accounts

If you signed up for a paid Fastly account via a method other than our web interface, refer to your contract with Fastly for specific details about your services.

Fastly's open source and non-profit program

We provide free services to members of our open source and nonprofit program. If you are interested in participating in the program, contact us at community@fastly.com to get started.

Upgrading your account

NOTE

To upgrade your account for Compute services, contact sales@fastly.com.

You can upgrade your account for delivery services directly in the web interface by following these steps:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Billing** > **Billing information**.
3. Enter your billing address information in the fields provided.
4. Click **Save and continue**

5. Enter your credit card information in the fields provided.

6. Review the terms of the purchase agreement and click **Complete purchase** to upgrade your account.

> TIP
>
> Fastly never sees your credit card number. All transactions are handled by our fully PCI compliant payment gateway and their privacy policy can be found at https://www.worldpay.com/en/privacy/online-privacy-notice.

Canceling your account

You can cancel your account at any time. Have your account owner follow these steps:

1. Deactivate and then delete all services on your account.

2. If you’ve purchased one of Fastly's hosted or managed certificate options, contact support to begin the process of deleting your certificates.

3. Go to **Account** > **Company settings**.

4. In the **Company settings** area, click **Cancel Account**.

5. In the **Your password** field of the confirmation window, enter the password associated with your account and click **Confirm and Cancel**.

After your account is canceled, you'll be billed for any outstanding charges accrued through the day you canceled. For questions about your final billing statement, contact our billing team for assistance. If you decide at a later date to reactivate your account, contact support and request reactivation.

* * *

How we calculate your delivery bill

/en/guides/how-we-calculate-your-delivery-bill

Your bill for content delivery and streaming delivery services is affected by a combination of things including the actual traffic Fastly has served on your behalf, the products you've purchased, the features you've enabled, and the specific configuration settings you've chosen (like enabling compression).

> NOTE
>
> Billing limits for this product may be different if you've purchased a package offering or are using a product or feature trial.

We charge for egress traffic from our POPs, including traffic served to end users and, if shielding is enabled, traffic served from the shield POP to other POPs. Specifically, we charge for each response and for the size of the response (which includes the header and body). Each response is billed as a single request, and the response size in bytes is billed as bandwidth. We charge for bandwidth and requests for content delivered to clients from the CDN and for bandwidth for traffic sent from the CDN to our customers' origins.

> NOTE

If you're using Anycast IP addresses, these IPs use our global network and will route a request to the nearest POP located in a billing region that may charge a higher rate. Our billing regions can be found on the Fastly Pricing page. We announce new billing regions regularly via our service status page.

Charges for any options you've chosen are applied in addition to the bandwidth and request usage we charge for normal content delivery and streaming and may be adjusted according to any packaged offering you've purchased. In accordance with local laws, Fastly may also collect sales tax based on your shipping or billing address on file.

About the measurements and calculations we use

We measure months according to Coordinated Universal Time (UTC). For usage-based charges, bandwidth is recorded in bytes and presented in gigabytes (GB), and requests are recorded individually and presented in units of 10,000.

Fastly uses The International System of Units (SI Units) to measure bandwidth. In our calculations, 1 gigabyte (GB) = 10^9 (1,000,000,000) bytes, 1 terabyte (TB) = 10^{12} bytes (or 1,000 GB), and 1 petabyte (PB) = 10^{15} bytes (or 1,000 TB). Your invoice shows your usage and that matches the usage shown on the Observability page.

* * *

Managing account plans and usage details

🔗 /en/guides/managing-account-plans-and-usage-details

If you've been assigned a superuser or billing role, you can manage your account plans, including upgrading to a paid account, and review usage details and the associated charges by selecting **Billing** from the user quick links menu at the top right of any page.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there. Pay particular attention to the Billing menu.

Monitoring bill-related metrics

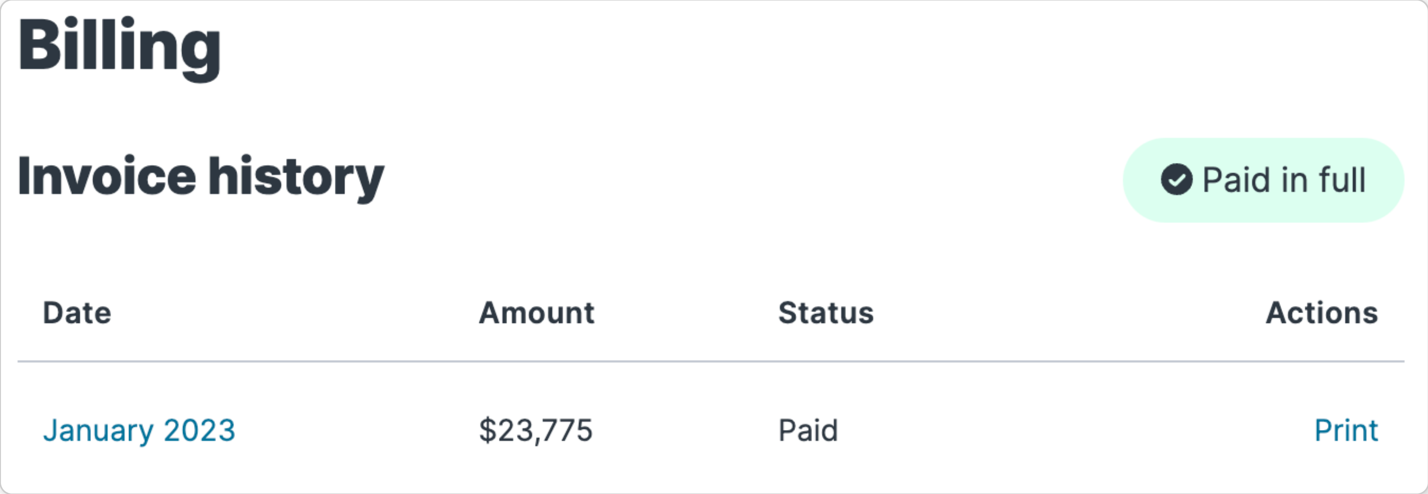
For paid-accounts, the billing overview page allows you to view bill-related metrics for your account broken down by product and region over key, monthly timeframes. These metrics provide insights into what you're being charged for by Fastly and how that amount applies to the various products you've purchased and the regions in which traffic to your services operates.

Monitoring account and service usage metrics

View your account usage metrics broken down by product and region over key, monthly timeframes by selecting **Plan usage** from the Billing menu. These metrics provide insights into the month-to-date usage of things your account's billing is based on (for example, Image Optimizer requests or Concierge TLS). Use the information here to help you estimate your bill and plan for the future.

Reviewing account charges and invoice details

View your account invoice information by selecting **Invoices** from the Billing menu. By default, the current balance for your account, if any, appears at the top, followed by your invoice history.



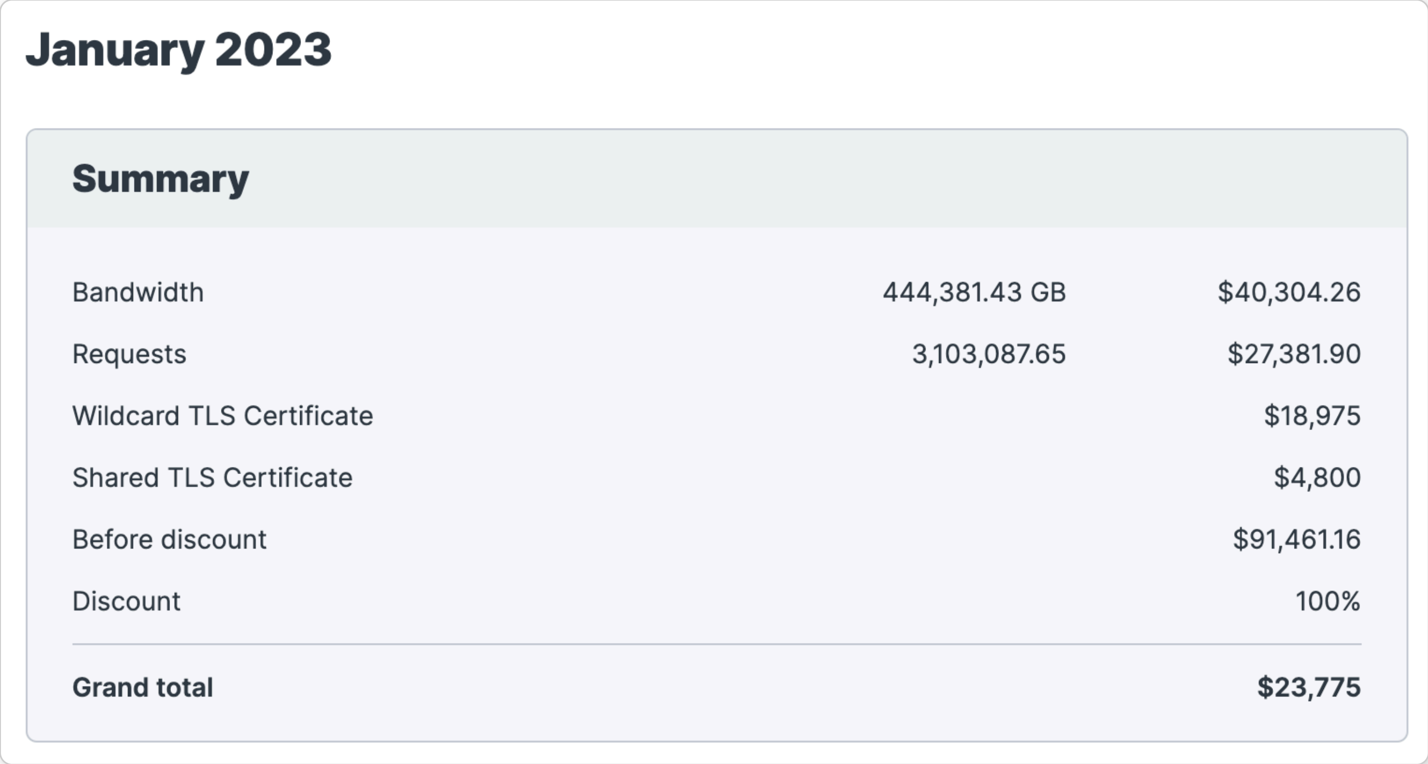
Billing

Invoice history

✓ Paid in full

Date	Amount	Status	Actions
January 2023	$23,775	Paid	Print

Clicking on the linked date of any invoice displays a summary of charges for that month. The billing invoice summary includes the overall bandwidth you used and the associated charges, followed by the charges you incurred for requests. The bottom of the summary displays the grand total dollar amount owed for the dated month.



January 2023

Summary

Bandwidth	444,381.43 GB	$40,304.26
Requests	3,103,087.65	$27,381.90
Wildcard TLS Certificate		$18,975
Shared TLS Certificate		$4,800
Before discount		$91,461.16
Discount		100%
Grand total		**$23,775**

Below the month's summary on the invoice, we include regional bandwidth and request details for each billing region. The bottom of each regional details section includes the total charge for bandwidth and requests for that region alone for the dated month.

NOTE

A breakdown of billing charges per service is not available. Check out our historical stats API for data on unrated request and bandwidth used by a service, aggregated by billing region.

You can print account use details for any month by finding that month in the invoice history and clicking **Print** in the **Actions** column for that month.

Upgrading to a paid account

If you are on a trial account for delivery services, you can upgrade to a paid account from the **Billing information** page, if you have the appropriate permissions to do so.

NOTE

To upgrade your account for Compute services, contact sales@fastly.com.

What's next

Continue to explore your account management and billing information.

* * *

Paying your bill

/en/guides/paying-your-bill

At the end of each month, your account's billing contact will be sent an email summarizing your current usage levels and the charges your account incurred for the month. The email contains a link to an online copy of the related invoice.

You'll need both a valid credit card and current billing address when you switch to a paid, month-to-month account. Once your invoice gets generated, your credit card is automatically charged for the full, outstanding balance.

Changing your credit card information or billing address

To change your billing address or the information for the credit card we use for automatic billing, follow the steps below:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Billing** > **Billing information**.
3. In the **Billing details** area, click **Edit**.
4. Make any necessary changes to the payment information in the fields provided.

TIP

Fastly never sees your credit card number. All transactions handled by our payment gateway are fully PCI compliant. The gateway privacy policy can be found at https://www.worldpay.com/en/privacy/online-privacy-notice.

5. Click **Save payment information**.
6. Make any necessary changes to the billing address in the fields provided.
7. Click **Save billing address**.

Changing who receives your bill

By default, your account owner is considered your billing contact and will receive your bill for Fastly services. To change who receives your bill or to add multiple email addresses for several billing contacts, contact billing@fastly.com with the addresses you'd like to send invoices to.

IMPORTANT

Invoices are only sent to the email addresses of the account owner or the billing contact. Invoices are not sent to every user assigned a billing role.

* * *

Category: User access and control

These articles describe how to manage users with permission to access to your account.

Adding and deleting user accounts

🔗 /en/guides/adding-and-deleting-user-accounts

Fastly allows superusers to add users to an existing Fastly account via invitation, assigning them different roles and permissions as appropriate. You can delete user accounts when you no longer want someone to have access.

TIP

Need to add users to Signal Sciences accounts? Our guide to managing users of the Next-Gen WAF provides details.

Adding account users

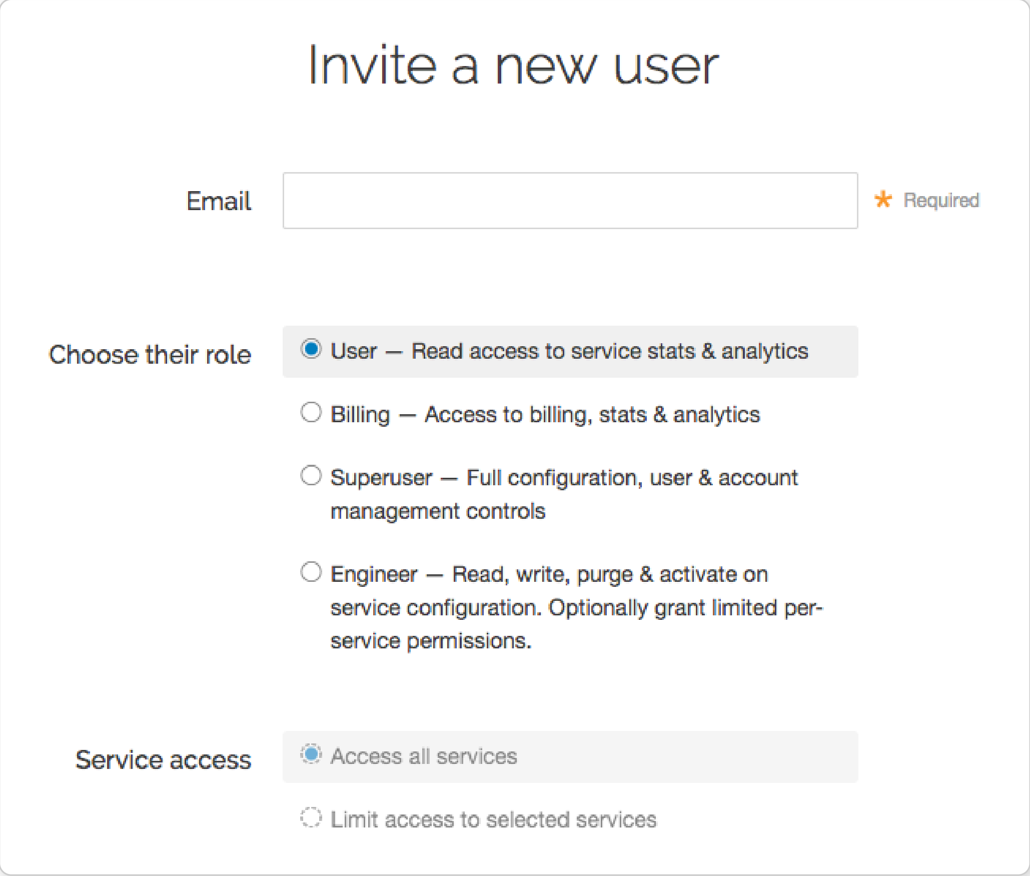
TIP

Adding a new user to make them the billing contact for a Fastly account? Follow our billing contact instructions instead.

Adding a user to an account

To add a Fastly user to an account, send them an invitation to join following the steps below:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Pending user invitations** area, click **Invite a user**.



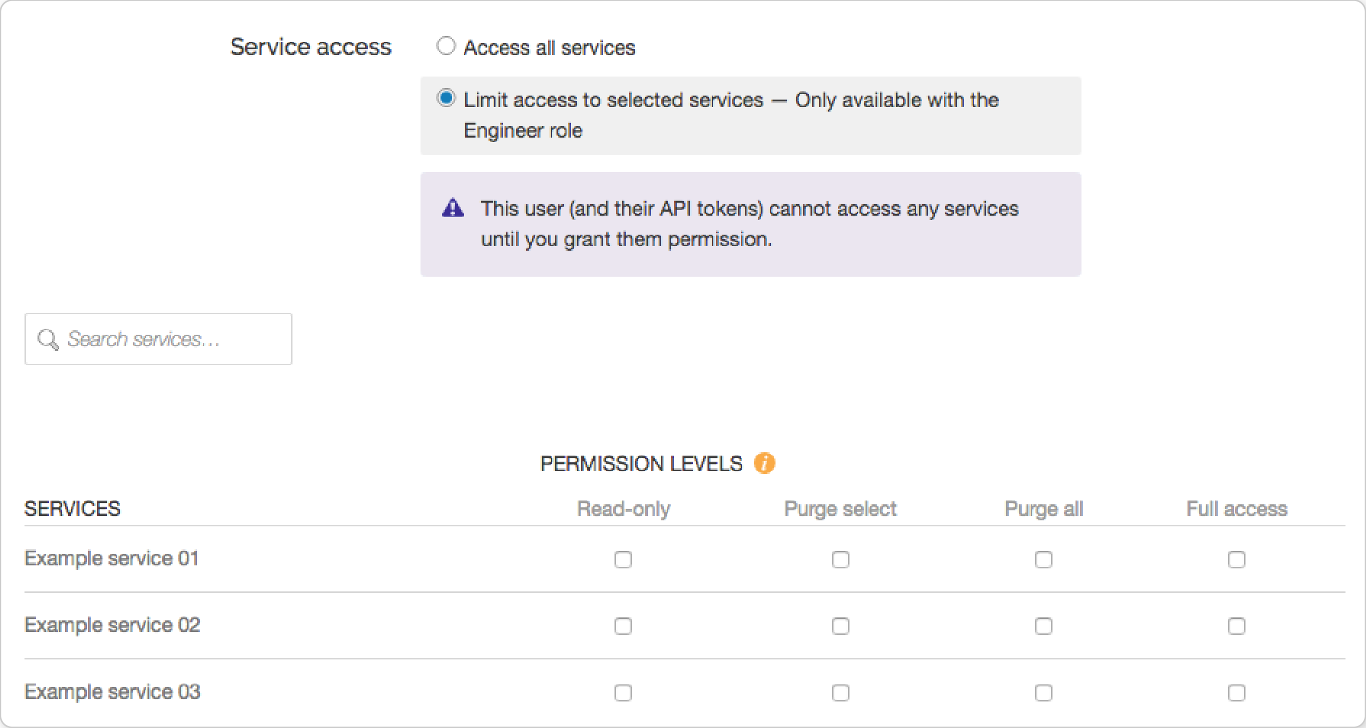
Invite a new user

Email ☐ * Required

Choose their role
● User — Read access to service stats & analytics
○ Billing — Access to billing, stats & analytics
○ Superuser — Full configuration, user & account management controls
○ Engineer — Read, write, purge & activate on service configuration. Optionally grant limited per-service permissions.

Service access
● Access all services
○ Limit access to selected services

4. In the **Email** field, enter the email address of the user to invite.
5. From the **Choose their role** options, select the role to assign the user once they accept the invitation.
6. *(Optional)* From the **Service access** controls, select **Limit access to selected services** to limit access to selected services for users assigned the role of engineer.



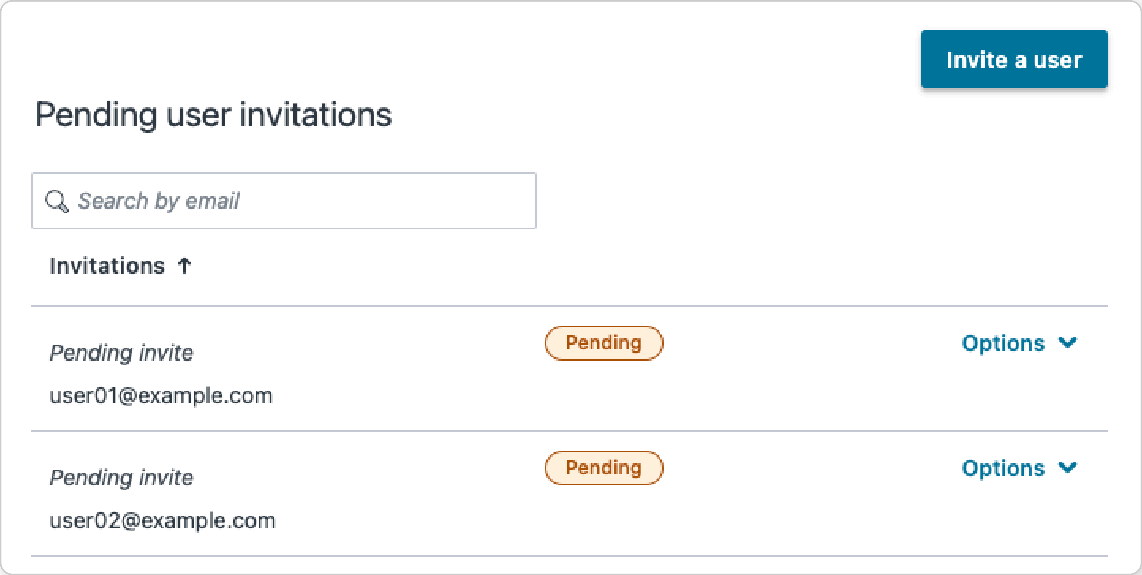
Service access
○ Access all services
● Limit access to selected services — Only available with the Engineer role

This user (and their API tokens) cannot access any services until you grant them permission.

PERMISSION LEVELS

SERVICES	Read-only	Purge select	Purge all	Full access
Example service 01	☐	☐	☐	☐
Example service 02	☐	☐	☐	☐
Example service 03	☐	☐	☐	☐

7. If you've chosen to limit access to selected services for a user assigned the role of engineer, select the specific permission levels for each service associated with the account.
8. Click **Invite** to send an invitation to the email you specified. The email address of the user you invited appears in the Pending user invitations area and remains there until the invitation is accepted or seven calendar days have passed.



Pending user invitations

Pending invite Pending
user01@example.com

Pending invite Pending
user02@example.com

Resending an account invitation

To resend a pending invitation before it has been accepted or expired:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Pending user invitations** area, find the email address associated with the original invitation.

4. From the **Options** menu, select **Resend email**.

Deleting an account invitation

To delete an account invitation before it has been accepted or expired:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Pending user invitations** area, find the email address associated with the original invitation.
4. From the **Options** menu, select **Delete invitation**.

Deleting account users

TIP

Deleting the owner of the account or the primary contact? Be sure to transfer ownership first or designate a new primary contact before deleting them.

To delete a user from an account, follow the steps below:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Active users** area, find the name or email of the user to delete.
4. Click the **Options** menu to the right of the user to be deleted, then select **Delete user** from the menu that appears. A confirmation window appears.
5. If the user has active API tokens associated with their account, click **Review this user's API tokens** to manually review and revoke them. Alternatively, select the checkbox to automatically revoke all of the user's API tokens and delete the user.

   WARNING

   Deleting an API token will break any integration actively using that credential. Verify you have changed the API token for your integrations before proceeding.

6. Click **Confirm and delete**.

TIP

Looking to close an account? Accounts can only be canceled by account owners.

* * *

Changing and resetting passwords

/en/guides/changing-and-resetting-passwords

All users can change or reset their own password. If you've been assigned the role of superuser, you can reset the password of any user currently associated with your company.

✓ TIP

Need to update a name or email address associated with your account or someone else's? Check out our guide on changing names and emails.

Before you begin

Keep in mind the following:

- Fastly follows guidelines from the National Institute of Standards and Technology when checking password strength and security (NIST SP 800-63B provides more details).
- Most password managers associate entries in their databases with specific website domains. If you've stored your username and password combination there for either your Fastly or Signal Sciences accounts, your password manager won't have any way of knowing when you've added or updated information. You must always manually enter and update the information in your password manager yourself.
- If your company sets up single sign-on for their Fastly accounts or Signal Sciences accounts, passwords for existing users will be deleted and active sessions will be expired. You'll be notified of password reset requirements associated with SSO enablement, if any, at that time.

Changing your password

Follow these instructions to change the password currently associated with your account:

1. Navigate to the appropriate login page for your account:
    - If you only have a Fastly account, use https://manage.fastly.com/ to log in.
    - If you only have a Signal Sciences account use https://dashboard.signalsciences.net/ to log in.
    - If you have both a Fastly account and a Signal Sciences account, use https://manage.fastly.com/ to log in.
2. Click **Change password**.
3. Fill out the page as follows:
    - In the **Current password** field, enter your existing password.
    - In the **New password** field, enter the new password.
    - In the **Confirm password** field, enter the new password a second time.
4. Click **Change Password** to save the changes.

Resetting your password

To reset your password (for example, when you forget it or believe it may have been compromised), follow these steps:

1. Navigate to the appropriate login page for your account:
    - If you only have a Fastly account, use https://manage.fastly.com/ to log in.
    - If you only have a Signal Sciences account use https://dashboard.signalsciences.net/ to log in.

  - If you have both a Fastly account and a Signal Sciences account, use https://manage.fastly.com/ to log in.

2. In the **Email** field, enter the email address associated with your account and click **Continue**.
3. Click **Forgot password** and then click **Continue**. Password reset instructions will be emailed to your email address, including a password reset link.
4. Check your email for the verification email. The subject line of the email will read `Fastly password reset requested`.
5. Click the password reset link in the email. The password reset screen appears.
6. In the **New password** field, enter a new password.
7. In the **Confirm new password** field, enter the new password a second time to confirm it.
8. Click **Continue** and then click **Log in** to access your account using the new password.

Resetting someone else's password

If you're a superuser, follow these instructions to reset the password currently associated with another user's account:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Active users** area, find the appropriate user and then click the **Options** menu next to their name.
4. From the Options menu, select **Send password reset**.
5. Verify the email address of the user, and then click **Confirm and send** to save the changes. Password reset instructions are emailed to the user.

* * *

Changing names and email addresses

/en/guides/changing-names-and-email-addresses

The Fastly web interface allows you to change the name associated with your Fastly account.

> TIP
>
> Need to update a password associated with your account or someone else's? Check out our guide on changing and resetting passwords.

Changing names

Anyone can change the name associated with their Fastly account. If you've been assigned the role of superuser, you can change the name of anyone currently associated with your company's Fastly account.

IMPORTANT

Looking to change your name on your Signal Sciences account? If you only have a Signal Sciences account or if you have both a Fastly account and a Signal Sciences account that share the same email address, you must contact support to change your name.

Changing your name

Follow these instructions to change the name currently associated with your Fastly account:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **Your profile**.
3. In the **Name** field, enter your name.
4. Click **Update Profile** to save the changes.

Changing someone else's name

Follow these instructions to change the name currently associated with someone's Fastly account:

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Users** area, find the appropriate user and then click the gear next to their name.
4. From the menu that appears, select **Edit user**.
5. In the **Name** field, enter the updated name.
6. Click **Update** to save the changes.

Changing email addresses

To change the email address associated with your account, contact support.

* * *

Configuring user roles and permissions

/en/guides/configuring-user-roles-and-permissions

Your Fastly account can be managed by multiple users through the role-based access controls in the web interface. These controls allow you to manage the scope of a user's service access and their specific permission levels for that service access, all based on the role assigned to them.

TIP

Need to configure roles for Signal Sciences accounts? Our guide to managing users of the Next-Gen WAF provides details.

User roles and what they can do

When invited to join a Fastly account, you'll be assigned a specific *role*. Think of roles as a way for your company to group the main business functions its users perform when invited to an account. Your role may afford you the ability to view and control a variety of things.

- **User** roles typically have some limited ability to view (but not manage) basic information about service configurations and controls, including those related to TLS management information. You'll also have the ability to view real-time and historical stats. You won’t have access to billing and payment information.

- **Billing** roles typically have full access to view (but not manage) basic information about service configurations, invoices, and account billing history. You'll also have the ability to manage payment information and account types and to view real-time and historical stats.

- **Engineer** roles typically have the ability to create services and manage their configurations. Some of these abilities may be restricted on a per service basis, however. When assigned this role, you'll also be able to invite new engineer and user roles via the API. You won't have access to billing and payment information.

- **Superuser** roles have full account access, with the ability to manage all aspects of service configurations and account settings, including full access to billing and payment information. When assigned this role, you cannot close or cancel an account unless you are also the account owner.

Abilities granted to user roles are selective, not additive. Regardless of your role, you'll have the ability to manage you personal profile, personal multi-factor authentication, and personal API tokens, view basic stats information, and submit help requests to Fastly Customer Support.

Service access and permission levels

The ability to do things on an account is governed by *permissions*. Each permission has a name associated with it that summarizes the type of actions you're allowed to do when that permission level is granted to you. By default, all roles grant access to every service on an account, including those services created in the future. The engineer role is unique, however, because superusers can limit an engineer's account access on a per-service basis and can assign permissions on each of those services separately as follows:

- **Read-only.** Allows an engineer to view a specific service's configuration but does not allow them to issue purge requests for that service nor make changes to its configuration.

- **Purge select.** Allows an engineer to view a specific service's configuration and also allows them to issue purge requests for that service via URL or surrogate key. They cannot use the purge all function on the service, nor can they make configuration changes to that service.

- **Purge all.** Allows an engineer to view a specific service's configuration and issue purge requests for the entire service via the purge all function. They cannot, however, make configuration changes to that service.

- **Full access.** Allows an engineer full access to a specific service, including permission to issue purge requests via any method on that service. They can make configuration changes to that service and can activate new versions of it at will.

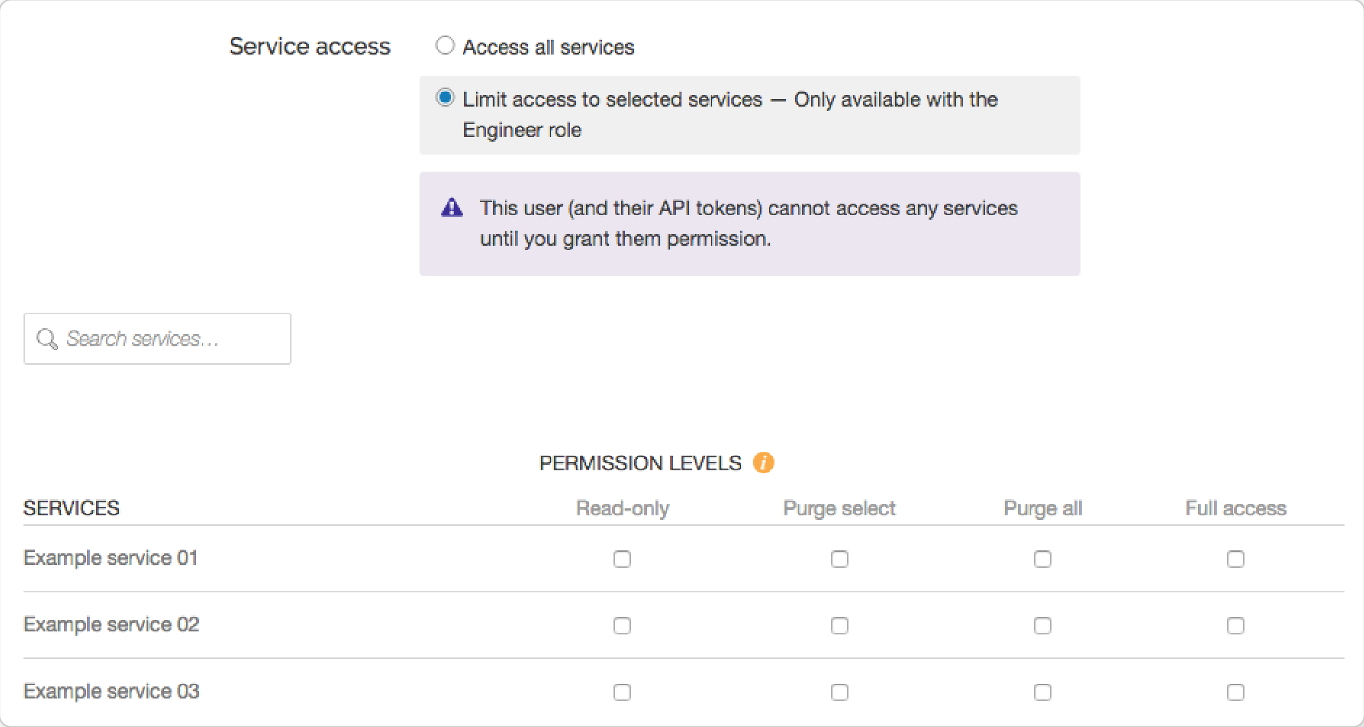
Permission levels are additive, not selective. Each level includes the previous level's permissions. When new services are added to an account by a superuser, engineers with anything but full access to services will not have access to those services until a superuser specifically grants a permission level manually.

Changing user roles and access permissions for existing users

Users assigned the superuser role can change the role, service access, or permission levels for any existing user on your account. Plan your changes carefully.

> WARNING
>
> Role, service access, and permission level changes for existing users apply instantly and get saved automatically.

1. Log in to the Fastly web interface.
2. Go to **Account** > **User management**.
3. In the **Active users** area, click the **Options** menu next to a user name and then select **Access controls**.
4. *(Optional)* From the **Choose their role** choices, select a new role for the user.
5. *(Optional)* Check the **TLS management** box to grant TLS configuration access to a user. Users with the role of superuser have this permission by default.
6. *(Optional)* From the **Service access** controls, select **Limit access to selected services** to limit access to selected services for users assigned the role of engineer.



Service access

○ Access all services

● Limit access to selected services — Only available with the Engineer role

This user (and their API tokens) cannot access any services until you grant them permission.

PERMISSION LEVELS

SERVICES	Read-only	Purge select	Purge all	Full access
Example service 01	☐	☐	☐	☐
Example service 02	☐	☐	☐	☐
Example service 03	☐	☐	☐	☐

7. *(Optional)* If you've limited access to selected services for a user assigned the role of engineer, select the specific permission levels for each service associated with the account.
8. Click **Update**. The user's role and permission levels will be changed accordingly.

> TIP
>
> Our guide to adding and deleting users provides instructions on how to add users to an account or delete those users from your account when you no longer want them to have access.

Account ownership

We assign the special role of *owner* to the first user who signs up for an account for your organization and we automatically assign that owner the superuser role. Any superuser on your account can change the permissions on an owner role or transfer ownership via the Company settings, which are accessible from the Account controls of the web interface.

Account owners typically serve as the primary point of contact for billing purposes. Invoices are sent to them, but if a specific billing contact has been defined for an account, invoices go to that contact instead. In addition, accounts can only be canceled by owners.

* * *

Managing multiple accounts

/en/guides/managing-multiple-accounts

Fastly's multi-account user access feature allows you to manage your access to multiple Fastly accounts. If you've been invited as a user to more than one Fastly account, you can quickly switch between those accounts without logging out of the Fastly web interface. The multi-account user access feature works with single sign-on (SSO) and two-factor authentication.

TIP

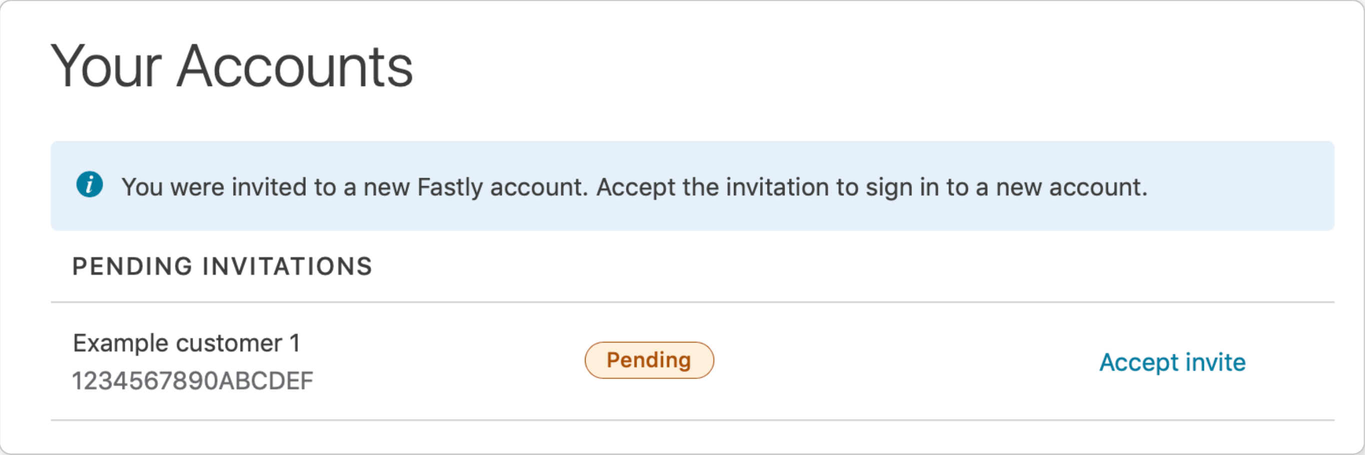
Looking to switch between Fastly and Next-Gen WAF control panels? Use our application switcher.

Accepting an invitation to join an account

If you've been invited to a customer account and you don't have a Fastly user account, follow the instructions sent via email to create a user account and accept the invitation.

If you're an existing user who already has a Fastly user account, follow the steps below:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **Your accounts**.



Your Accounts

You were invited to a new Fastly account. Accept the invitation to sign in to a new account.

PENDING INVITATIONS

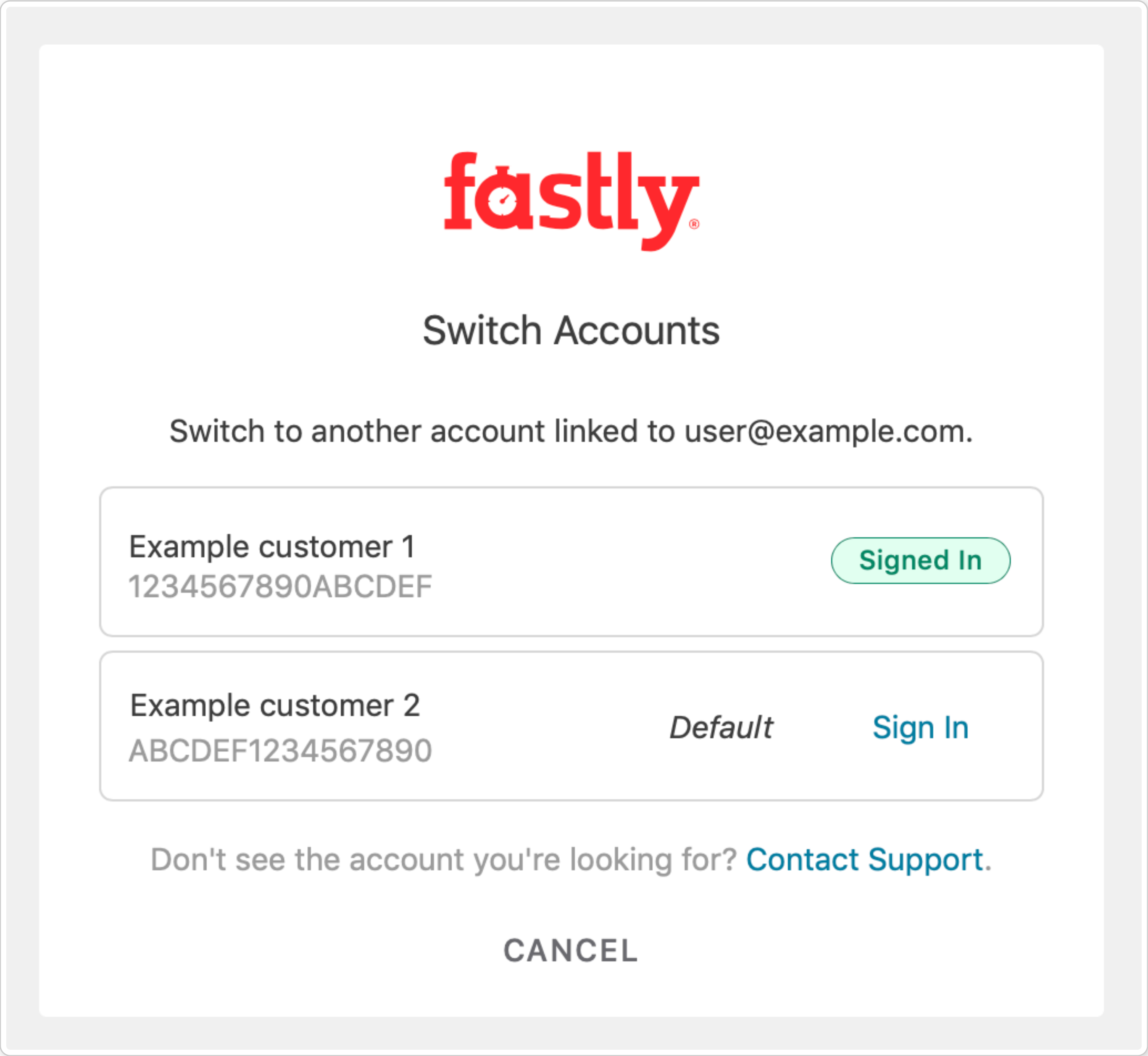
Example customer 1
1234567890ABCDEF

3. Find the invitation in the list and click the **Accept invite** link.

Switching accounts

Follow the steps below to switch between the customer accounts you can access:

1. Log in to the Fastly web interface and click **Switch Account** from the user quick links menu.



Switch Accounts

Switch to another account linked to user@example.com.

Example customer 1
1234567890ABCDEF

Example customer 2
ABCDEF1234567890
Default

Don't see the account you're looking for? Contact Support.

2. Click **Sign in** next to the service you want to log in to.
3. If prompted, enter your password.
4. If you have two-factor authentication enabled for your account, enter the time-based authentication code from your mobile device, then click **Sign in**.

Viewing the accounts you can access

Follow the steps below to view the customer accounts you can access:

1. Log in to the Fastly web interface.

2. Go to **Account** > **Personal profile** > **Your accounts**. The Your Accounts page appears.

Setting a default account

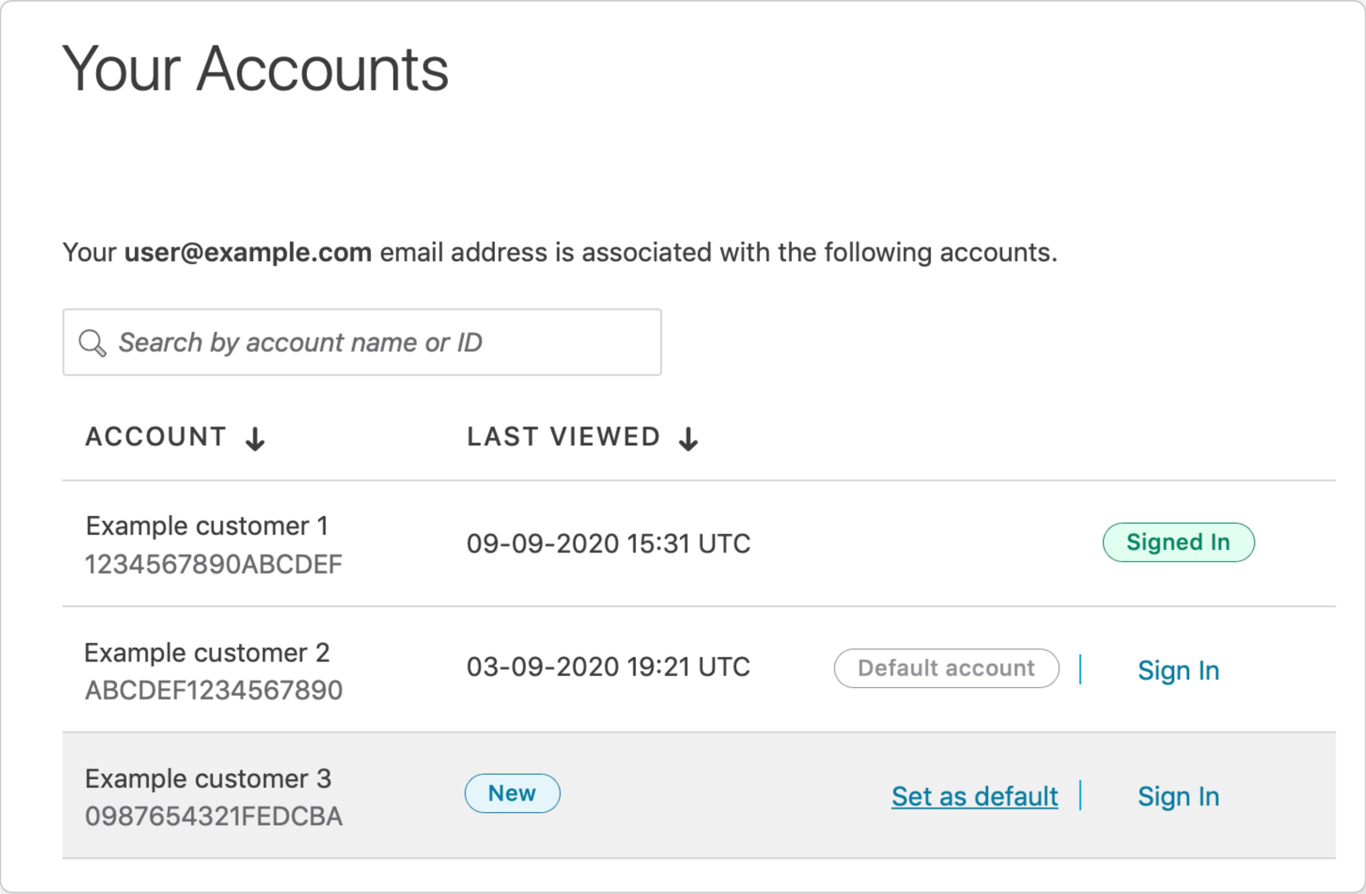
You can set a default customer account that appears automatically when you log in to the Fastly web interface.

IMPORTANT

Customer accounts configured to use single sign-on (SSO) can't be set as the default account.

Follow the steps below to set a default account:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Personal profile** > **Your accounts**.



Your Accounts

Your **user@example.com** email address is associated with the following accounts.

ACCOUNT ↓	LAST VIEWED ↓		
Example customer 1 1234567890ABCDEF	09-09-2020 15:31 UTC		Signed In
Example customer 2 ABCDEF1234567890	03-09-2020 19:21 UTC	Default account	Sign In
Example customer 3 0987654321FEDCBA	New	Set as default	Sign In

3. Hover your cursor over a service, then click **Set as default**.

When you log in with your email address and password, the customer account you set as the default will automatically appear.

* * *

Setting up single sign-on (SSO)

/en/guides/setting-up-single-sign-on-sso

If your company uses an identity provider (IdP) like Okta or OneLogin to manage user authentication, you can enable Fastly's single sign-on (SSO) feature to either allow or require your organization's users to sign in to the Fastly web interface using the IdP instead of an email address and password.

> ✓ TIP
>
> You can set up SSO for Signal Sciences accounts too. Check out our enablement guide for details.

Prerequisites

To enable SSO or require that it be applied to all of your organization's users when they log in to the Fastly web interface, you must:

- be assigned the role of superuser for your Fastly account
- have access to your IdP's administration console

In addition, your IdP must support:

- Security Assertion Markup Language 2.0 (SAML 2.0)
- Service Provider (SP) initiated SSO
- IdP-initiated SSO

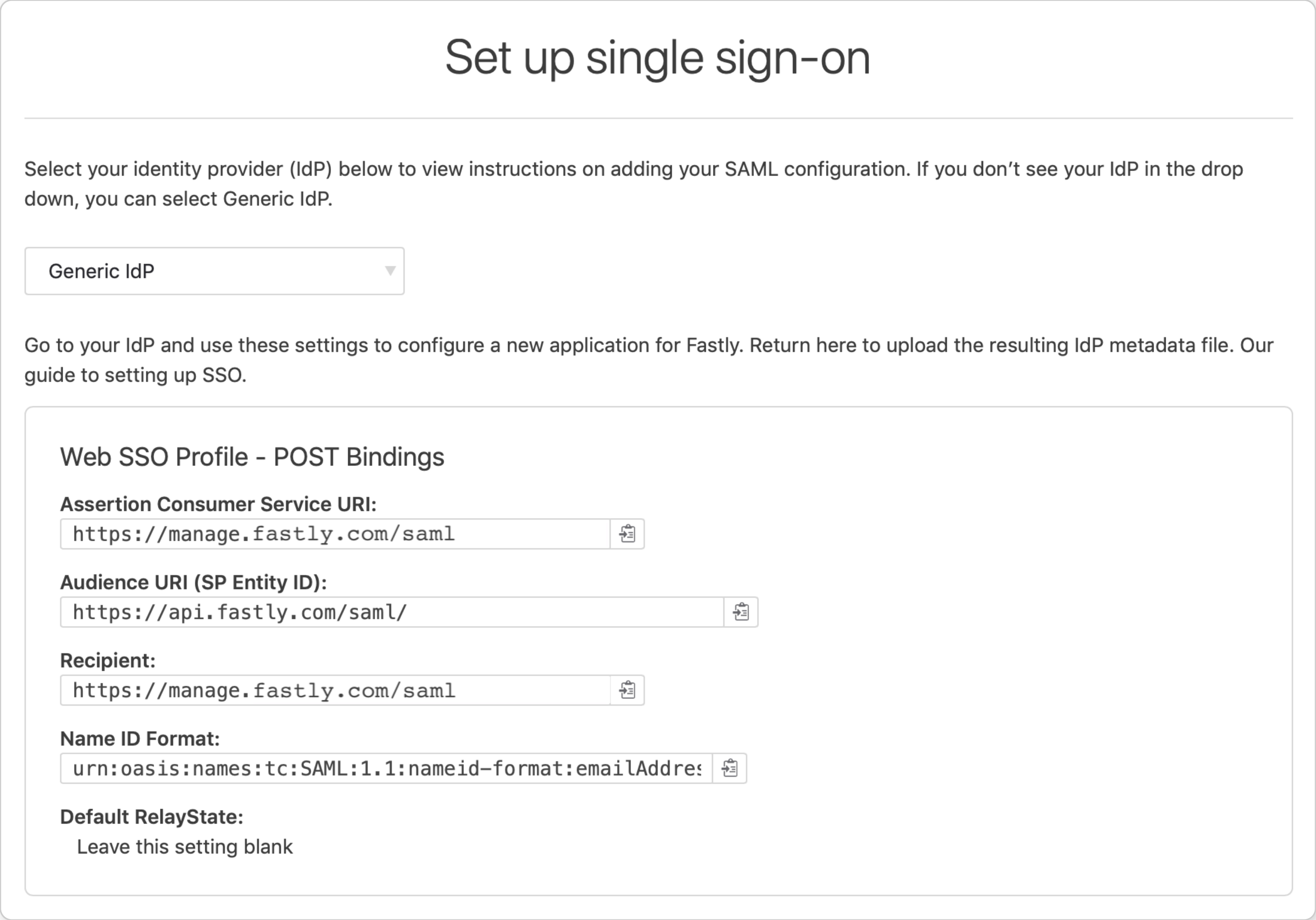
You should also review this feature's limitations before enabling SSO.

> IMPORTANT
>
> If you build a custom SAML application using Okta as your IdP, you must enable the configuration option "Honor Force Authentication" in Okta. If you are using a pre-built application in the Okta Application Network, the setting remains static. Read more about this requirement in Okta's configuration documentation.

Enabling SSO

Start by selecting an IdP and configure that provider's settings keeping in mind the prerequisites. You'll need to retrieve a metadata file containing your IdP's SAML configurations for use in the Fastly web interface:

1. Log in to the Fastly web interface.
2. Go to **Account** > **Single sign-on**.
3. Click **Add SAML Configuration**.
4. From the menu, select your organization's IdP.



Set up single sign-on

Select your identity provider (IdP) below to view instructions on adding your SAML configuration. If you don’t see your IdP in the drop down, you can select Generic IdP.

Generic IdP

Go to your IdP and use these settings to configure a new application for Fastly. Return here to upload the resulting IdP metadata file. Our guide to setting up SSO.

Web SSO Profile - POST Bindings

Assertion Consumer Service URI: `https://manage.fastly.com/saml`

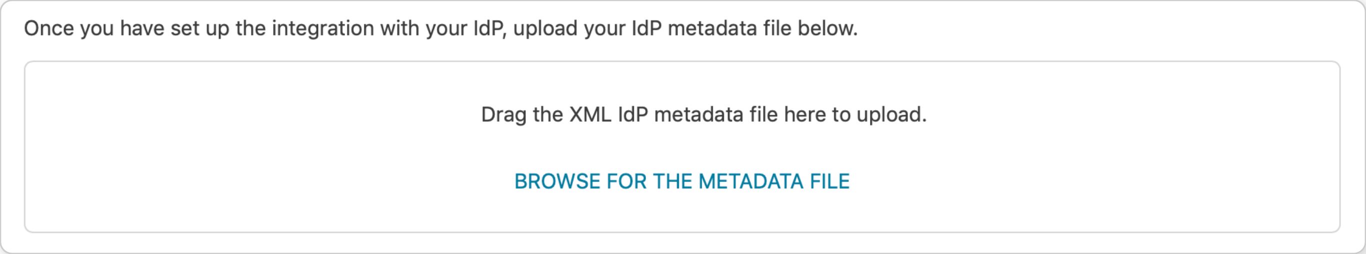
Audience URI (SP Entity ID): `https://api.fastly.com/saml/`

Recipient: `https://manage.fastly.com/saml`

Name ID Format: `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddres`

Default RelayState: Leave this setting blank

5. Using the configuration details that appear in the Fastly web interface, create a new SAML 2.0 application in your IdP's administration console and assign the application to new and existing users. Refer to your IdP's documentation for more information.
6. After creating the SAML 2.0 application in your IdP, download the XML metadata file with your application’s SAML configuration. The XML file includes a public certificate used to verify the signature of SAML assertions.
7. Upload your IdP metadata file. You can do this by dragging and dropping the file into the area provided or by browsing for the file and uploading it.



Once you have set up the integration with your IdP, upload your IdP metadata file below.

Drag the XML IdP metadata file here to upload.

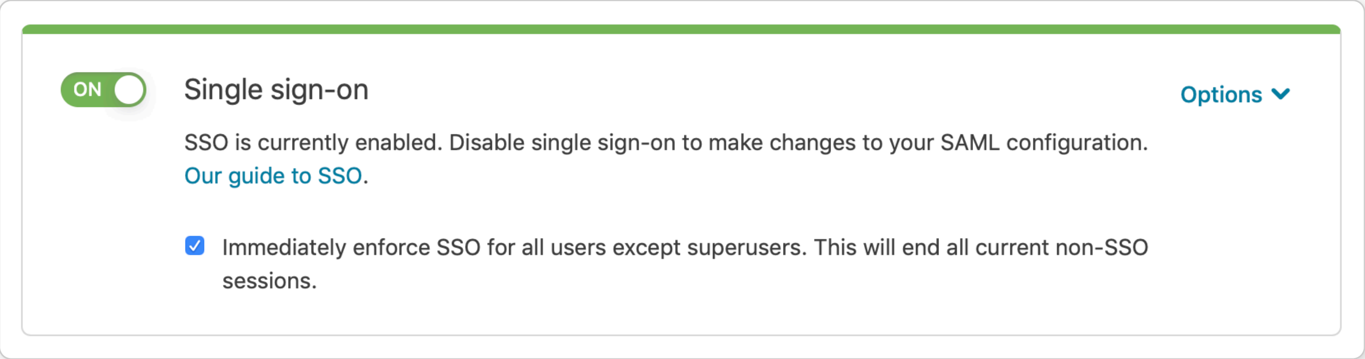
8. Click **Save and Enable SSO**. Your metadata will be saved and the SSO controls will indicate that SSO is enabled.

Requiring SSO for your organization

To require SSO for everyone in your organization except superusers, follow these instructions.

1. Log in to the Fastly web interface.

2. Go to **Account** > **Single sign-on**.



ON Single sign-on

SSO is currently enabled. Disable single sign-on to make changes to your SAML configuration. Our guide to SSO.

☑ Immediately enforce SSO for all users except superusers. This will end all current non-SSO sessions.

3. Select the **Immediately enforce SSO** checkbox. Currently open non-SSO sessions for users assigned the role of user, billing, or engineer will be logged out and they will need to re-authenticate using SSO via your IdP.

> NOTE
>
> Users who have been assigned the role of superuser can always log in with their email address and password, whether or not **Single sign-on** is enabled.

Performing account tasks differently with SSO enabled

If your organization has enabled SSO, you may notice different feature availability in the Fastly web interface. This section describes the differences.

- **Changing your email address and password.** Because SSO requires user email addresses in Fastly to match those in the IdP, you won't be able to change your email address while logged in using SSO. You also won't be able to modify your password or enable two-factor authentication.
- **Creating an API token.** To create an API token while logged in to the Fastly web interface using SSO, you'll need to reauthenticate with your IdP. Follow the instructions for creating an API token and click the **Re-Authenticate** button on the Create a Token page.

  > NOTE
  >
  > You can't create API tokens when using G Suite for authentication.

- **Managing sessions.** Sessions created by logging in to the Fastly web interface using SSO or with a username and password expire after 3 hours.

Changing SSO providers

To change SSO providers, follow these instructions.

> WARNING
>
> Disabling the SSO feature for your organization will expire all active SSO sessions, including your own. Users will automatically be logged out of the Fastly web interface.

1. Log in to the Fastly web interface.
2. Go to **Account** > **Single sign-on**.

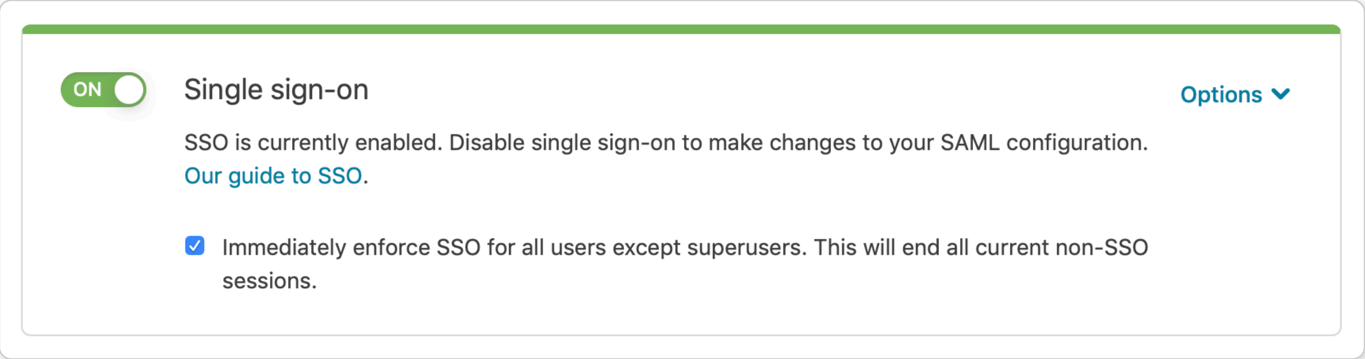
3. From the **Options** menu, select **Change your SAML configuration**.
4. In the confirmation window, click **Confirm, Disable SSO and Delete**.
5. Follow the instructions in the enabling SSO section.

Disabling SSO

To disable SSO for your organization, either permanently or temporarily (e.g., your SSO provider is experiencing an outage), follow these instructions. Disabling SSO won't delete your SSO settings and you can re-enable SSO at any point using the same IdP configuration metadata you uploaded when you first enabled SSO.

> WARNING
>
> Disabling the SSO feature for your organization, even temporarily, will expire all active SSO sessions, including your own, and will automatically log users out of the Fastly web interface.

1. Log in to the Fastly web interface.
2. Go to **Account** > **Single sign-on**.



SSO is currently enabled. Disable single sign-on to make changes to your SAML configuration. Our guide to SSO.

☑ Immediately enforce SSO for all users except superusers. This will end all current non-SSO sessions.

3. Click the **Single sign-on** switch to disable SSO for your organization.
4. In the confirmation window, click **Disable SSO**. SSO will be disabled and will not be required for your organization's users. All active SSO sessions will expire, including your own, and users will automatically be logged out of the Fastly web interface.

Limitations

Fastly's SSO feature has the following limitations:

- Users cannot create API tokens from the Fastly web interface when using G Suite SSO for a session's authentication.
- Fastly does not support automatic provisioning and de-provisioning of users using SCIM for Fastly accounts. To enable SCIM for your Signal Sciences account, check out our guide to automating user management.

* * *

Transferring services to other accounts

🔗 /en/guides/transferring-services-to-other-accounts

If several employees at your company independently create testing accounts when learning about Fastly products and services, you can have the services that were created on the testing accounts transferred to another account by contacting support with the following information:

- the Customer IDs of the accounts
- the names of the services to be transferred
- which account should be considered the primary account

After you contact us, we'll reach out to verify the ownership of each account. If we can confirm ownership, we'll initiate the transfer.

> NOTE
>
> Account API tokens can't be transferred and don't work across multiple accounts. If you use account API tokens for purging or service automation on the service that you are transferring, an engineer or superuser will need to recreate those API tokens in the account that is receiving the service.

> TIP
>
> Fastly's multi account user access feature allows a single user to manage multiple customer accounts. If you've been invited to more than one customer account, you can quickly switch between accounts without logging out of the Fastly web interface.

* * *

Unsubscribing from Fastly marketing email

🔗 /en/guides/unsubscribing-from-fastly-marketing-email

If you receive what appears to be a legitimate marketing communication or promotion from Fastly, you may opt-out of these emails at any time by clicking the unsubscribe link provided in the email or by forwarding it to abuse@fastly.com so that we can address it. You will still receive emails from support@fastly.com.

Dealing with other unsolicited emails

You may receive unsolicited email messages that are neither sent by Fastly nor associated with Fastly, even if you've never created a Fastly account. These unwanted emails use email spoofing to give the false impression that they were sent by Fastly. They may contain links to Fastly websites to give them the appearance of legitimacy.

If an email was not sent by Fastly's mail servers, there is nothing we can do to stop additional spam emails from being sent to your address. To block these messages entirely, you will need to use the spam protections made available by your email provider.

* * *

Compute guides

Category: Developer guides

These articles describe how to develop with Compute.

Frameworks

/en/guides/frameworks

---

* * *

Getting started with Compute

/en/guides/getting-started-with-compute

---

* * *

Migrate from VCL

/en/guides/migrate-from-vcl

---

* * *

Testing and debugging

/en/guides/testing-and-debugging

---

* * *

Using Go

🔗 /en/guides/using-go

---

* * *

Using JavaScript

🔗 /en/guides/using-javascript

---

* * *

Using other languages

🔗 /en/guides/using-other-languages

---

* * *

Using Rust

🔗 /en/guides/using-rust

---

* * *

Category: Edge Data Storage

These articles describe how to work with the edge data storage features available with Compute.

Working with config stores

🔗 /en/guides/working-with-config-stores

Config stores are a type of versionless container that allow you to store often repeated data as key-value pairs that can be read from the edge and shared by multiple Compute services in your account.

You can also create and work with config stores via the API.

Prerequisites

Config Store is only available for Fastly's Compute services, not for CDN (VCL-based) services.

Limitations and considerations

When creating or making changes to config stores, keep the following things in mind:

- Config stores can only be used by Compute services, not CDN (VCL-based) services.
- Trials for Compute include one config store with a maximum of 100 entries.
- Paid accounts include a minimum of five config stores (each having a maximum of 500 entries).
- Config store keys are limited to 255 characters and their values are limited to 8,000 characters.
- Config stores and the names of keys and their values are case-sensitive.

⚠ WARNING

Personal information, secrets, or sensitive data should not be included in config stores or incorporated into edge logic. In addition, we do not maintain version histories of your config stores. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

Creating a config store

Creating a config store requires you to create at least one key-value pair before you associate it with a service. To create a new config store and its key-value pairs, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Click **Create a config store**.
4. In the **Name of config store** field, enter a name for the config store and then click **Add**.
5. Click the **Key-value pairs** link.
6. Click **Add item**.
7. Enter the key and the value in the appropriate column and then click **Add**.
8. Continue adding additional key-value pairs as necessary.

Linking config stores to a service

Once you've added at least one key-value pair to a config store, you can link it to a service from the Resources controls or from the service configuration for the service.

Using the Resources controls to link config stores to a service

To link a config store to a service from the Resources controls:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Click **Link to services** to the right of the store you want to link.
4. Select the checkbox next to any services you want to link your config store to.
5. Click **Next**.
6. Decide which version of the service to link to. By default, the system will assume you want to clone the most recently active version of your service. You can choose an existing draft version of the service instead by selecting it specifically from the **Version** menu.
7. Select one of the following options for linking the store to your service:
    - **Link only:** links the store to the selected service versions but leaves any cloned or draft versions un-activated so you can activate them at a later time.
    - **Link and activate:** links the store to the selected service versions and activates those versions at the same time.

   A success message appears once the config store is linked to the service.
8. Finally, do one of the following:
    - Click **Activate versions** to activate any cloned or draft versions of services linked to the config store.
    - Click **Finish** to leave the cloned or draft service versions un-activated so you can make additional configuration changes to them and activate them at a later time.

   You can immediately start referencing the config store in your edge logic.

Using the service configuration to link config stores to a service

To link a config store to a service from the service configuration:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.
4. From the **Resources** options in the on-page navigation, click **Config stores**.
5. From the **Link Config Store to service** menu, select the config store you want to link to the service. A success message appears once the config store is linked to the service.

Once linked, you can immediately start referencing the config store in your edge logic and activate the service when you’re ready.

Unlinking config stores

You can unlink a config store from a service from the service configuration.

To unlink a config store:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.
4. From the **Resources** options in the on-page navigation, click **Config stores**.
5. Click **Unlink from service** next to the config store you want to unlink from your service.
6. Click **Confirm and unlink**. A new, draft version of the service is created.
7. Activate the service to finalize unlinking the config store.

Editing a config store

You can edit the name of a config store as well as the key-value pairs within the store.

To rename a config store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Click the pencil to the right of the store you want to rename.
4. Enter a new name for the config store.
5. Click **Save** and then **Confirm and rename** to continue.

> ✓ TIP
>
> Be sure to update any custom logic with the new name of the object store.

To add new key-value pairs to a config store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Expand the **Key-value pairs** section.
4. Under **Key-value** pairs, click **Add item**.
5. Enter the key and the value in the appropriate column and then click **Add**.
6. Repeat for any additional key-value pairs.

To edit the key-value pairs within a config store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Expand the **Key-value pairs** section.

4. Hover your cursor over the entry you want to edit, then click **Edit**.
5. Edit the key or value as necessary.
6. Click **Save**.

The changes you make will be immediately applied to your configuration including any deployed service versions associated with the config store.

Deleting a config store

You can delete a config store at any time. A config store must be unlinked from any services or an error will appear if you try to delete it.

Deleting a config store also deletes all key-value pairs within the store. If you have custom logic referencing these keys, ensure you update them before deleting the store.

To delete a config store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Config stores**.
3. Click the trash to the right of the store you want to delete.
4. Click **Confirm and delete**.

* * *

Working with KV stores

🔗 /en/guides/working-with-kv-stores

A KV store is a type of container that allows you to store data in the form of key-value pairs for use in high performance reads and writes at the edge. A single KV store can be associated with multiple Compute services in your account. Because KV stores are versionless, the data in them never expires and can be updated at any time after a KV store is created, without requiring you to increment a service's version. Also, because KV stores are stored at the edge, they allow you to offload that data from your origin and keep it closer to your end users.

You can also create and work with KV stores via the API.

Prerequisites

KV Store is only available for Fastly's Compute services, not for CDN (VCL-based) services.

Limitations and considerations

When creating or making changes to KV stores, keep the following things in mind:

- KV stores can only be used by Compute services, not CDN (VCL-based) services.

- The number of KV stores you can create is limited to your account's available resources allotment.
- KV stores support a maximum key size of 1024 bytes in UTF-8 format.
- KV stores support values submitted using any file type and have a maximum size of 25MB. This maximum can be raised to 100MB upon request. Contact your account manager or sales@fastly.com for more information.

⚠ WARNING

Personal information, secrets, or sensitive data should not be included in KV stores or incorporated into edge logic. In addition, we do not maintain version histories of your KV stores. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

Creating a KV store

Creating a KV store requires you to create at least one key-value pair before you associate it with a service. To create a new KV store and its key-value pairs, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Click **Create a KV store**.
4. In the **Name of KV store** field, enter a name for the KV store and then click **Add**.
5. Click **Key-value pairs**.
6. In the **Name of key** field, enter a name for the first set of key-value pairs associated with your KV store.
7. Upload the values for your key, clicking **Browse for file** to navigate to the file on your system using the file picker. Alternatively, drag and drop your key file directly into the drag and drop area below these controls.
8. Click **Add** to the right of the key controls.
9. Click **Add item** to continue adding additional key-value pairs as necessary.

✓ TIP

By default, the web interface displays only the first 1,000 keys in the store. To view more keys, click the **Load more keys** button.

Linking KV stores to a service

Once you've added at least one key-value pair to a KV store, you can link it to a service from the Resources controls or from the service configuration for the service.

Using the Resources controls to link KV stores to a service

To link a KV store to a service from the Resources controls:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Click **Link to services** to the right of the store you want to link.

4. Select the checkbox next to any services you want to link your KV store to and then click **Next**.
5. Decide which version of the service to link to. By default, the system will assume you want to clone the most recently active version of your service. You can choose an existing draft version of the service instead by selecting it specifically from the **Version** menu.
6. Select one of the following options for linking the store to your service:
   - **Link only:** links the store to the selected service versions but leaves any cloned or draft versions un-activated so you can activate them at a later time.
   - **Link and activate:** links the store to the selected service versions and activates those versions at the same time.

   A success message appears once the KV store is linked to the service.
7. Finally, do one of the following:
   - Click **Activate versions** to activate any cloned or draft versions of services linked to the KV store.
   - Click **Finish** to leave the cloned or draft service versions un-activated so you can make additional configuration changes to them and activate them at a later time.

   You can immediately start referencing the KV store in your edge logic.

Using the service configuration to link KV stores to a service

To link a KV store to a service from the service configuration:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.
4. From the **Resources** options in the on-page navigation, click **KV stores**.
5. From the **Link KV Store to service** menu, select the KV store you want to link to the service. A success message appears indicating the store is linked to your service.

Once linked, you can immediately start referencing the KV store in your edge logic and activate the service when you're ready.

Unlinking KV stores

You can unlink a KV store from a service from the service configuration.

To unlink a KV store:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.

4. From the **Resources** options in the on-page navigation, click **KV stores**.
5. Click **Unlink from service** next to the KV store you want to unlink from your service.
6. Click **Confirm and unlink**. A new, draft version of the service is created.
7. Activate the service to finalize unlinking the KV store.

Editing a KV store

You can edit the name of a KV store as well as the key-value pairs within the store.

To rename a KV store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Click the pencil to the right of the store you want to rename.
4. Enter a new name for the KV store.
5. Click **Save** and then **Confirm and rename** to continue.

> ✓ TIP
>
> Be sure to update any custom logic with the new name of the KV store.

To add new key-value pairs to an object store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Expand the **Key-value pairs** section.
4. Under **Key-value** pairs, click **Add item**.
5. Enter a key name and upload a key file that has the values and then click **Add**.
6. Repeat for any additional key-value pairs.

To edit the key-value pairs within a KV store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Expand the **Key-value pairs** section.
4. Hover your cursor over the entry you want to edit, then click **Edit** .
5. Drag and drop your key file that has the values into the drag and drop area to upload your list of keys. Alternatively, click **Browse for file** to navigate to the file on your system using the file picker.
6. Click **Save**.

The changes you make will be immediately applied to your configuration including any deployed service versions associated with the KV store.

Deleting a KV store

You can delete a KV store at any time.

Before deleting a KV store:

- Unlink the KV store from your services. If the KV store is linked to any service, an error will appear when you try to delete the store.
- Update any custom logic that references the key-value pairs in the KV store. Deleting a KV store also deletes all key-value pairs within the store.

To delete a KV store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **KV stores**.
3. Click trash to the right of the store you want to delete.
4. Click **Confirm and delete**.

* * *

Working with secret stores

/en/guides/working-with-secret-stores

Secret stores give you a secure location to place credentials so they are available to Compute services operating at the Fastly edge.

You can also create and work with secret stores via the API.

Prerequisites

Secret Store is only available for Fastly's Compute services, not for CDN (VCL-based) services.

Limitations and considerations

When creating or making changes to secret stores, keep the following things in mind:

- Trials for Compute include one secret store.
- Paid accounts include a minimum of 10 secrets regardless of the number of stores with additional secrets available for purchase.
- Secret store names can only contain letters, numbers, dashes, underscores, and periods.
- Secret stores support a maximum size of 64KB per secret.
- Secrets are limited to 5 secret reads per Compute request. To increase this limit contact Fastly Support.

- Secrets are limited to 1000 writes per month.

Creating a secret store

Creating a secret store requires you to create at least one key-value pair containing secrets and then associating the store with a service. To create a new secret store and add secrets, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Secret stores**.
3. Click **Create a secret store**.
4. Enter a name for the secret store and then click **Add**. The secret store you just created appears.
5. Click **Key-value pairs**.
6. Click **Add item**. A row appears where you can add secrets.
7. In the **Name** field, enter a name for the secret.
8. In the **Secret** field, enter the secret value. Alternatively, click **Browse for file** to upload a secrets file on your system using the file picker.
9. Click **Add**.
10. Click **Add item** to continue adding additional secrets as necessary.

Linking secret stores to a service

Once you've added at least one secret to a secret store, you can link it to a service from the Resources controls or from the service configuration for the service.

Using the Resources controls to link secret stores to a service

To link a secret store to a service from the **Resources** controls:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Secret stores**.
3. Click **Link to services** to the right of the store you want to link.
4. Select the checkbox next to any services you want to link your secret store to and then click **Next**.
5. Decide which version of the service to link to. By default, the system will assume you want to clone the most recently active version of your service. You can choose an existing draft version of the service instead by selecting it specifically from the **Version** menu.
6. Select one of the following options for linking the store to your service:
    - **Link only:** links the store to the selected service versions but leaves any cloned or draft versions un-activated so you can activate them at a later time.
    - **Link and activate:** links the store to the selected service versions and activates those versions at the same time.

   A success message appears once the secret store is linked to the service.

7. Finally, do one of the following:
    - Click **Activate versions** to activate any cloned or draft versions of services linked to the secret store.
    - Click **Finish** to leave the cloned or draft service versions un-activated so you can make additional configuration changes to them and activate them at a later time.

Using the service configuration to link secret stores to a service

To link a secret store to a service from the service configuration:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.
4. From the Resources options in the on-page navigation, click **Secret stores**.
5. From the **Link Secret Store to service** menu, select the secret store you want to link to the service. A success message appears indicating the store is linked to your service.

Unlinking secret stores

You can unlink a secret store from a service from the service configuration.

To unlink a secret store:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Service configuration**.
4. From the **Resources** options in the on-page navigation, click **Secret stores**.
5. Click **Unlink from service** next to the secret store you want to unlink from your service.
6. Click **Confirm and unlink**. A new, draft version of the service is created.
7. Activate the service to finalize unlinking the secret store.

Editing a secret store

After creating a secret store, you can edit the secrets within the story or add new secrets to the store.

To edit secrets within a store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Secret stores**.
3. Expand the **Key-value pairs** section.

4. Hover your cursor over the entry you want to edit, then click the **Edit** link that appears.
5. Update the secret value. Alternatively, click **Browse for file** to navigate to a file on your system using the file picker.
6. Click **Save**.

To add new secrets to a secret store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Secret stores**.
3. Expand the **Key-value pairs** section.
4. Under **Key-value** pairs, click **Add item**.
5. Enter the name and secret in the appropriate columns and then click **Add**.
6. Repeat for any additional secrets.

The changes you make will be immediately applied to your configuration including any deployed service versions associated with the secret store.

Deleting a secret store

You can delete a secret store at any time. Before deleting a secret store:

- Unlink the secret store from your services. If the secret store is linked to any service, an error will appear when you try to delete the store.
- Update any custom logic that references the key-value pairs in the secret store. Deleting a secret store also deletes all key-value pairs within the store.

To delete a secret store:

1. Log in to the Fastly web interface.
2. Go to **Resources** > **Secret stores**.
3. Click the trash to the right of the store you want to delete.
4. Click **Confirm and delete**.

* * *

Full Site Delivery guides

Category: Caching

These articles describe configuration settings and changes you can make to your cache settings when setting up Fastly services.

Caching configuration best practices

/en/guides/caching-best-practices

To ensure optimum origin performance during times of increased demand or during scheduled downtime for your servers, consider the following best practices for your service's caching configurations.

Integrate Fastly with your application platform

You can optimize caching with Fastly by customizing your application platform settings. For instructions, see our documentation on integrating third-party services and configuring web server software. We also provide a variety of plugins to help you directly integrate Fastly with your content management system.

Check your cache hit ratio

The number of requests delivered by a cache server, divided by the number of cacheable requests (hits + misses), is called the *cache hit ratio*. A high cache hit ratio means you've kept request traffic from hitting your origin unnecessarily. Requests come from cache instead. In general, you want your cache hit ratio as high as possible, usually in excess of 90%. You can check your hit ratio by viewing the Observability page for your service.

Set a fallback TTL

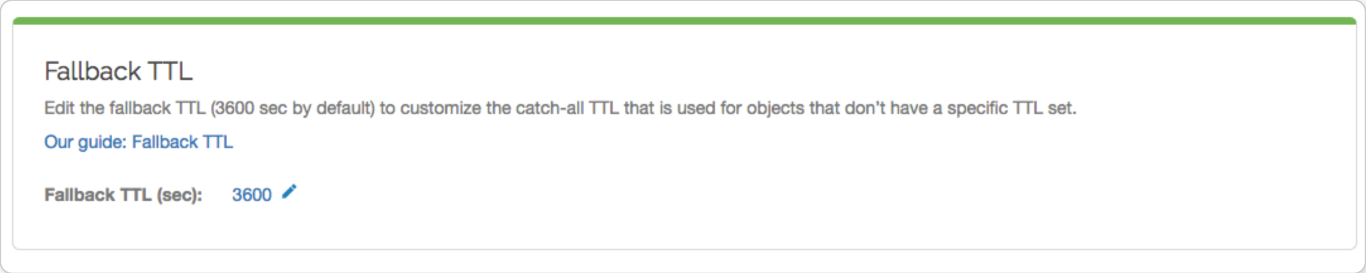
The amount of time information can be retained in cache memory is considered its *time to live* or TTL. TTL is set based on the cache related headers information returned from your origin server. You can specifically set a fallback TTL (sometimes called a *default TTL*).

> ⚠ WARNING
>
> If you're using custom VCL, the fallback TTL will be specified in the VCL boilerplate and the fallback TTL configured via the web interface or the API will not be applied. Refer to our documentation on cache freshness and TTLs for more information.

> ✓ TIP
>
> If there's no other source of freshness in the response, setting the fallback TTL to 0 seconds in the web interface will set `return(pass)` in `vcl_fetch`.

We set a default fallback TTL that you can update at any time as follows:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. In the **Fallback TTL** area, click the pencil next to the TTL setting.



Fallback TTL

Edit the fallback TTL (3600 sec by default) to customize the catch-all TTL that is used for objects that don't have a specific TTL set.

Our guide: Fallback TTL

Fallback TTL (sec): 3600

6. In the **Fallback TTL (sec)** field, enter the new TTL in seconds.
7. Click **Save** to save your changes.
8. Click **Activate** to deploy your configuration changes.

> NOTE
>
> See our Google Cloud Storage instructions if you're changing the default TTL for a GCS bucket.

Understand how cache control headers work

You can use cache control headers to set policies that determine how long your data is cached.

Fastly looks for caching information in each of these headers as described in our documentation on cache freshness. In order of preference:

- `Surrogate-Control:`
- `Cache-Control: s-maxage`
- `Cache-Control: max-age`
- `Expires:`

`Surrogate` headers

`Surrogate` headers are a relatively new addition to the cache management vocabulary (described in this W3C tech note). These headers provide a specific cache policy for proxy caches in the processing path. `Surrogate-Control` accepts many of the same values as `Cache-Control`, plus some other more esoteric ones (read the tech note for all the options).

One use of this technique is to provide conservative cache interactions to the browser (for example, `Cache-Control: no-cache`). This causes the browser to re-validate with the source on every request for the content. This makes sure that the user is getting the freshest possible content. Simultaneously, a `Surrogate-Control` header can be sent with a longer `max-age` that lets a proxy cache in front of the source handle most of the browser traffic, only passing requests to the source when the proxy's cache expires.

With Fastly, one of the most useful `Surrogate` headers is `Surrogate-Key`. When Fastly processes a request and sees a `Surrogate-Key` header, it uses the space-separated value as a list of tags to associate with the request URL in the cache. Combined with Fastly's Purge API an entire collection of URLs can be expired from the cache in one API call (and typically happens in around 1ms). `Surrogate-Control` is the most specific.

`Cache-Control` headers

`Cache-Control` headers, as defined by section 5.2 of RFC 7234, include:

- `Cache-Control: public` - Any cache can store a copy of the content.
- `Cache-Control: private` - Don't store, this is for a single user.
- `Cache-Control: no-cache` - Re-validate before serving this content.
- `Cache-Control: private, no-store` - Don't ever cache this content in Fastly or a web browser.
- `Cache-Control: public, max-age=[seconds]` - Caches can store this content for *n* seconds.
- `Cache-Control: s-maxage=[seconds]` - Same as max-age but applies specifically to proxy caches.

Only the `max-age`, `s-maxage`, and `private` Cache-Control headers will influence Fastly's caching. All other Cache-Control headers will not, but will be passed through to the browser. For more in-depth information about how Fastly responds to these Cache-Control headers and how these headers interact with Expires and Surrogate-Control, check out our documentation on cache freshness.

> NOTE
>
> For more information on the rest of the Cache-Control headers, see the relevant section in Mark Nottingham's caching tutorial.

`Expires` header

The `Expires` header tells the cache (typically a browser cache) how long to hang onto a piece of content. Thereafter, the browser will re-request the content from its source. The downside is that it's a static date and if you don't update it later, the date will pass and the browser will start requesting that resource from the source every time it sees it.

Fastly will respect the `Expires` header value only if the `Surrogate-Control` or `Cache-Control` headers are not found in the request.

Increase Cache-Control header times

During times of increased demand, you can instruct Fastly to keep objects in cache as long as possible by increasing the times you set on your Cache-Control headers. Consider changing the `max-age` on your `Cache-Control` or `Surrogate-Control` headers. Our documentation on cache freshness describes this in more detail.

Configure Fastly to temporarily serve stale content

If your origin becomes unavailable for an extended period of time (for example, being taken offline for maintenance purposes), temporarily serving stale content may help you. Serving stale content can also benefit you if your site's static content is updated or published quite frequently.

You can instruct Fastly to serve stale content by adding a `stale-while-revalidate` or `stale-if-error` statement on your `Cache-Control` or `Surrogate-Control` headers. Our guide to serving stale content describes this in more detail.

Decrease your first byte timeout time

After you have configured Fastly to temporarily serve stale, decreasing your first byte timeout time will cause stale content to be served to the requestor faster while fetching fresh content from the origin. Decreasing your first byte timeout time as well as serving stale will reduce unnecessary 503 first byte timeout errors. Decrease the first byte timeout time to your origin as follows:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. In the **Hosts** area, find your origin server and click the pencil to edit the host.
6. Click **Advanced options** at the bottom of the page.
7. In the **First byte timeout** field, enter the new first byte timeout in milliseconds. Approximately 15000 milliseconds is a good default to start with.
8. Click **Update** to save your changes.
9. Click **Activate** to deploy your configuration changes.

Configure caching actions for specific workflows

When you create a cache setting, the **Action** setting determines how the request will be handled. The action settings and the most common workflow for each are described below.

- **Do nothing now** - Selecting this option will apply the request setting options, but won't force a lookup or a pass action.
- **Pass (do not cache)** - Selecting this option will make the request and subsequent response bypass the cache and go straight to origin. Use this option if you need to conditionally prevent pages from caching or when using conditions
- **Restart processing (not common)** - Selecting this option will restart processing of the request. Use this option if you need to check multiple backends for a single request
- **Deliver (not common)** - Selecting this option will deliver the object to the client. Use this option if you need to create an override condition.

Consider custom error handling

When downtime can't be avoided, standard error messages might not ensure the best user experience. Consider creating custom error messages that include information specific to the request being made and pertinent to the user. Our guide to creating error pages with custom responses provides more detail.

HTTP status codes cached by default

Fastly caches the following response status codes by default. In addition to these statuses, you can force an object to cache under other states using conditions and responses.

Code	Message
`200`	OK
`203`	Non-Authoritative Information

Code	Message
`300`	Multiple Choices
`301`	Moved Permanently
`302`	Moved Temporarily
`404`	Not Found
`410`	Gone

✓ TIP

You can override caching defaults based on a backend response. For example, if you don't want 404 error responses to be cached for the full caching period of a day, you could add a cache object and then create conditions for it.

To cache status codes other than the ones listed above, set `beresp.cacheable = true;` in `vcl_fetch`. This tells Varnish to obey backend HTTP caching headers and any other custom `ttl` logic. A common pattern is to allow all 2XX responses to be cacheable:

```
sub vcl_fetch {
  # ...
  if (beresp.status >= 200 && beresp.status < 300) {
    set beresp.cacheable = true;
  }
  # ...
}
```

Inform Fastly Customer Support

We like to be sure we're readily available for assistance during customer events. When you know in advance that an event is forthcoming, contact support with details. Be sure to include details about:

- the date and time of the event
- the type of event happening
- how long you expect it to last (if it's planned)
- the Fastly services that might be affected

If the event you're planning is designed to validate the security of your service behind Fastly, be sure to read our guide to penetration testing first.

---

* * *

Checking cache

🔗 /en/guides/checking-cache

Checking the cache status of an object on your website can help when troubleshooting problems. You can use the web interface or the curl command (an open-source command line tool for transferring data with URL syntax from or to a server using one of many supported protocols) to check Fastly's cache nodes for a cached object, and you can use the information provided to examine the object's status, response time, and content hash.

Using curl

You can use one of two curl commands to tell if your request is caching in the Fastly network:

- a simple curl command that displays the request and response headers for a given object
- a slightly more complex curl command that uses the `Fastly-Debug` header to expose information normally stripped by the simple curl command

Using the simple curl command

The following curl command displays the request and response headers for a given object:

```
$ curl -svo /dev/null www.example.com/index.html
```

where `www.example.com/index.html` is replaced with the full object path of the object you're testing.

For example, using `curl -svo /dev/null www.example.com` produces something like the following section of output:

```
[...]

< Age: 142
< X-Served-By: cache-jfk1041-JFK, cache-ord1720-ORD
< X-Cache: HIT, HIT
< X-Cache-Hits: 1, 7

[...]
```

This output tells us the current age of the object in cache. It also shows shielding is enabled because two cache nodes display in `X-Served-By`. However, we're most interested in the output of the `X-Cache` header. A properly caching object displays a value of `X-Cache: HIT`, `X-Cache: HIT, HIT`, `X-Cache: HIT, MISS`, or `X-Cache: MISS, HIT`.

Using a Fastly-Debug header with curl

The `Fastly-Debug` header provides additional information for debugging by exposing specific information that is normally stripped when using a simple curl command:

```
$ curl -svo /dev/null -H "Fastly-Debug:1" www.example.com/index.html
```

where `www.example.com/index.html` is replaced with the full object path of the object you're testing.

For example, with optional shielding being used and a TTL set to 86400 (24 hours) using `Surrogate-Control`, the command `curl -svo /dev/null -H "Fastly-Debug:1" www.example.com` produces something like the following section of output:

```
[...]

< Surrogate-Control: max-age=86400
< Surrogate-Key: articles articles/1 articles/2

[...]
< Age: 403
< Fastly-Debug-Path: (D cache-ord1722-ORD 1470672957) (F cache-ord1743-ORD 1470672629) (D ca
< Fastly-Debug-TTL: (H cache-ord1722-ORD 85997.246 0.000 403) (H cache-jfk1041-JFK - - 75)
< X-Served-By: cache-jfk1041-JFK, cache-ord1722-ORD
< X-Cache: HIT, HIT
< X-Cache-Hits: 1, 6

[...]
```

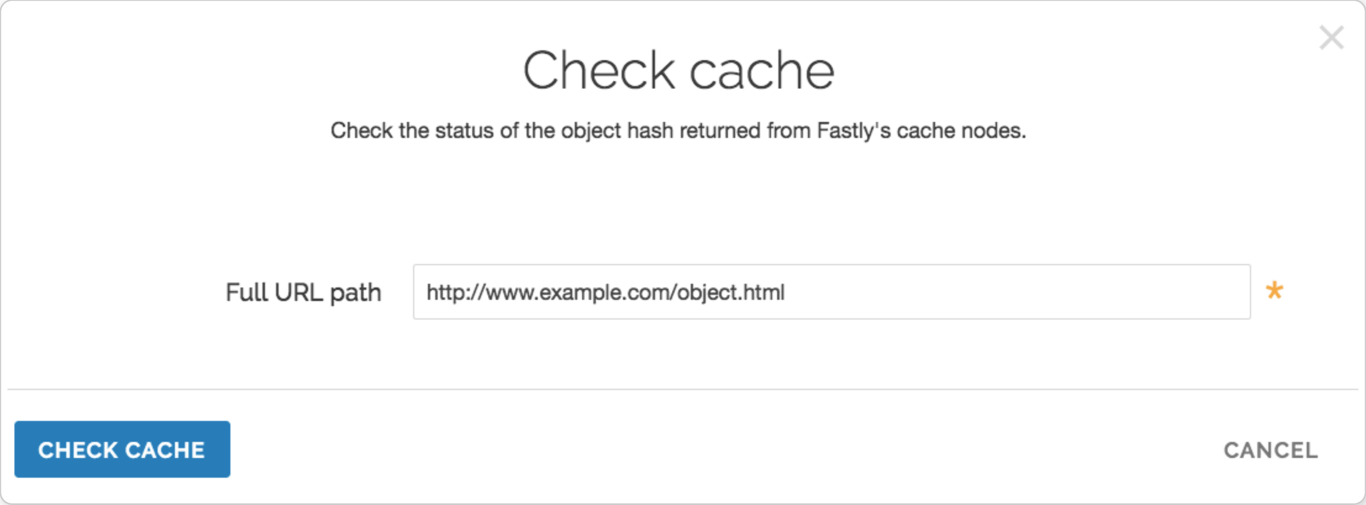
Because surrogate keys are present, the `Fastly-Debug` header exposes them. As with the simple curl command, this section of output tells us the current age of the object in cache. In addition, `Fastly-Debug` exposes specific header details to help with debugging as noted below.

Using the web interface

> WARNING
>
> When you click **Check Cache**, every Fastly cache node will send a request to your origin for the object. Clicking the button multiple times could overwhelm your origin with requests. Consider using curl to check the cache status instead.

Follow the steps below to check the cache status of an object using the Fastly web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Check Cache**.

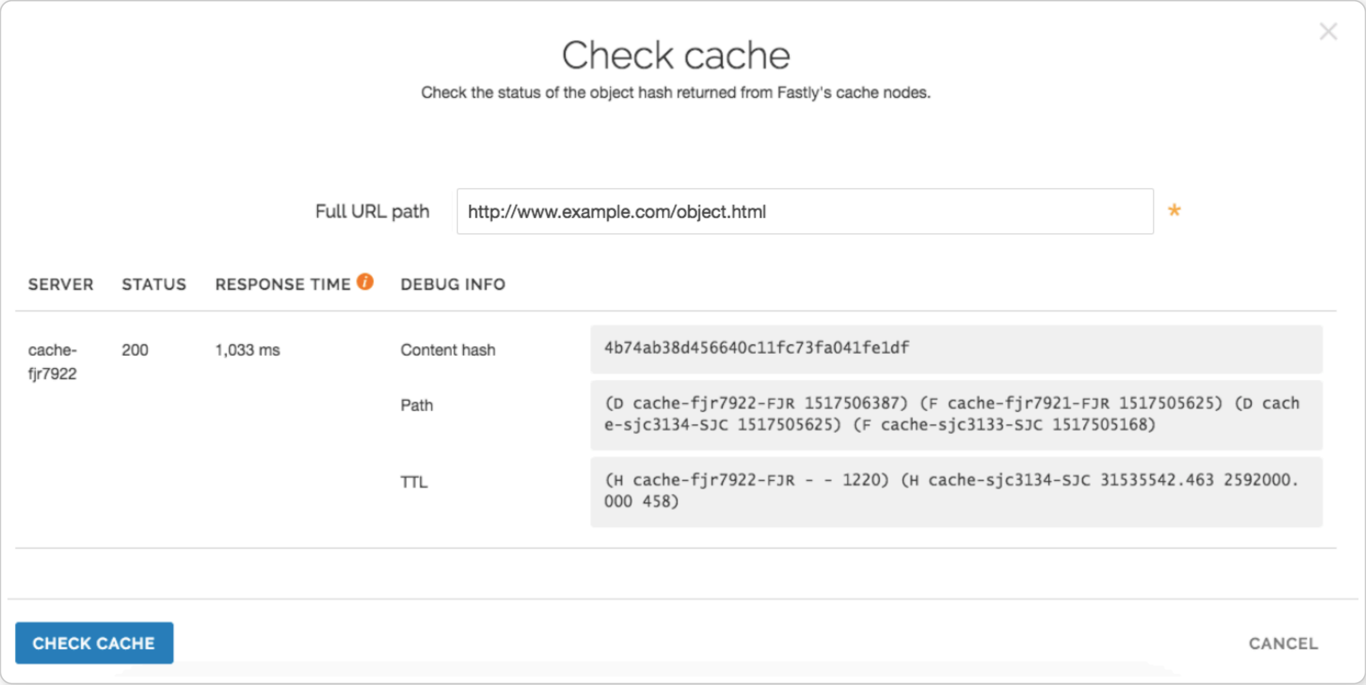


Check cache

Check the status of the object hash returned from Fastly's cache nodes.

Full URL path: http://www.example.com/object.html *

4. In the **Full URL path** field, enter the full path to the object (e.g., `http://www.example.com/object.html`).
5. Click **Check Cache**. The results are displayed in the Check Cache window.



Check cache

Check the status of the object hash returned from Fastly's cache nodes.

Full URL path: http://www.example.com/object.html *

SERVER	STATUS	RESPONSE TIME	DEBUG INFO	
cache-fjr7922	200	1,033 ms	Content hash	`4b74ab38d456640c11fc73fa041fe1df`
			Path	`(D cache-fjr7922-FJR 1517506387) (F cache-fjr7921-FJR 1517505625) (D cache-sjc3134-SJC 1517505625) (F cache-sjc3133-SJC 1517505168)`
			TTL	`(H cache-fjr7922-FJR - - 1220) (H cache-sjc3134-SJC 31535542.463 2592000.000 458)`

You can use this information to verify that the same copy of an object is stored on all of our servers. If the content hash is different across nodes, that usually indicates that there's a caching problem.

Information exposed by the Fastly-Debug header

`Fastly-Debug Path` contains information about which cache server handles fetching and delivery of an object. The edge POP appears first in the sequence and the shield POP appears second.

- `D` represents which cache by name in the edge or shield ran `vcl_deliver`
- `F` represents which cache by name in the edge or shield ran `vcl_fetch`
- the number following each specific server name is a timestamp in seconds

With shielding enabled, you should generally see four cache servers listed in this header. In rare cases where a cache server exists as both an edge and a shield within the cluster for that object, you may see two or three caches listed.

`Fastly-Debug-TTL` provides information on HIT and MISS timings.

- `H` represents a HIT, meaning the object was found in the cache
- `M` represents a MISS, meaning the object was not cached at the time of the query

For each of these timings:

- the first number specifies the TTL remaining for the object
- the second number specifies the grace period
- the third number specifies the current age of the object in cache

It may take a few requests to see these numbers populate as expected because they need to either hit the cluster node or a node where the content already exists in temporary memory.

`X-Served-By` indicates the shield and edge servers that were queried for the request. The shield POP appears first in the sequence and the edge POP appears second.

`X-Cache` indicates whether the request was a HIT or a MISS for the data center.

NOTE

Our shielding debugging documentation provides in depth details key to understanding the `X-Served-By`, `X-Cache`, and `X-Cache-Hits` headers with shielded services.

* * *

Controlling caching

/en/guides/controlling-caching

When we store your content in cache, we calculate a Time to Live (TTL). The TTL is the maximum amount of time we will use the content to answer requests without consulting your origin server. After the TTL expires, we may keep the content in storage, but we won't use it to answer requests unless we're able to revalidate it with your origin.

There's no guarantee that the content will stay cached for the length of time specified in the TTL. Depending on the size of the object and how popular it is relative to other objects, we may delete it before its TTL expires.

For more information about controlling how long Fastly caches your resources, start with our documentation on cache freshness. In general, we will honor any Cache-Control headers you send to us from your origin.

Determining the TTL of an object

You can determine the TTL of an individual object as follows:

- If you've set the `Surrogate-Control: max-age`, `Cache-Control: max-age`, or `Expires` headers, the TTL is whatever you specified in those headers

- If you've specified the TTL in the web interface or custom VCL, the TTL is whatever you specified
- If you've specified the TTL in the web interface or custom VCL and you've set the `Surrogate-Control: max-age`, `Cache-Control: max-age`, or `Expires` headers, the TTL specified in the web interface might override the TTL specified in the origin response
- If you haven't specified the TTL in the web interface or custom VCL and you haven't set the `Surrogate-Control: max-age`, `Cache-Control: max-age`, or `Expires` headers, the TTL is 3600 seconds

You can change the fallback TTL in the service configuration.

Setting different TTLs for Fastly cache and web browsers

Purging objects from the Fastly cache is easy. Clearing the caches of users' web browsers is much harder. For that reason, it can make sense to set different TTLs for content in the Fastly cache versus users' web browsers. You can set different TTLs for the Fastly cache and web browsers through Surrogate-Control headers defined by the W3C. For example, if you wanted Fastly to cache something for a year but you also wanted to set a maximum age of a single day for users viewing that object in a web browser, then you could return the following HTTP headers:

```
Surrogate-Control: max-age=31557600
Cache-Control: max-age=86400
```

The Surrogate-Control header in this example tells Fastly to cache the object for a maximum of 31557600 seconds (one year). The Cache-Control header in this example tells the browser to cache the object for a maximum of 86400 seconds (1 day).

> TIP
>
> CDNs can't invalidate an end user's web browser cache. If your content is going to change frequently or if you want new content immediately available to end users, the best strategy is to cache that content on Fastly and specifically instruct browsers not to cache it. Then, when you purge the Fastly cache, an end user's browser will display the new version the next time the content is requested.
>
> One way to do this is to return the following headers:
>
> ```
> Surrogate-Control: max-age=31557600
> Cache-Control: no-store, max-age=0
> ```
>
> This Surrogate-Control header tells Fastly to cache the object for a maximum of 31557600 seconds (one year). The Cache-Control header tells the browser not to cache the object.

For Surrogate-Control, Fastly supports the `max-age`, `stale-if-error`, and `stale-while-revalidate` parameters.

For more information about controlling caching, check out our documentation on cache freshness.

Conditionally preventing pages from caching

To conditionally prevent pages from caching, follow the steps below.

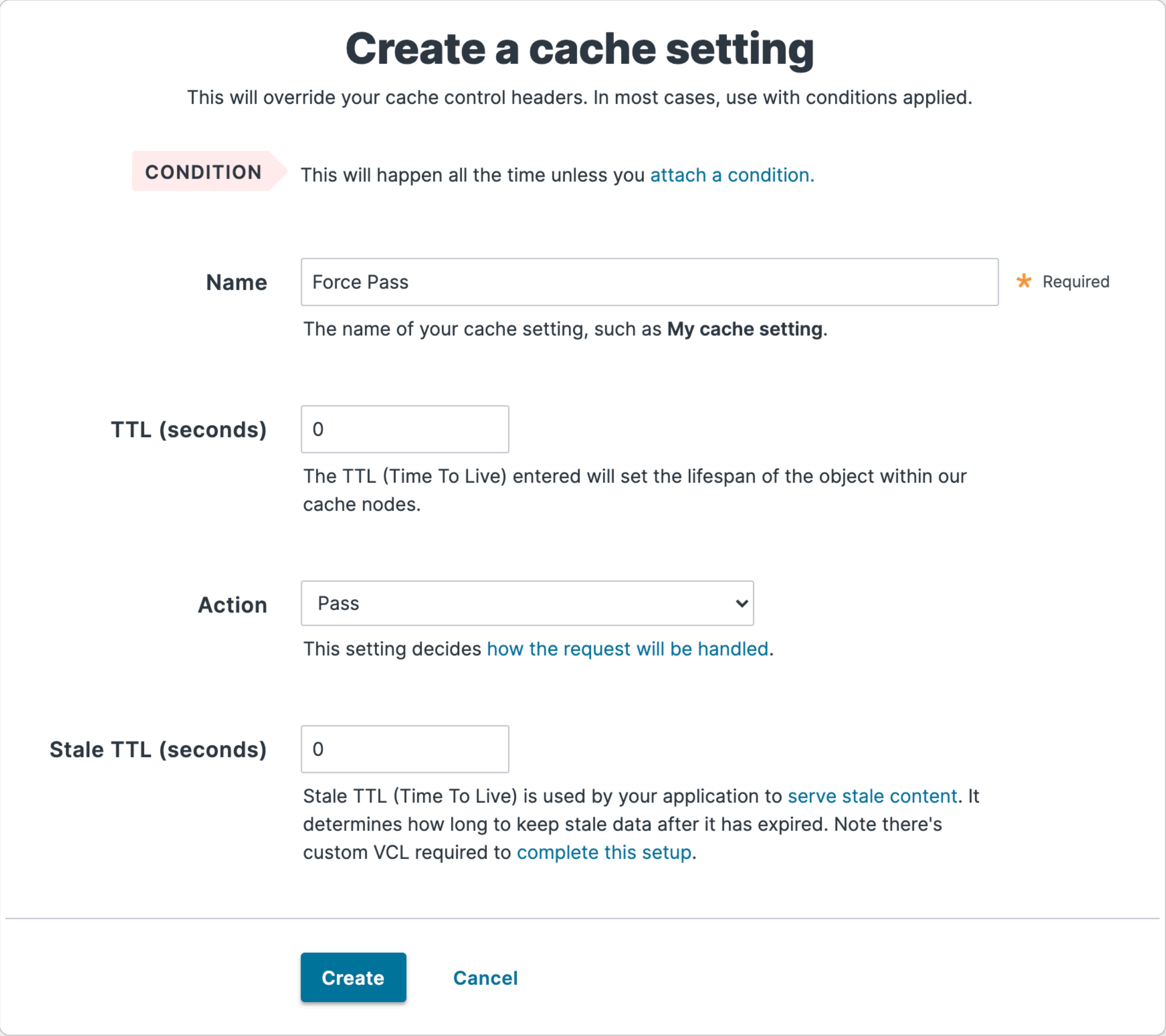
1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Settings**.

5. Click **Create cache setting** to create a new cache setting.



Create a cache setting

This will override your cache control headers. In most cases, use with conditions applied.

CONDITION This will happen all the time unless you attach a condition.

Name: Force Pass * Required

The name of your cache setting, such as **My cache setting**.

TTL (seconds): 0

The TTL (Time To Live) entered will set the lifespan of the object within our cache nodes.

Action: Pass

This setting decides how the request will be handled.

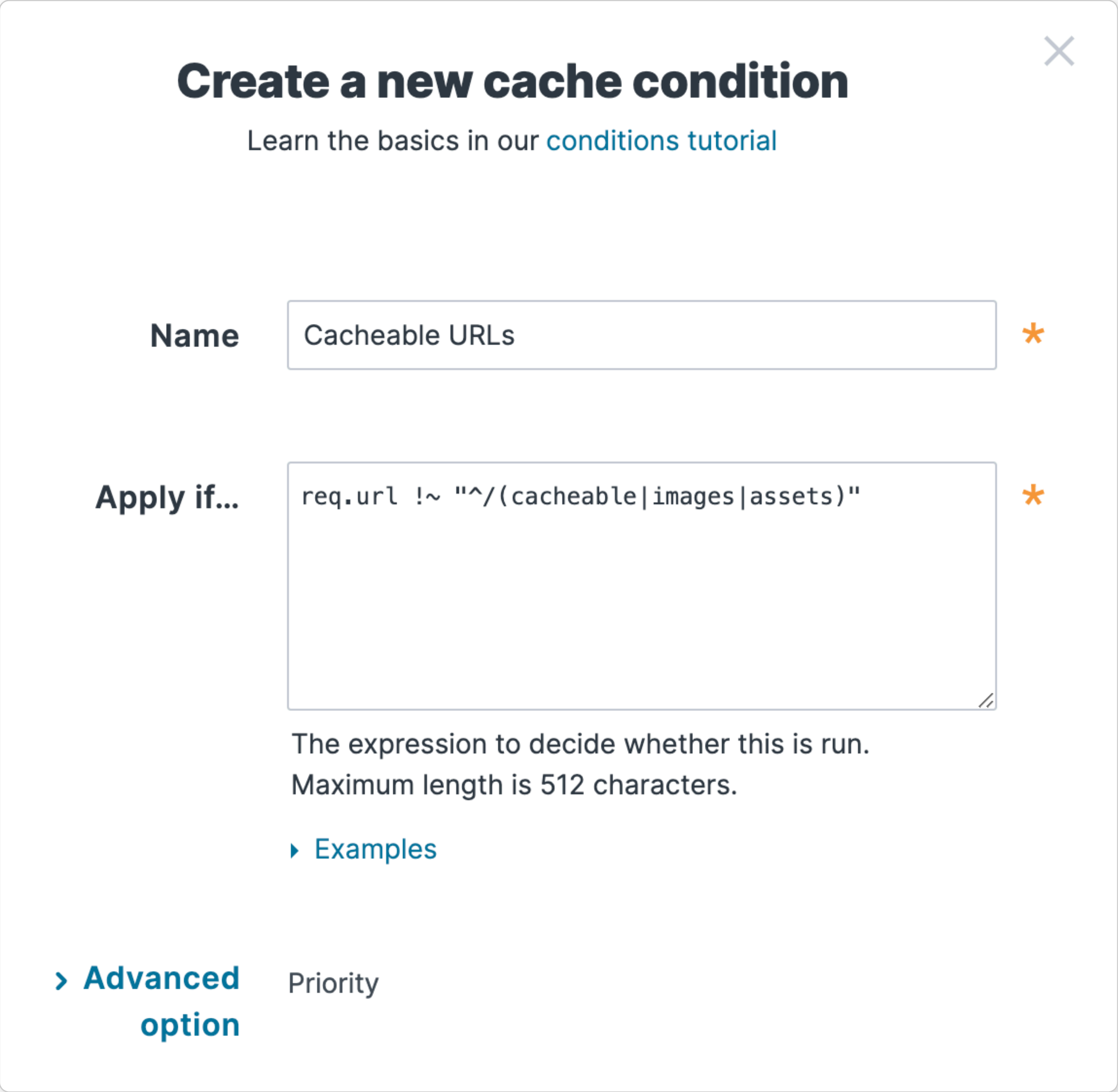
Stale TTL (seconds): 0

Stale TTL (Time To Live) is used by your application to serve stale content. It determines how long to keep stale data after it has expired. Note there's custom VCL required to complete this setup.

6. Fill out the **Create a cache setting** fields as follows:
   - In the **Name** field, enter a descriptive name for the new cache setting (e.g., `Force Pass`).
   - In the **TTL (seconds)** field, enter `0` to set the lifespan of the object in our cache nodes to zero.
   - From the **Action** menu, select **Pass (do not cache)** to pass the request and avoid caching it.
   - In the **Stale TTL (seconds)** field, enter `0` to set the amount of time to serve stale content, in seconds, to zero.

7. Click **Create**.

8. Click the **Attach a condition** link to the right of the newly created cache setting.



Create a new cache condition
Learn the basics in our conditions tutorial
Name
Apply if...
The expression to decide whether this is run.
Maximum length is 512 characters.
Examples
Advanced option
Priority

Name: Cacheable URLs

Apply if...: `req.url !~ "^/(cacheable|images|assets)"`

9. Fill out the **Create a new cache condition** fields as follows:
   - In the **Name** field, enter a descriptive name for the condition (e.g., `Cacheable URLs`).
   - In the **Apply if** field, create a condition that matches the URLs to not cache. For example, you could enter `req.url !~ "^/(cacheable|images|assets)"` to set the condition to look for URLs that do not start with `/cacheable`, `/images`, or `/assets`. If the condition finds them, the URLs should be cached. If the condition doesn't find them, the cache setting that is set to `Pass` ensures the URLs are explicitly not cached.

10. Click **Save and apply to**.

11. Click **Activate** to deploy your configuration changes.

TIP

You can use these steps to override default caching based on a backend response.

* * *

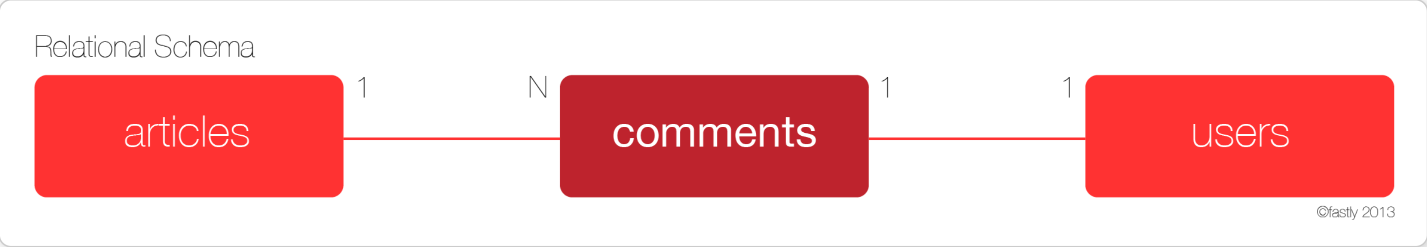
Enabling API caching

/en/guides/enabling-api-caching

Application Programming Interfaces (APIs) allow you to retrieve data from a variety of web services. Fastly makes it possible for you to cache your API so you can accelerate the performance of your service-oriented architecture. It optimizes your API's performance by efficiently handling traffic bursts and reducing latency.

An example

Let's look at an example to learn how API caching works. Imagine we're an online magazine with articles on which users can make comments. Each article can have many comments, and each comment is created by exactly one user.



We'll design a RESTful API specification and use it to manipulate and retrieve comments:

- `GET /comment` - Returns a list of all comments
- `GET /comment/:id` - Returns a comment with the given ID
- `POST /comment` - Creates a new comment
- `PUT /comment/:id` - Updates a comment with the given ID
- `DELETE /comment/:id` - Deletes a comment with the given ID

The create, read, update, and delete (CRUD) methods ensure the API can perform its basic operations, but they don't expose the relational aspect of the data. To do so, you would add a couple of relational endpoints:

- `GET /articles/:article_id/comments` - Get a list of comments for a given article
- `GET /user/:user_id/comments` - Get all comments for a given user

Endpoints like these allow programmers to get the information they need to do things like render the HTML page for an article, or display comments on a user's profile page. While there are many other possible endpoints we could construct, this set should suffice for the purposes of this guide. Let's assume that the API has been programmed to use an Object-Relational Mapper (ORM), such as ActiveRecord, when interacting with the database.

Determining which API endpoints to cache

Start by identifying the URLs you want to cache. We recommend splitting the specification endpoints into two groups.

The first group, called *accessors*, retrieves or accesses the comment data. These are the endpoints you want to cache using Fastly. Using the example, four endpoints match this description:

- `GET /comment`
- `GET /comment/:id`
- `GET /article/:article_id/comments`
- `GET /user/:user_id/comments`

The second group, called *mutators*, changes or mutates the comment data. These endpoints are always dynamic, and are therefore uncacheable. Using the example, three endpoints match this description:

- `POST /comment`
- `PUT /comment/:id`
- `DELETE /comment/:id`

You should see a pattern emerging. Because the example API is RESTful, we can use a simple rule to identify the accessor and mutator endpoints: GET endpoints can be cached, but PUT, POST, and DELETE endpoints cannot.

Once you've gathered this information, you're ready to program the API to configure PURGE requests.

Configuring PURGE requests

Don't be tempted to point at the PUT, POST, and DELETE endpoints as the place where data is modified. In most modern APIs, these endpoints represent an interface to the actual model code responsible for handling the database modifications.

In the example, we assumed that we'd be using an ORM to perform the actual database work. Most ORMs allow programmers to set special *callbacks* on models that will fire when certain actions have been performed (e.g., before or after validation, or after creating a new record).

For purging, we are interested in whether a model has saved information to the database — whether it's a new record, an update to an existing record, or the deleting of a record. At this point, we'd add a callback that tells the API to send a PURGE request to Fastly for each of the cacheable endpoints.

For an ActiveRecord comments model, you could do something like this:

```ruby
require 'fastly'

class Comment < ActiveRecord::Base
  fastly = Fastly.new(api_key: 'FASTLY_API_TOKEN')

  after_save do
    fastly.purge "/comment"
    fastly.purge "/comment/#{self.id}"
    fastly.purge "/article/#{self.article_id}/comments"
    fastly.purge "/user/#{self.user_id}/comments"
    end
end
```

Keep two things in mind when creating the callback:

- The purge code should be triggered *after* the information has been saved to the database, otherwise a race condition could be created where Fastly fetches the data from the origin server before the data has been saved to the database. This would cache the old data instead of the new data.
- These URLs are being purged because they have content that changes when a comment is changed.

With the model code in place, the API is now ready to be cached.

Setting up Fastly

The final step to enabling API caching involves setting up Fastly. You'll need to:

- Create a new service
- Add the domain for the API
- Add the origin server that powers the API

In addition, you can optionally create rules that tell Fastly how to work with the specific elements that are exclusive to your API.

> NOTE
>
> By default, Fastly will not cache PUT, POST, and DELETE requests.

Creating a new service

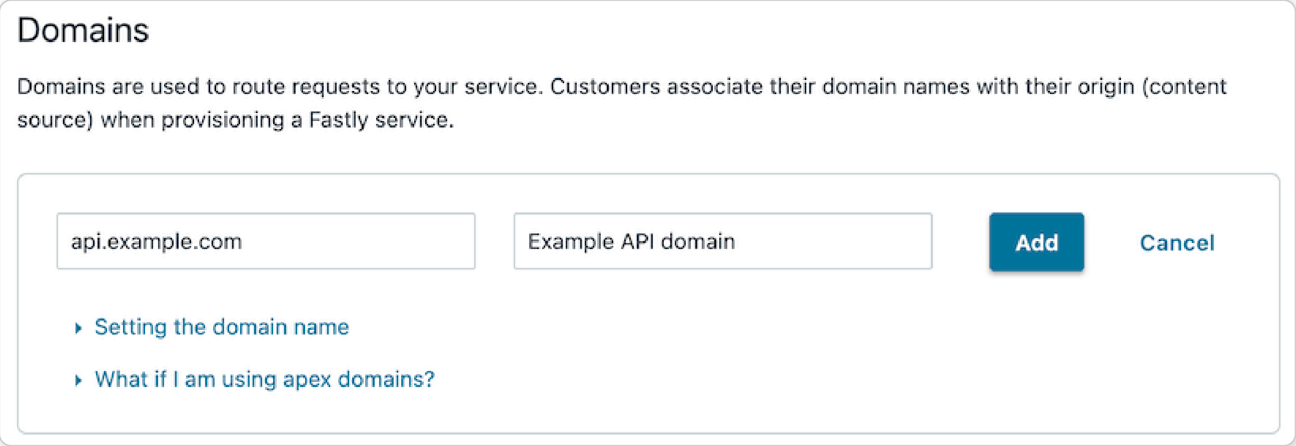
Follow the instructions for creating a new service. You'll add specific details about your API server when you fill out the **Create a new service** fields:

- In the **Name** field, enter a name for this service that helps you identify it's related to caching your API information (e.g., `My API Service`).
- In the **Domain** field, enter the domain name associated with your API (e.g., `api.example.com`).
- In the **Address** field, enter the IP address or hostname of your API server.

Adding the domain

Follow these instructions to add the API's domain name to your Fastly service:

1. Click **Edit configuration** and then select the option to clone the active version.
2. Click **Create domain**.


Domains
Domains are used to route requests to your service. Customers associate their domain names with their origin (content source) when provisioning a Fastly service.
api.example.com
Example API domain
Add
Cancel
Setting the domain name
What if I am using apex domains?

3. Fill out the **Create a domain** fields as follows:
   - In the **Domain Name** field, enter the domain name for the API.
   - In the **Comment** field, enter an optional comment that describes your domain.
4. Click **Create**. Your API's domain name appears in the list of domains.

Adding the origin server

Follow the instructions for working with hosts. You'll add specific details about your API server when you fill out the **Create a host** fields:

- In the **Name** field, enter a name for the origin server that helps you identify it's related to caching your API information.
- In the **Address** field, enter the IP address (or hostname) of the API server.

* * *

Implementing API cache control

/en/guides/implementing-api-cache-control

This guide explains how to implement API cache control. Once you've enabled API caching, and ensured purging works properly with your cached data, you can set up specific headers like Cache-Control and Surrogate-Control to change when data is cached.

Understanding Cache-Control headers

In general, we assume that GET requests are cached and PUT, POST, and DELETE requests are not. For an ideal REST API, this rule works well. Unfortunately, most APIs are far from ideal and require additional caching rules for some requests.

For these reasons, it's a good idea to set Cache-Control headers when migrating APIs to Fastly. Cache-Control, as defined by RFC 7234 (the HTTP specification), includes many different options for appropriate handling of cached data. Specifically, Cache-Control headers tell user agents (e.g., web browsers) how to handle the caching of server responses. For example:

- `Cache-Control: private`
- `Cache-Control: max-age=86400`

In the first example, `private` tells the user agent the information is specific to a single user and should not be cached for other users. In the second example, `max-age=86400` tells the user agent the response can be cached, but that it expires in exactly 86,400 seconds (one day).

Fastly respects Cache-Control headers by default, but you can also use another proxy-specific header: Surrogate-Control. Surrogate-Control headers are similar to Cache-Control headers, but provide instructions to reverse proxy caches like Fastly. You can use Cache-Control and Surrogate-Control headers together. For more information about Cache-Control and Surrogate-Control headers, see our documentation on cache freshness.

An updated example

Let's take a look at how the Cache-Control headers could be used in our original example, the comments API. Recall the API endpoint that provided a list of comments for a given article:

`GET /article/:article_id/comments`

When a user submits a comment for a given article, the response from this endpoint will be purged from the Fastly cache by the comment model. It's hard to predict when content will change. Therefore, we'd like to ensure the following:

1. If the content doesn't change, it should stay cached in Fastly for a reasonable amount of time.
2. If the content does change, it should not be cached by the client longer than it needs to be.

The goal is to ensure that API responses will reach clients in a timely manner, but we also want to ensure that clients always have the most up-to-date information. The first constraint can be solved by using the Surrogate-Control header, and the second constraint can be solved by using the Cache-Control header:

```
Surrogate-Control: max-age=86400
Cache-Control: max-age=60
```

These headers tell Fastly that it is allowed to cache the content for up to one day. In addition, the headers tell the client that it is allowed to cache the content for 60 seconds, and that it should go back to its source of truth (in this case, the Fastly cache) after 60 seconds.

Implementing cache control

Migrating APIs isn't easy, even for experienced teams. When migrating an API to Fastly, we recommend separating the task into three strategic endpoint migrations to make the process more manageable while still maintaining the validity of the API as a whole.

Preparing the API

To ensure that the API bypasses the cache during the piecewise migration, we must have every API endpoint return a specific control header:

```
Cache-Control: private
```

This header tells Fastly that a request to any endpoint on the API should bypass the cache and be sent directly to the origin. This will allow us to serve the API via Fastly and have it work as expected.

> NOTE
>
> Modern web frameworks allow for blanket rules to be overridden by specific endpoints (for example, by the use of middlewares). Depending on how the API has been implemented, this step might be as simple as adding a single line of code.

Serving traffic with Fastly

The next step is configuring a Fastly service to serve the API's traffic. After you save the configuration, there will be an immediate speed improvement. This happens because Fastly's cache servers keep long-lasting connections to the API's origin servers, which reduces the latency overhead of establishing multiple TCP connections.

Migrating endpoints

Now we can implement instant purge caching for each cacheable API endpoint, one at a time. The order in which this is done depends on the API, but by targeting the slowest endpoints first, you can achieve dramatic improvements for endpoints that need them the most. Because each endpoint can be worked on independently, the engineering process is easier to manage.

Excluding endpoints

The last step is deciding which API endpoints you don't want Fastly to cache. To disable caching for endpoints, you'll need to add new conditions for the endpoints. As you learned in Preparing the API, using the `Cache-Control: private` header is another option for disabling caching.

* * *

Overriding caching defaults based on a backend response

/en/guides/overriding-caching-defaults-based-on-a-backend-response

---

In certain situations you may want to conditionally apply a different caching policy based on a backend response. In this particular case we have backend that on occasion returns 404 errors (e.g., document not found). We don't want those responses to be cached for the full caching period of a day but only for 5 minutes. To override default caching we add a cache object and then create conditions for it.

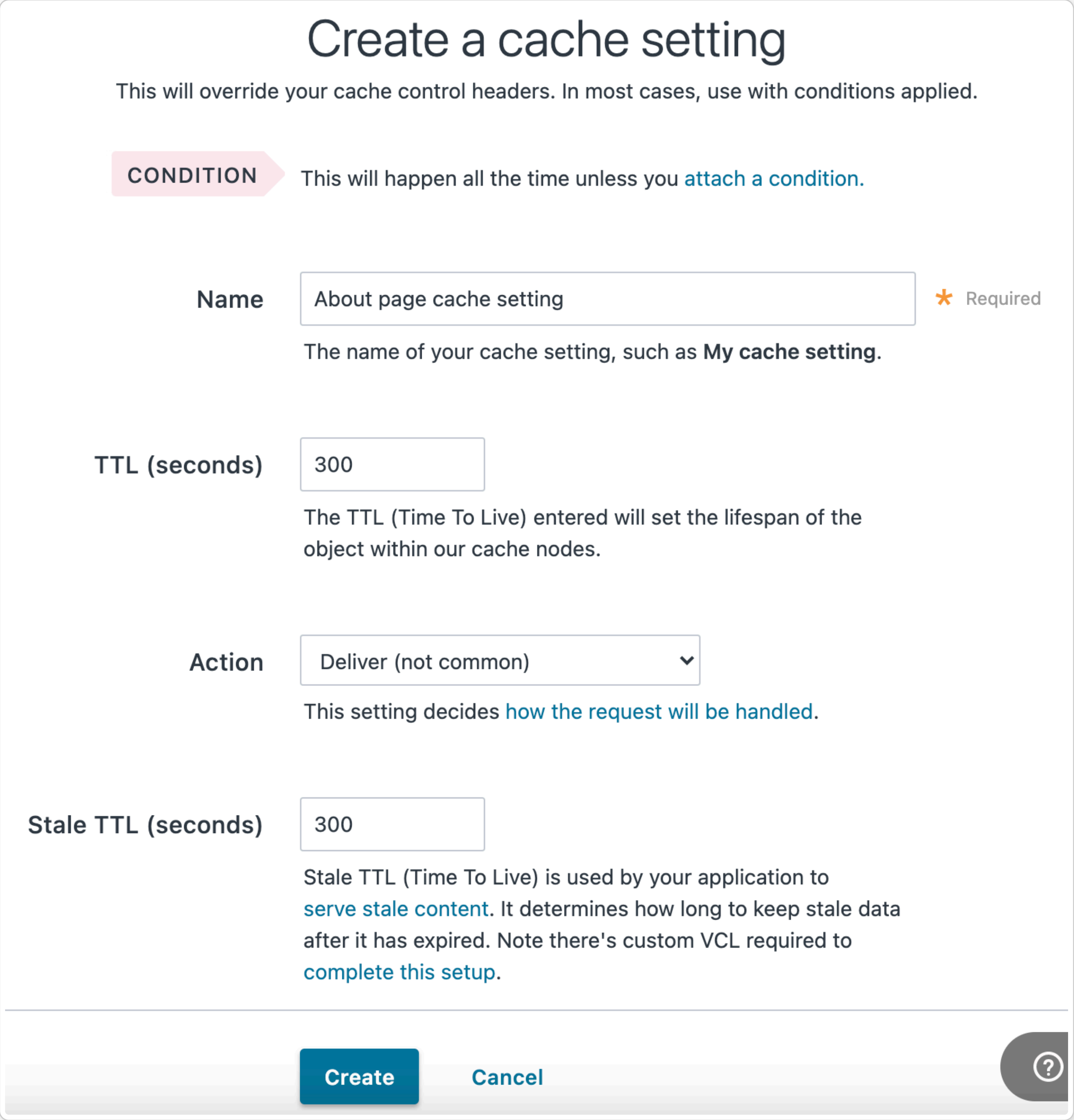
Creating the new Cache Object

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Settings**.

5. Click **Create cache setting**.



Create a cache setting

This will override your cache control headers. In most cases, use with conditions applied.

CONDITION This will happen all the time unless you attach a condition.

Name: About page cache setting * Required

The name of your cache setting, such as **My cache setting**.

TTL (seconds): 300

The TTL (Time To Live) entered will set the lifespan of the object within our cache nodes.

Action: Deliver (not common)

This setting decides how the request will be handled.

Stale TTL (seconds): 300

Stale TTL (Time To Live) is used by your application to serve stale content. It determines how long to keep stale data after it has expired. Note there's custom VCL required to complete this setup.

6. Fill out the **Create a cache setting** fields as follows:

   - In the **Name** field, enter a descriptive name for the new cache settings.

   - In the **TTL (seconds)** field, enter the amount of time, in seconds, to cache the objects (e.g., `300`).

   - From the **Action** menu, select **Deliver**.

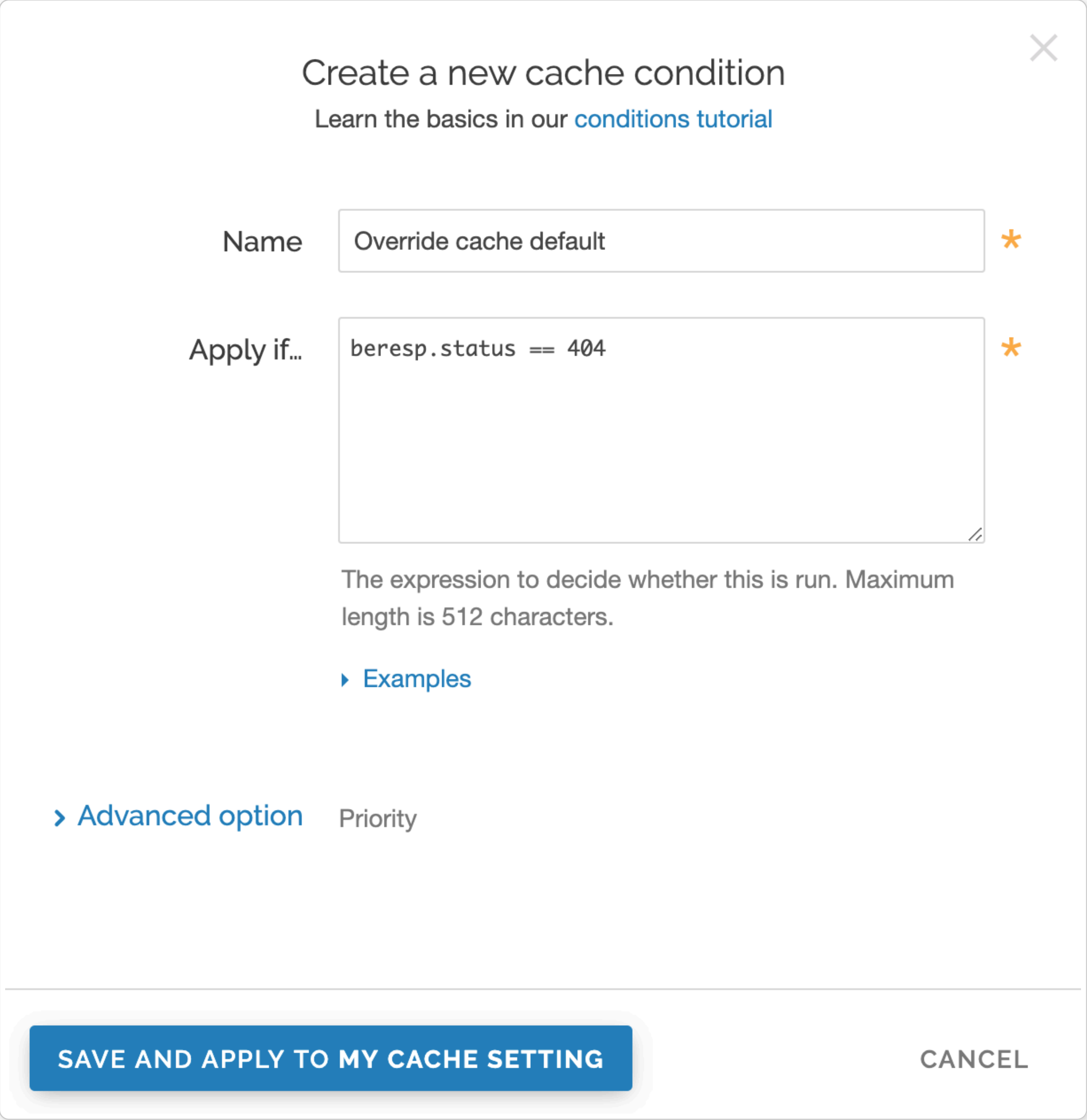
  - In the **Stale TTL (seconds)** field, enter the amount of time to serve stale or expired responses, in seconds, should the backend become unavailable (e.g., `300`).

7. Click **Create**.

Creating an Override Condition for the new Cache Object

Once the object is created, add a condition to it.

1. Click **Attach a condition** to the right of the object.
2. Click **Create cache setting**.



Create a new cache condition

Learn the basics in our conditions tutorial

Name: Override cache default

Apply if...: `beresp.status == 404`

The expression to decide whether this is run. Maximum length is 512 characters.

3. Fill out the **Create a new cache setting** fields as follows:

  - In the **Name** field, enter a descriptive name for the new condition. For example, `Override cache default`.
  - In the **Apply if** field, enter an appropriate backend response header to specify when the condition will be applied. For example, `beresp.status == 404`.

4. Click **Save and apply to**.
5. Click **Activate** to deploy your configuration changes.

Other notes

You can use any backend response header in the **Apply if** field to make decisions on caching.

For example, `beresp.http.Content-Type ~ "^text/html"` can be used to specify different caching rules for HTML documents.

* * *

Preventing cache poisoning via HTTP X-headers

🔗 /en/guides/preventing-cache-poisoning

Fastly service configurations may be vulnerable to cache poisoning if they do not take into consideration the interaction between HTTP "X-" headers used by backends to select content. This vulnerability can be mitigated using a VCL patch or by modifying backend configurations.

When cache poisoning can occur

If one or more of your backends uses the contents of the X-Forwarded-Host, X-Rewrite-URL, or X-Original-URL HTTP request headers to decide which of your users (or which security domain) it sends an HTTP response for, you could be impacted by vulnerability. If your site's Fastly configuration passes one of these headers to your backend and does not factor the contents of it into the effective edge cache key (for example, explicitly or via the Vary HTTP response header), an attacker could potentially cause the edge to store a response with arbitrary content inserted into a victim's cache.

An attacker might be able to poison a Fastly customer URL by sending an HTTP request to a site that causes the affected backend to respond with an attacker-controlled response. The malicious response object would be stored in the site's cache at a poisoned URL. An attacker could then potentially lure a victim site user into browsing the poisoned URL, where they would be served malicious content.

How to mitigate cache poisoning

If your origin uses special values to select content for users or to otherwise select between security domains, consider reconfiguring your origin server and applying any corresponding security updates as suggested in our original security advisory. In addition, strip or normalize the values of X-Forwarded-Host, X-Rewrite-URL, or X-Original-URL in VCL.

To do this, set the vulnerable headers to a known-safe value or unset the headers completely. For example, the X-Forwarded-Host header can be set to the value of the Host header via the following VCL snippet:

```
set req.http.x-forwarded-host = req.http.host;
```

The X-Original-URL header can be unset via the following VCL snippet:

```
unset req.http.x-original-url;
```

And X-Rewrite-URL can be unset via the following VCL snippet:

```
unset req.http.x-rewrite-url;
```

Alternatively, the values could be included in your cache key or Vary header to prevent caching of content across security domains. See our guide to manipulating the cache key for more information.

* * *

Segmented Caching

/en/guides/segmented-caching

Fastly's Segmented Caching feature allows you to cache resources of any size. Segmented Caching works by breaking resources into smaller segments in Fastly's cache then recombining or splitting these resources to respond to arbitrary size byte `Range:` requests from clients. Once enabled, Segmented Caching improves performance for `Range:` requests and allows Fastly to efficiently cache resources of any size.

⚠ WARNING

Fastly recommends enabling Segmented Caching on services that will be serving large resources. Without Segmented Caching enabled, the resource size limits for your account depend on when you become a Fastly customer:

- If you created your account on or after June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum object size of 20 MB.
- If you created your account prior to June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum cacheable object size of 2 GB for requests without Streaming Miss or 5 GB for requests with Streaming Miss.

How Segmented Caching works

When an end user makes a `Range:` request for a resource with Segmented Caching enabled and a cache miss occurs (that is, at least part of the range is not cached), Fastly will make the appropriate `Range:` requests back to origin. Segmented Caching will then ensure only the specific portions of the resource that have been requested by the end user (along with rounding based on object size) will be cached rather than the entire resource. Partial cache hits will result in having the cached portion served from cache and the missing pieces fetched from origin. (Requests for an entire resource would be treated as a byte `Range:` request from 0 to end of resource.)

Once Fastly has all of the objects necessary to respond to an end user's request, the Segmented Caching feature will assemble the response by concatenating or pulling portions of objects. The requests back to origin, also called "inner requests," will have a `true` value for `segmented_caching.is_inner_req` and requests from end users, also called "outer requests," will have a `true` value for `segmented_caching.is_outer_req`. If a request is made for an object without segmented caching enabled, both variables will have a FALSE value.

> NOTE
>
> The feature will only go back to origin for missing objects needed to handle the end-client's byte `Range:` request. Cache hits will occur based on having that portion of resource in cache even if the end user's range exact request is unique.

Limitations and considerations

This feature has the following limitations you should take into account:

- **Segmented Caching isn't intended for dynamically generated content.** If your content isn't fully static (e.g., you have dynamic content like user-generated comments), be sure to set up purging any time the content changes to avoid any potential for data corruption.
- **Segmented Caching is not compatible with object compression and ESI.** To use either of these features, you must ensure Segmented Caching is disabled.
- **Segmented Caching is not compatible with Fastly's Image Optimizer (IO).** If IO is enabled, Segmented Caching is disabled automatically.
- **HTTP chunked transfer encoding between Fastly and origin isn't supported.** Your origin server must frame responses to `Range:` requests with the `Content-Length` header.
- **URL purges must be authenticated.** Segmented caching allows you to purge all range objects for the resource by URL purge, but authentication for URL purge needs to be enabled due to its underlying implementation. Make sure you've provided an authorization token for URL purges as described in our purging documentation.
- **Segmented Caching cannot be enabled based on resource size.** The VCL code to enable Segmented caching must run *before* the resource is requested from cache, so it is not possible to know how large it will be. However, it is possible to make segmented caching conditional upon the URL (e.g. `/video/`) or file extension (e.g. *.m4v), and this is a common use case for resources which will benefit from it. Our instructions below contain an example of how to enable the Segmented Caching feature based on extension.

This feature also has the following considerations you should take into account:

- **Resources cached prior to enabling Segmented Caching are not used.** If you are enabling Segmented Caching on an existing service, whole resources already in cache are ignored and Varnish will go back to origin to build range request objects. You can choose to purge these or ignore them to be aged out of cache.
- **Your cache hit ratio (CHR) will appear lower than it actually is** because only outer requests are used in the calculation. For example, if there is a request for the first 100 MB of a resource but Fastly only has 99 MB of 100 MB in cache, the entire request will be counted as a miss in the CHR stat. In practice, only 1 MB had to be fetched from origin and you experienced 99% origin offload.

Enabling Segmented Caching

Use the following steps to enable Segmented Caching.

1. Determine which resources should use Segmented Caching.

   > TIP
   >
   > We recommend focusing on a set of file extensions or a well-defined URL structure that distinguishes the resources.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create your first VCL snippet**.
6. Fill out the **Create a VCL snippet** fields as follows:
   - In the **Name** field, enter an appropriate name (e.g., `Enable segmented caching`).
   - From **Type (placement of the snippets)** menu, select **within subroutine**.
   - From the **Select subroutine** menu, select **recv (**`vcl_recv`**)**.
   - In the **VCL** field, add a VCL snippet that sets the `req.enable_segmented_caching` VCL variable to `true` in `vcl_recv`. For example, to ensure proper caching of the large resources you've identified that contain MPEG-2-compressed video data, you could add this VCL snippet in `vcl_recv`:

   ```
   # my custom enabled Segmented Caching code
   if (req.url.ext == "ts") {
     set req.enable_segmented_caching = true;
   }
   ```

   This snippet tells Fastly to look for requests for files with the `ts` extension and then enable Segmented Caching for those files.

7. Click **Create** to create the snippet.
8. Click **Activate** to deploy your configuration changes.

* * *

Temporarily disabling caching

/en/guides/temporarily-disabling-caching

Caching can be disabled:

- at the individual URL level,
- at the browser level, and

- at the site level.

Disabling caching at the individual URL level

To disable caching at the individual URL level:

1. Create a request setting that always forces a pass.
2. Add a condition to the request setting that looks for specific URLs.
3. Activate the new version of your service to enable the setting.

Disabling caching at the browser level

Theoretically, all browsers should follow the stated rules of the HTTP standard. In practice, however, some browsers don't strictly follow these rules. The following combination of headers seems to force absolutely no caching with every browser we've tested.

```
Cache-Control: no-cache, no-store, private, must-revalidate, max-age=0, max-stale=0, post-ch
Pragma: no-cache
Expires: 0
```

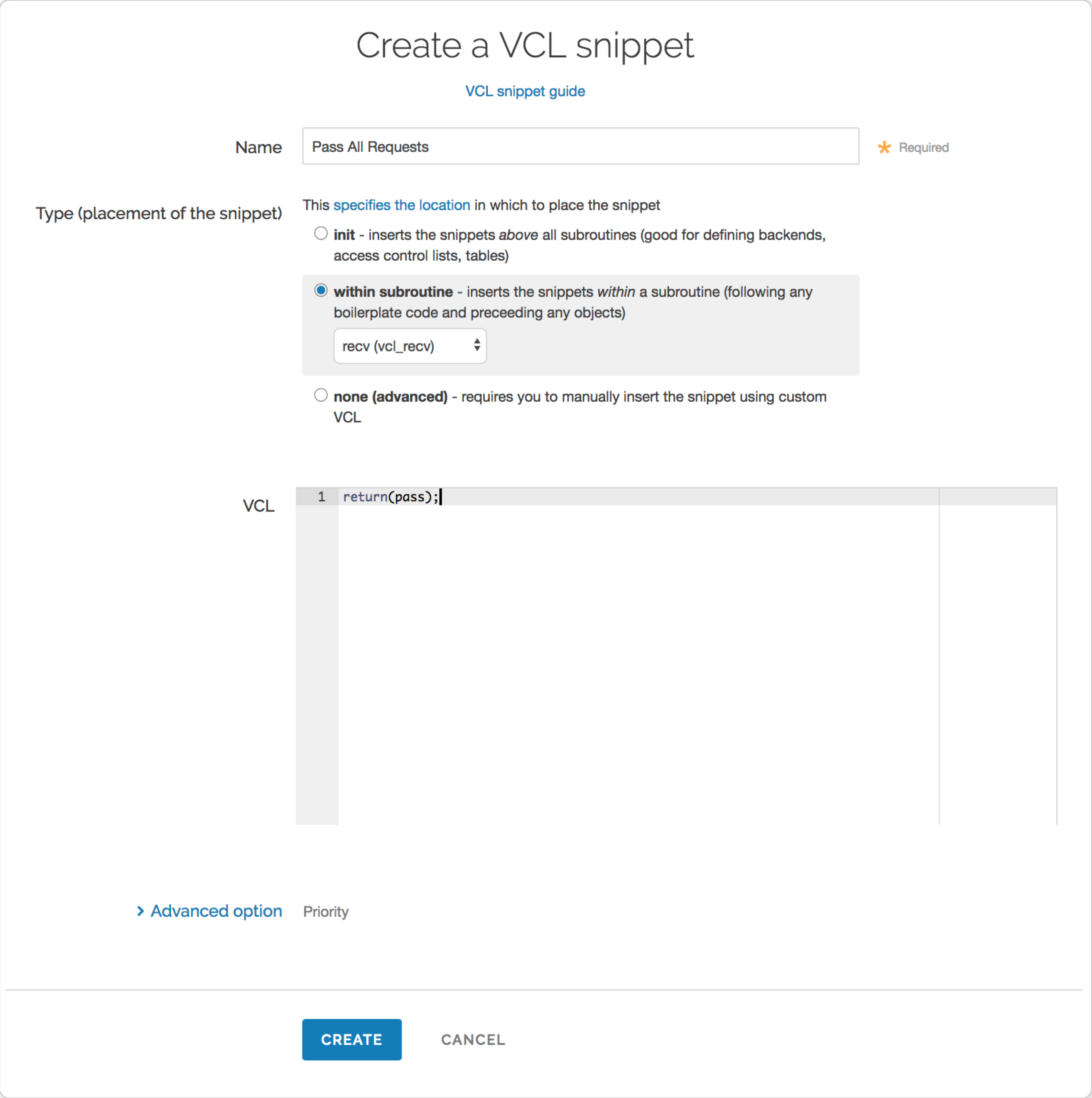
In addition, IE8 has some odd behavior to do with the back button. Adding `Vary: *` to the headers seems to fix the problem.

> IMPORTANT
>
> If you want your content cached in Fastly but not cached on the browser, you must not add these headers on your origin server. Instead, add these as new Headers on the Content page and be sure the Type is set to Response.

Disabling caching at the site level

You can disable caching at the site level by creating a VCL Snippet to pass on all requests to your service:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create Snippet**.



Create a VCL snippet

VCL snippet guide

Name: Pass All Requests

Type (placement of the snippet): This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

recv (vcl_recv)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
return(pass);
```

6. Fill out the Create a VCL snippet fields as follows:
   - In the **Name** field, enter an appropriate name (e.g., `Pass All Requests`).
   - From the **Type** controls, select **within subroutine**.
   - From the **Select subroutine** menu, select **recv (**`vcl_recv`**)**.
   - In the **VCL** field, add the following condition:

```
return(pass);
```

7. Click **Create** to create the snippet.
8. Click **Activate** to deploy your configuration changes.

All requests will continue to be passed until you remove return(pass); from `vcl_recv` in your VCL or you delete this snippet.

* * *

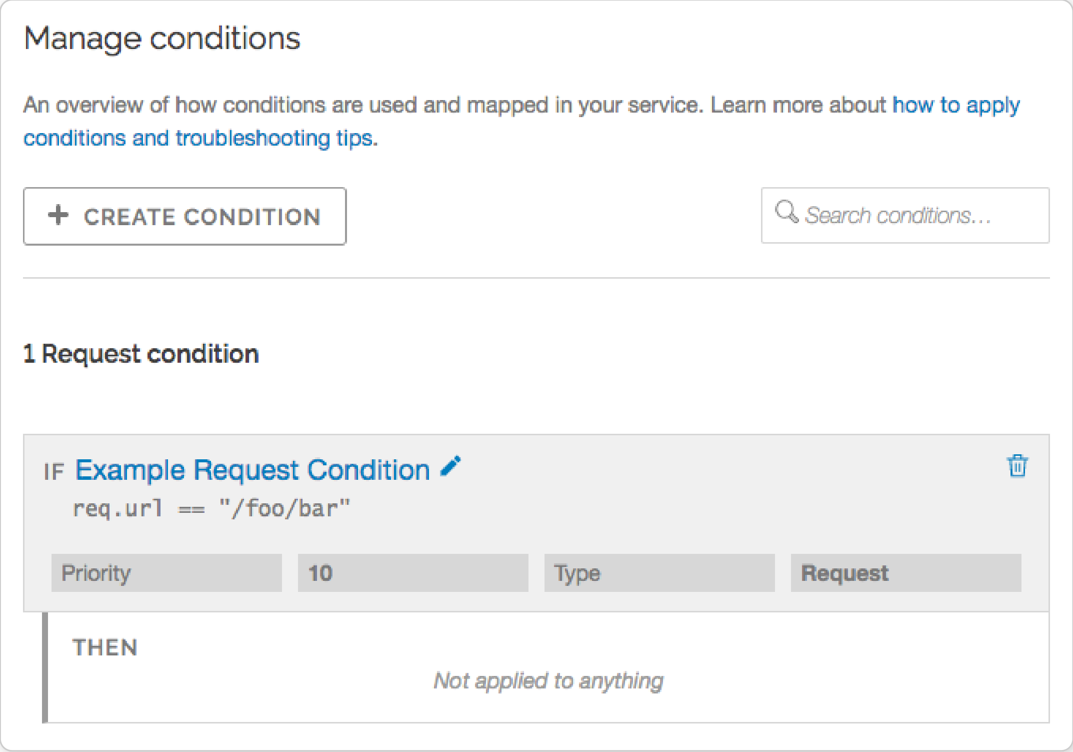
Category: Conditions

These articles describe conditions and how to use them in VCL and the Fastly web interface.

About conditions

/en/guides/about-conditions

Conditions are a configuration mechanism that you can add to your service to control how particular requests are processed. Like conditions in general programming, Fastly conditions use IF-statements to evaluate requests using the logic you specify and formulate a response based on whether the request meets the condition (TRUE) or does not meet the condition (FALSE).



Manage conditions

An overview of how conditions are used and mapped in your service. Learn more about how to apply conditions and troubleshooting tips.

1 Request condition

IF Example Request Condition

`req.url == "/foo/bar"`

Priority 10 Type Request

THEN

Not applied to anything

You can apply a condition using the Conditions feature of the web interface or by using a VCL IF statement on a snippet or in custom VCL

Before you start using conditions

Be sure you understand the construction of basic logical expressions before you start using conditions. Specifically, you should understand basic C-style logical expression syntax (e.g., basic logic, operators such as && and precedence) when working with conditions. A basic programming guide that deals with IF style expressions in either the C or Perl language (the Tizag Perl tutorial is a good one to start with). Even though they aren't directly applicable to our condition examples, the syntax of these languages is similar to VCL.

A simple condition example

The simplest way to explain how Fastly handles conditions is this IF statement:

```
IF
   this condition happens
THEN
   respond this way
```

A practical example can demonstrate this. The vast majority of the time, your site processes requests for information as usual, but every so often customers mistype a search term or simply can't find what they're looking for and you're forced to display a 404 Not Found error. You've realized that when that happens, the standard 404 Not Found errors on your website aren't as helpful as they could be. To fix this, any time your server can't find what a customer is looking for (a condition), you want to display a customized 404 message instructing customers to contact your support team for help (a response).

In plain English, the IF statement might look like this:

```
IF
   404 Not Found is what we have to tell the customer
THEN
   respond with the special Contact Support page
```

The IF line in the example above is the *condition* you've set. The THEN line describes what will happen if that condition is met.

If you were to replace the English in the example above with VCL variables and a little bit of HTML, it might look like this instead:

```
IF
  beresp.status == 404
THEN
  respond with <html><body><h1>Can't find it?</h1><p>Contact support@example.com for he
```

Interested in doing this? We have step-by-step instructions for creating error pages with custom responses.

Ideas for using conditions

Need some more ideas for when you could use conditions? Explore these:

Condition	Response	Learn how
A web robot wants to crawl a particular area of your website	Provide a customized robots.txt file defining which areas of your website should not be processed or scanned	Creating and customizing a robots.txt file

Condition	Response	Learn how
Your server needs to return a 404 Not Found response	Change the default caching time for only 404 responses from 3600 seconds (60 minutes) to 120 seconds (2 minutes)	Overriding caching defaults based on a backend response
Users request a popular page on your site but it's been moved to a different area	Have Fastly redirect the page requests at the edge, without having to go back to your origin server for it	Redirects

Types of conditions and when you can use them

We group conditions into three types:

- request conditions
- response conditions
- cache conditions

A condition's type dictates which configuration objects it can be applied to during a specific stage of the caching process. In addition, each stage of caching works with a different set of VCL variables that can be used to create conditions.

Condition type	Applied when Fastly ...	Works with which VCL variables

Request	processes a request	`client.*`
		`server.*`
		`req.*`

Response	processes a response to a request	`client.*`
		`server.*`
		`req.*`
		`resp.*`

Cache	receives a response from your origin, just before that response is (potentially) cached	`client.*`
		`server.*`
		`req.*`
		`beresp.*`

Where to go for more information

The Varnish Cache documentation provides a complete list of variables you can use to craft conditions. Keep in mind, however, some of the variables Varnish allows may not be available or may have no meaning in the context of Fastly services. For more information, see our Guide to VCL.

* * *

Troubleshooting conditions

🔗 /en/guides/troubleshooting-conditions

If you are having problems using conditions, here are some common things to look for.

Check the Apply if field for if statements

Most problems with conditions occur in the Apply if parameter because it uses logical expressions to represent actual VCL variables that specify when a condition should be applied to a configuration object. If you are having problems using conditions, start by checking to see if you've put an `if ()` statement in the wrong place. A condition's if statement is implied and doesn't need to be placed in the Apply if field of the condition window. You only need to enter an evaluated expression (e.g., `req.url ~ "^/special/"`).

Check the construction of inverse regex matches

Consider if inverse regular expression (regex) matching might be the issue, especially if you're using it to exclude a particular URL in your condition. When using the `!~` (inverse regex match) to build expressions that exclude particular URLs, be thoughtful when also using the `||` or `&&` operators and multiple patterns.

For example, if you want to apply something to all URLs except those that start with `/admin`, the condition you'd enter into the Apply if field would be `req.url !~ "^/admin"`. If you also wanted to exclude URLs starting with `/internal`, that expression would be `!(req.url ~ "^/admin" || req.url ~ "^/internal")`.

✓ TIP

Keep in mind De Morgan's laws if you're using multiple conditions and negation.

Check for general regex formatting mistakes

Consider the following general regex issues that may have caused trouble:

- **Is case sensitivity the problem?** Varnish regex is case sensitive by default. To use a case insensitive check, you must use the `(?i)` flag.
- **Have you escaped forward slashes?** Forward slashes don't need to be escaped in Varnish regex.

Our cheatsheet provides additional examples of using VCL with regular expressions.

* * *

Using conditions

🔗 /en/guides/using-conditions

Conditions use the Varnish Configuration Language (VCL) to define when a configuration object should be applied while processing requests to a cache server. Once you understand some basics about conditions, use this guide to learn about how to create conditions using the Fastly web interface and when to use them.

Where to find conditions

Conditions appear in two areas of Fastly's web interface:

- The Manage conditions page lists all conditions available to your configuration settings.
- Each configuration object displays conditions specifically attached to them.

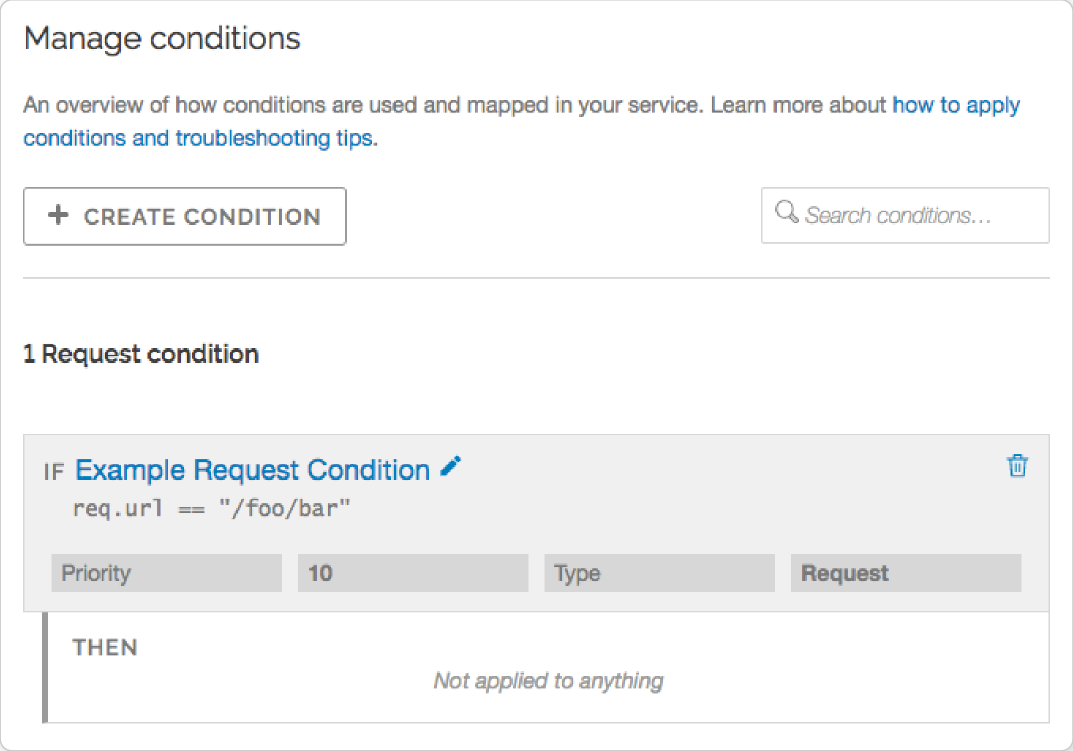
Conditions on the Manage conditions page

The Manage conditions page provides an overview of all the conditions currently available to your service. You can see at a glance which conditions are mapped to configuration objects. It allows you to create new conditions and search for existing ones.

To view conditions on the Manage conditions page:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Conditions**. A list of all conditions for your service appears.

For example, this service has one request condition available:



Manage conditions

An overview of how conditions are used and mapped in your service. Learn more about how to apply conditions and troubleshooting tips.

1 Request condition

IF Example Request Condition

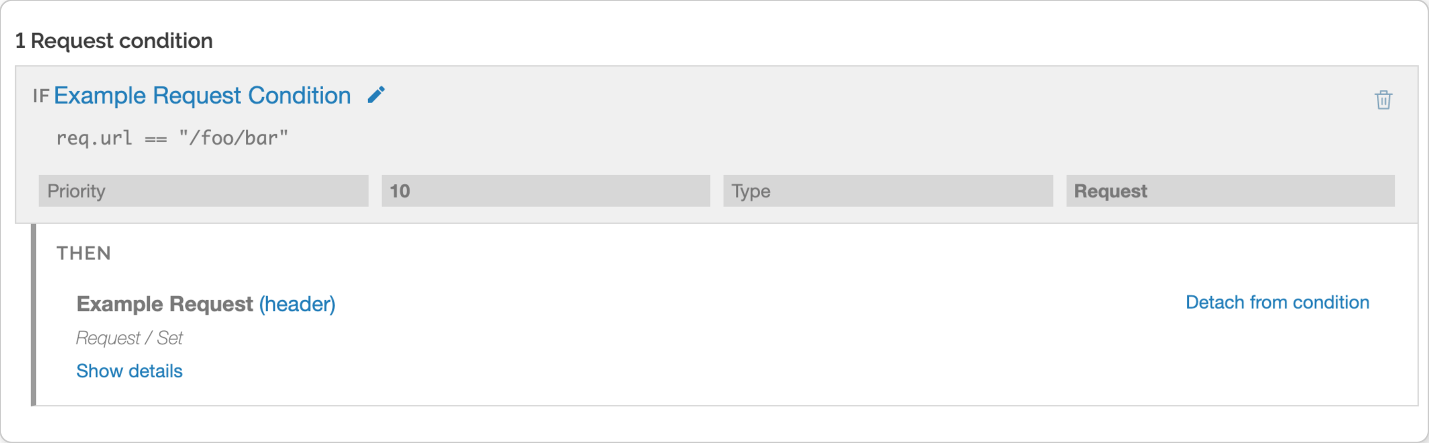
```
req.url == "/foo/bar"
```

Priority	10	Type	Request

THEN

Not applied to anything

The Example Request Condition shown above currently isn't applied to a configuration object (as indicated by "Not applied to anything"). If it was, it would instead appear similar to this:



1 Request condition

IF Example Request Condition

`req.url == "/foo/bar"`

Priority	10	Type	Request

THEN

Example Request (header)

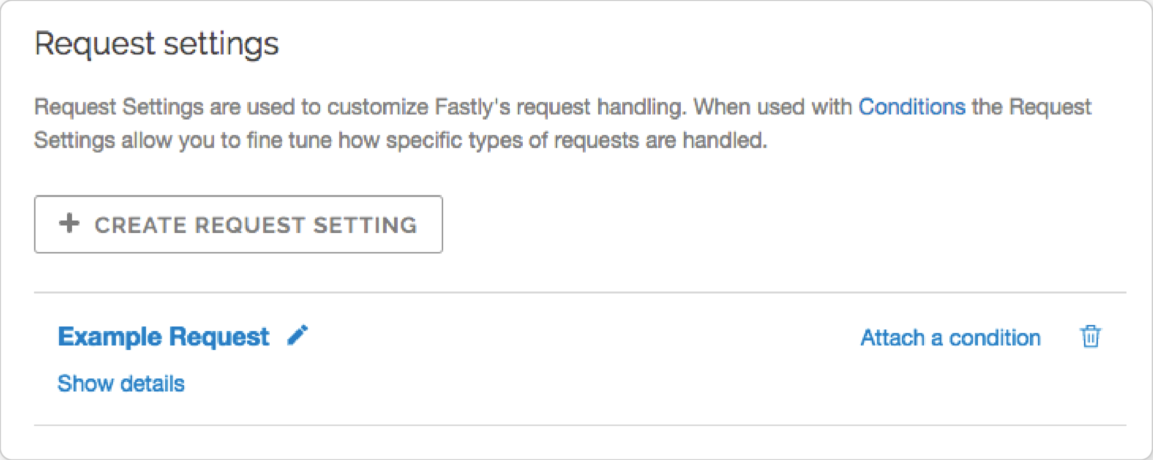
Request / Set

Show details

Detach from condition

Conditions attached directly to a configuration object

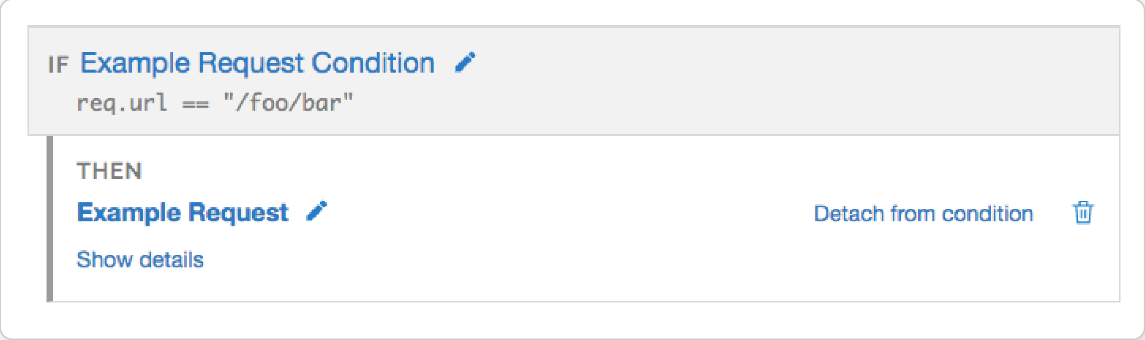
Configuration objects appear differently in the web interface when conditions are attached to them. For example, this request setting has no condition attached to it:



Request settings

Request Settings are used to customize Fastly's request handling. When used with Conditions the Request Settings allow you to fine tune how specific types of requests are handled.

Once you click the Attach a condition link to create a new condition or attach an existing condition, however, the web interface changes how the configuration object appears:



IF Example Request Condition

`req.url == "/foo/bar"`

By default, configuration objects hide the majority of details for any attached conditions. You can unhide those details by clicking the Show details link. When expanded, the details vary depending on the type of condition.

Parts of a properly configured condition

Conditions require only a few parameters, making them appear deceptively simple. Specifically, they require:

- a **Type** parameter that classifies the condition being added. If added via the Manage conditions page, the type can always be manually selected. If added via the Attach a condition link on a configuration object, the type is automatically applied whenever possible.
- a **Name** parameter that serves human-readable identifier of the condition.
- an **Apply if** statement containing the logical expression to execute in VCL to determine if condition resolves as True or False.

Most problems with conditions occur in the Apply if parameter.

Performing matches on basic logical expressions

Properly configured conditions can perform matches on complicated logical expressions specified in the Apply if parameter. For example:

This logical expression ...	Matches when ...
`client.ip == "127.0.0.1"`	The client requesting a resource on your service has the IP `127.0.0.1`.
`req.http.host == "example.com"`	The Host header of the incoming request is `example.com`.
`req.method == "POST" && req.url ~ "^/api/articles/"`	The request is a POST and the URL begins with `/api/articles/`.

The `client.ip`, `req.http.host`, `req.method`, and `req.url` conditions shown above all represent configuration variables in VCL.

Using operators to perform matches on complex logical expressions

You could also get creative and create a more complex condition used by Fastly that might have an Apply if parameter that looked like this:

```
req.http.host == "www.example.com" && (req.url !~ "/foo" && req.url !~ "/bar" && req.url
```

This condition tells the cache server that the Host should equal `www.example.com` and the URL cannot match `/foo` or `/bar` or `/baz`. You might use this type of condition when you have multiple variables or options and want to fine-tune your results. In this example, you are indicating that you don't want URLs that contain `foo`, `bar`, or `baz` by using the following operators:

This operator ...	Does this ...
`( )`	groups expressions and restricts alteration to part of the regex
`&&`	ensures each equation is true
`!~`	excludes any URLs that include the specified variables

An example of adding conditions

The scenario: You want to add a new origin server that handles a specific portion of your API requests. Some requests to this API must be cached differently than other requests to your API, so you want to set special headers for specific types of requests. Specifically, you don't want your new origin server to cache PUT, POST, or DELETE requests because they're special for this particular API and send back extra, time dependent, meta-information in each response. And finally, you want to track the effectiveness of doing this. To accomplish all of this using conditions via the Fastly web interface, you would:

1. Create a new origin server to handle the special API traffic.
2. Create a new condition that tells the cache how to route some of the API requests to that origin server.
3. Create a new cache setting object to ensure the origin server caches only the correct responses.
4. Create a new condition that specifies when the cache settings object should be applied.
5. Create a new header to track the specific type of API requests.
6. Create a new response condition to make sure that the header is only set on specific type of request.
7. Check your work.

Create a new origin server

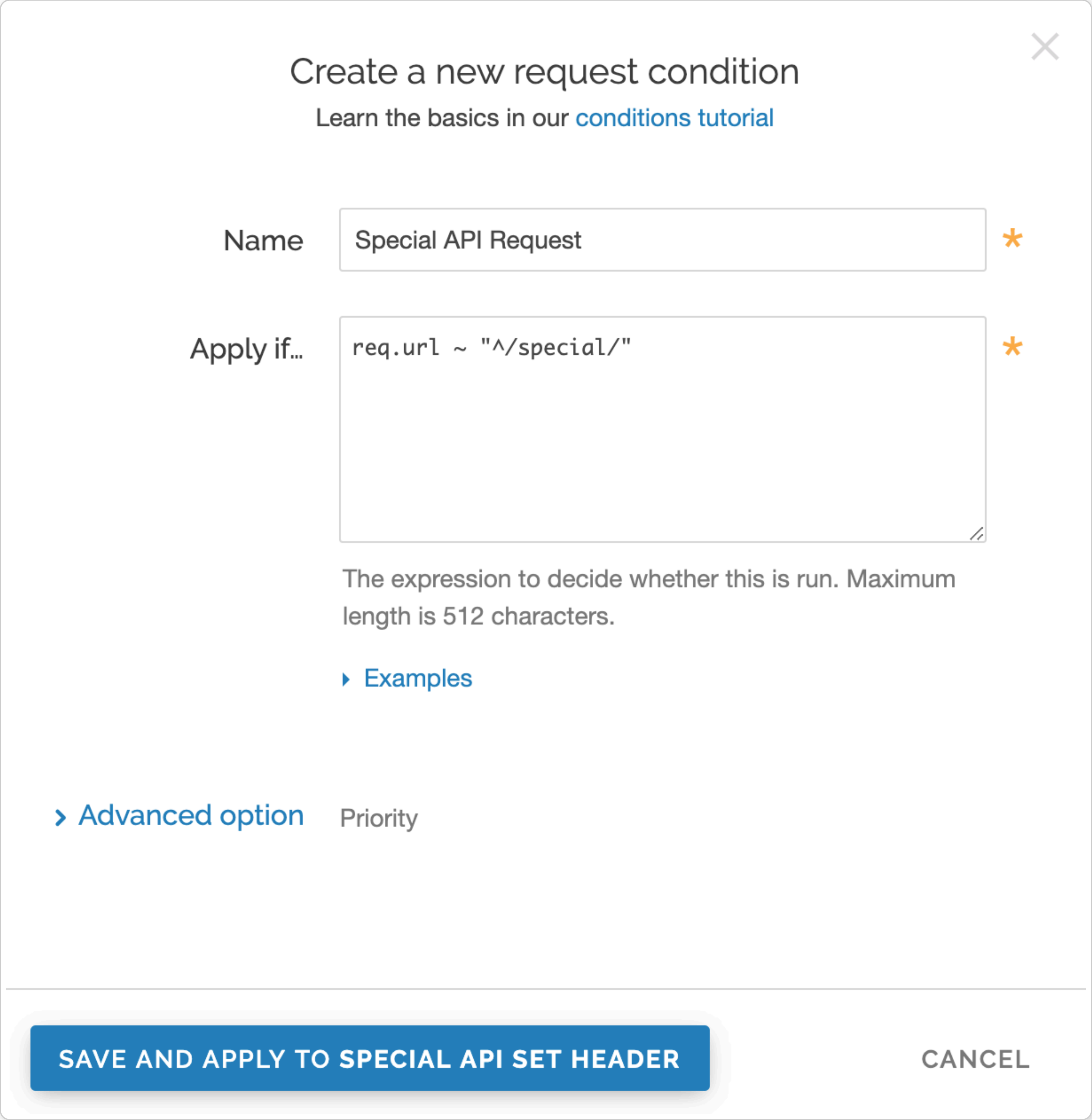
To create a new origin server that will handle the special API traffic, follow the instructions for working with hosts. You'll add specific details about your API server when you fill out the **Create a host** fields:

- In the **Name** field, enter a name for your API server (for example, `Special API Server`).
- In the **Address** field, enter the IP address (or hostname) of the API server.

Create a request condition

Once you've created a new origin server to handle the special API traffic, tell the cache how to route requests to this origin server by creating a request condition.

1. In the **Hosts** area, click **Attach a condition** next to the name of the origin server you just created.
2. You can either select an available condition or you can click **Create a new request condition**.



Create a new request condition

Learn the basics in our conditions tutorial

Name: Special API Request

Apply if...

```
req.url ~ "^/special/"
```

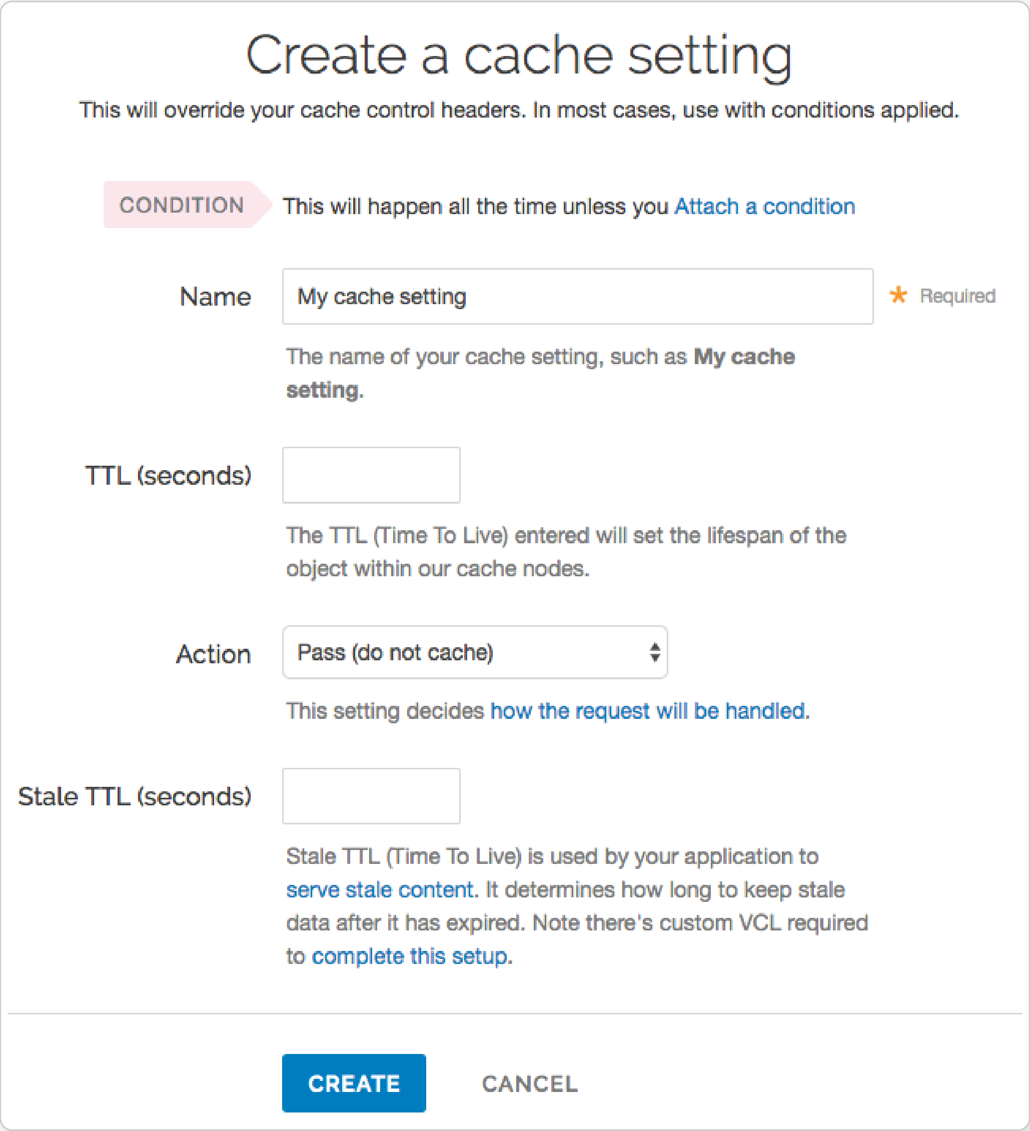
The expression to decide whether this is run. Maximum length is 512 characters.

3. Fill out the **Create a new request condition** fields as follows:
    - In the **Name** field, enter a descriptive name for the new condition (for example `Special API Request`).
    - In the **Apply if** field, enter the appropriate request condition that will be applied (for example, `req.url ~ "^/special/"` could address all requests related to the special API server).
4. Click **Save and apply to**. The new condition for the host is created.

Create a cache settings object

Requests are now are being properly routed to the new origin server. Next, create a cache settings object to ensure Fastly doesn't cache any responses from PUT, POST, or DELETE requests. They're special for this particular API and send back extra, time dependent, meta-information in each response.

1. Click **Settings**.
2. In the **Cache Settings** area, click the **Create cache setting**.



Create a cache setting

This will override your cache control headers. In most cases, use with conditions applied.

CONDITION This will happen all the time unless you Attach a condition

Name: My cache setting ★ Required

The name of your cache setting, such as **My cache setting**.

TTL (seconds):

The TTL (Time To Live) entered will set the lifespan of the object within our cache nodes.

Action: Pass (do not cache)

This setting decides how the request will be handled.

Stale TTL (seconds):

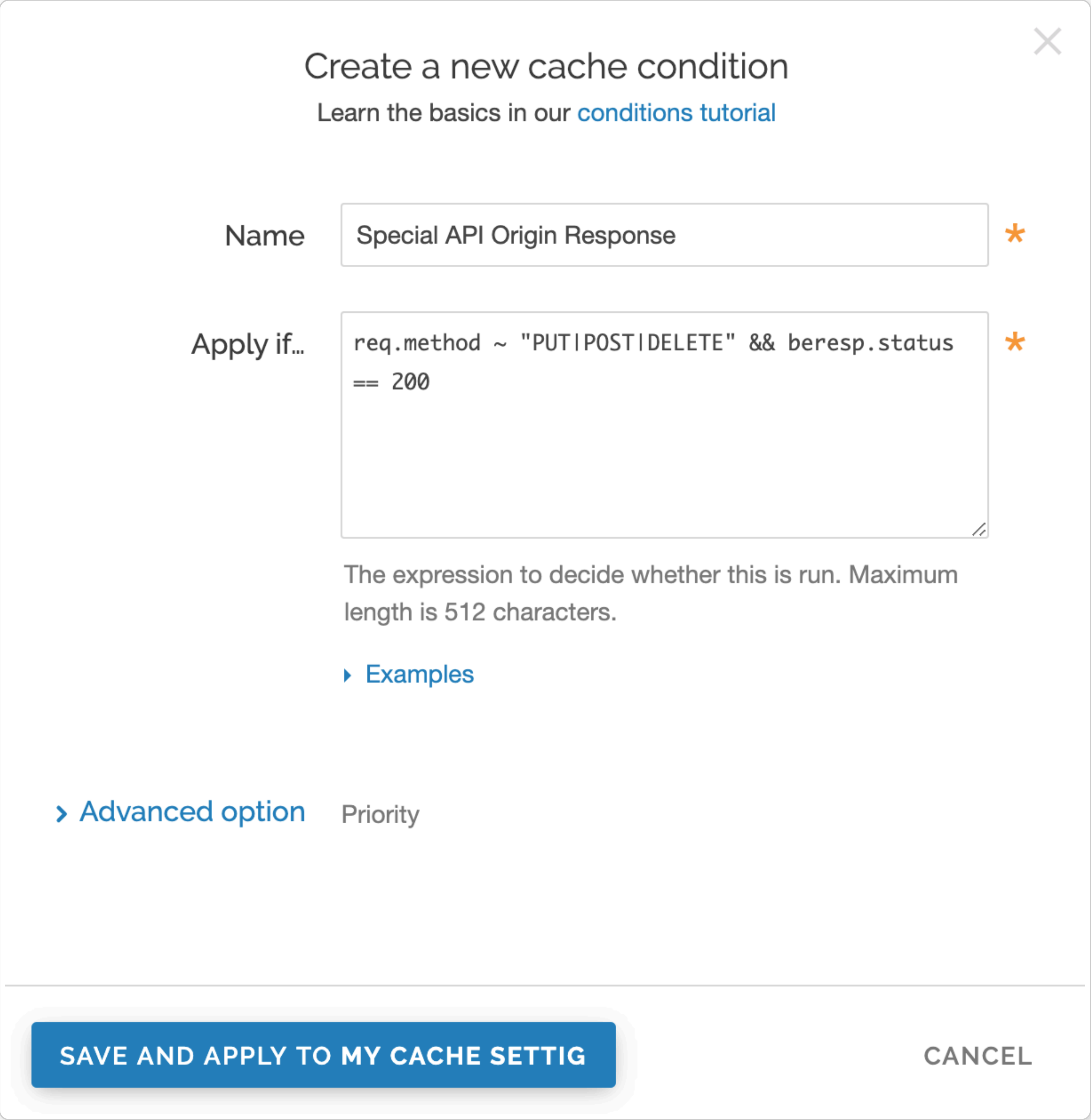
Stale TTL (Time To Live) is used by your application to serve stale content. It determines how long to keep stale data after it has expired. Note there's custom VCL required to complete this setup.

3. Fill out the **Create a cache setting** fields as follow:
    - In the **Name** field, enter a descriptive name for the new cache settings.
    - Leave the **TTL (seconds)** field set to its default value.
    - From the **Action** menu, select **Pass (do not cache)**.
    - Leave the **Stale TTL (seconds)** field set to its default value.
4. Click **Create**.

Create and apply a condition to the cache settings object

Create a new condition that specifies when the cache settings object should be applied.

1. In the **Cache Settings** area, click the **Attach a condition** link next to the name of the cache setting you just created.
2. Click **Create a new cache condition**.



Create a new cache condition

Learn the basics in our conditions tutorial

Name: Special API Origin Response

Apply if...:

```
req.method ~ "PUT|POST|DELETE" && beresp.status
== 200
```

The expression to decide whether this is run. Maximum length is 512 characters.

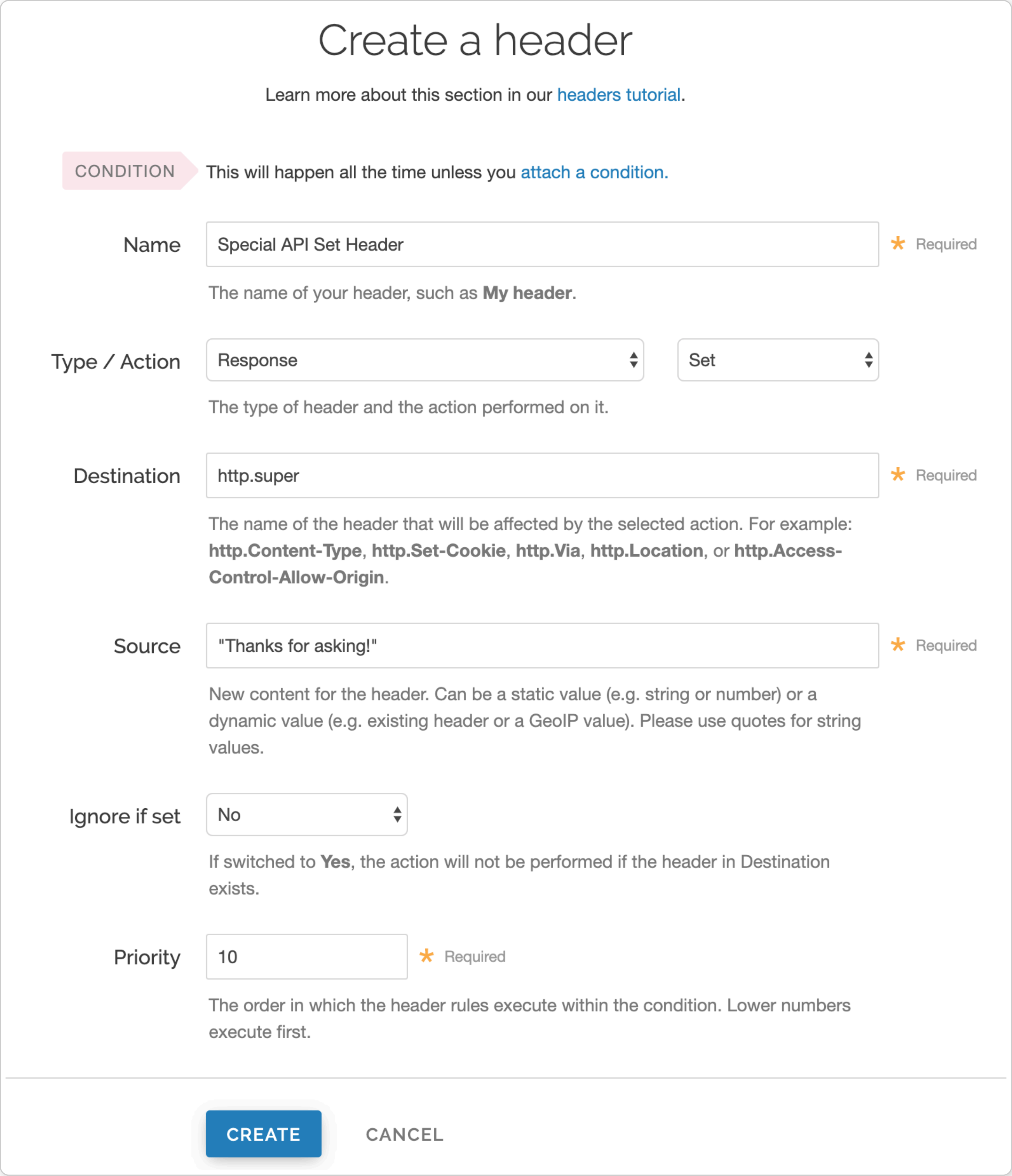
3. Fill out the **Create a new cache condition** fields as follows:
   - In the **Name** field, enter a descriptive name for the new condition (for example, `Special API Origin Response`).
   - In the **Apply if** field, enter the appropriate request condition that will be applied (for example, `req.method ~ "PUT|POST|DELETE" && beresp.status == 200`).
4. Click **Save and apply to**. The new condition for the cache setting is created.

Create a new header

To make sure you can track the effectiveness the new API, create a new header so you can use it to gather information about the special API requests as they happen.

1. Click **Content**.

2. In the **Headers** area, click the **Create header** button to create a new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Special API Set Header ★ Required

The name of your header, such as **My header**.

Type / Action: Response / Set

The type of header and the action performed on it.

Destination: http.super ★ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "Thanks for asking!" ★ Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 ★ Required

The order in which the header rules execute within the condition. Lower numbers execute first.

3. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter a descriptive name for the new header (for example, `Special API Set Header`).
   - From the **Type** menu, select **Response** and from the **Action** menu, select **Set**.

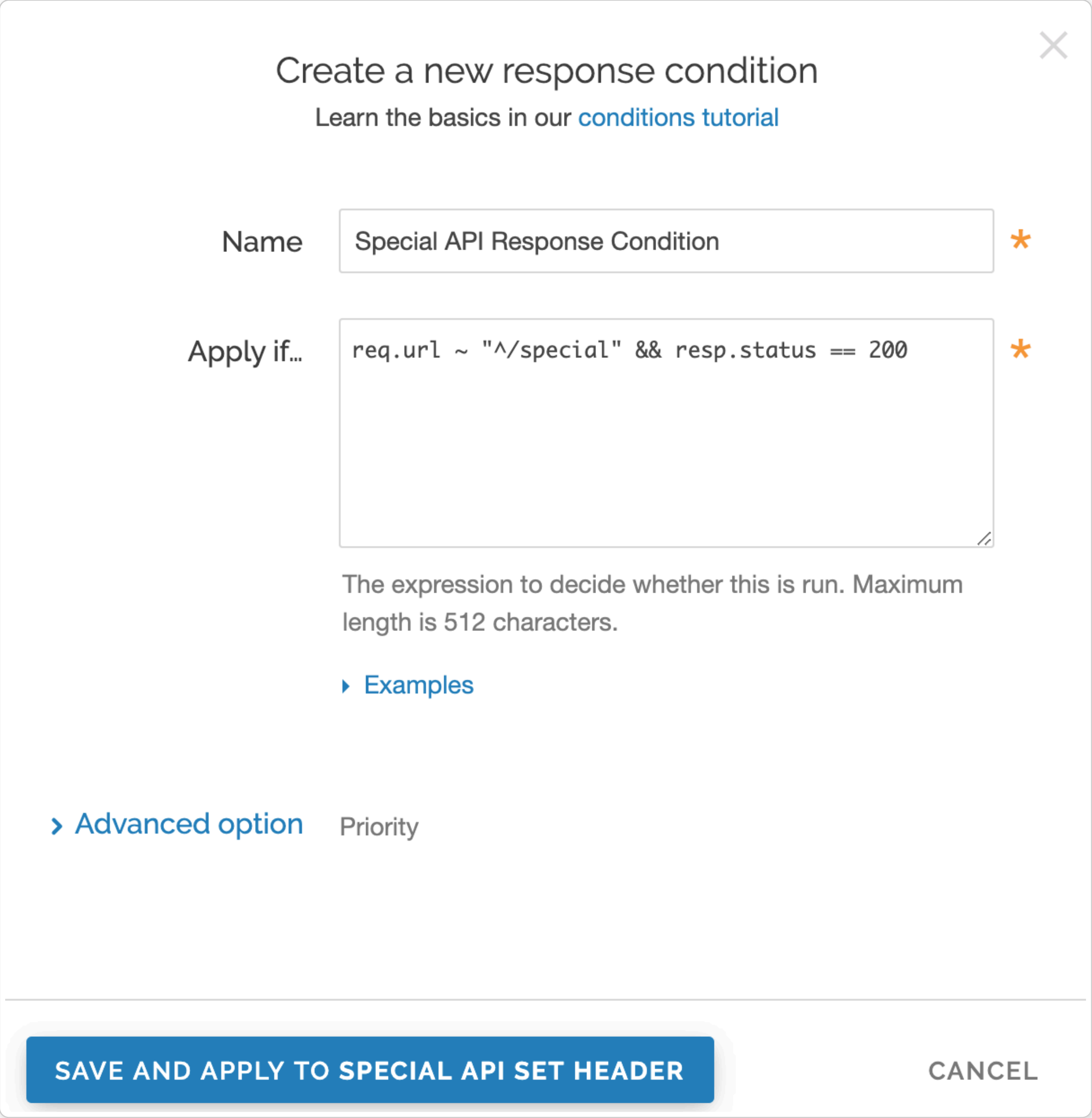
- In the **Destination** field, enter the name of the header that will be affected by the action (for example, `http.super`).
- In the **Source** field, enter a description of the source where the content for this header comes from (for example, `"Thanks for asking!"`).
- Leave the **Ignore if set** and **Priority** fields set to their default settings.

4. Click **Create**.

Create a response condition for the new header

Once the header is created, create an associated condition to ensure this header is only set on that special type of request.

1. In the **Headers** area, click **Attach a condition** next to the name of the new header you just created.



Create a new response condition

Learn the basics in our conditions tutorial

Name: Special API Response Condition

Apply if...

```
req.url ~ "^/special" && resp.status == 200
```

The expression to decide whether this is run. Maximum length is 512 characters.

2. Fill out the **Create a new response condition** fields as follows:
    - In the **Name** field, enter a descriptive name for the new condition (for example, `Special API Response Condition`).
    - In the **Apply if** field, enter the appropriate request condition that will be applied (for example, `req.url ~ "^/special" && resp.status == 200`).
3. Click **Save and apply to**. The new condition for the header is created.

Check your work

Before activating the configuration, review the generated VCL to see how Fastly converted the objects and conditions into actual VCL. For the example shown above, the VCL for the request condition appears as:

```
# Condition: Special API Request Prio: 10
if (req.url ~ "^/special/") {
  set req.backend = F_Special_API_Server;
}
#end condition
```

The cache settings and condition VCL appears as:

```
if (req.method ~ "PUT|POST|DELETE" && beresp.status == 200) {
  set beresp.ttl = 0s;
  set beresp.grace = 0s;
  return(pass);
}
```

And the new header response condition VCL appears as:

```
# Condition Special API Response Condition Prio: 10
if (req.url ~ "^/special" && resp.status == 200) {

  # Header rewrite Special API Set Header: 10
  set resp.http.super = "Thanks for asking!";
}
```

As you become more familiar with the VCL syntax and programming, look at the generated VCL to see if the configuration is doing what you think it is doing (most VCL is pretty simple once you know what the variables are referring to).

* * *

Category: Custom VCL

These articles describe how to create your own VCL files with specialized configurations.

About VCL Snippets

/en/guides/about-vcl-snippets

VCL Snippets are short blocks of VCL logic that can be included directly in your service configurations. They're ideal for adding small sections of code when you don't need more complex, specialized configurations that sometimes require custom VCL. Fastly supports two types of VCL Snippets:

- **Regular VCL Snippets** get created as you create versions of your Fastly configurations. They belong to a specific service and any modifications you make to the snippet are locked and deployed when you deploy a new version of that service. You can treat regular snippets like any other Fastly objects because we continue to clone them and deploy them with a service until you specifically delete them. You can create regular snippets using either the web interface or via the API.

- **Dynamic VCL Snippets** can be modified and deployed any time they're changed. Because they are versionless objects (much like dictionaries or ACLs at the edge), dynamic snippets can be modified independently from service changes. This means you can modify snippet code rapidly without deploying a service version that may not be ready for production. You can only create dynamic snippets via the API.

Limitations of VCL Snippets

- Snippets are limited to 1MB in size by default. If you need to store snippets larger than the limit, contact support.
- Snippets don't currently support conditions created through the web interface. You can, however, use `if` statements in snippet code.
- Snippets cannot currently be shared between services.
- Snippets cannot be nested (e.g., you can't include a snippet into another snippet).

* * *

Basic authentication

/en/guides/basic-authentication

Basic authentication is a simple way of protecting a website at the edge. Users enter a username and password combination to access pages protected by basic authentication. You can use basic authentication to restrict access to low-risk assets like testing and staging environments.

WARNING

Basic authentication shouldn't be used to restrict access to sensitive information. See the security considerations section for more information.

Implementing basic authentication

Follow our HTTP basic auth example to implement basic authentication using custom VCL or Compute.

Using basic authentication with GCS

To use basic authentication with Google Cloud Storage (GCS) as a origin server, add a request header to delete the `http.Authorization` header and prevent it from being sent to GCS. That header causes GCS to respond with a "Not Authorized" message instead of your request.

Security considerations

There are several security considerations you should take into account before using basic authentication:

- Basic authentication can't protect high-risk information. Don't use it to restrict access to sensitive information.

- If you're not using TLS, the password will be transmitted over the wire in Base64 encoding. The encoded string could easily be captured using an application like Wireshark and converted to plaintext.
- The password is cached by the user's web browser, and it can be permanently saved by the user's web browser.

Using access control lists

As an alternative to basic authentication, you can use access control lists (ACLs) to restrict access to your assets by allowlisting a set of IP addresses. To allowlist IP addresses with an ACL, add custom VCL to Fastly's boilerplate VCL.

```
# Who is allowed access ...
acl local {
    "localhost";
    "192.168.1.0"/24; /* and everyone on the local network */
    ! "192.168.1.23"; /* except for the dial-in router */
}
```

See our ACL guides for more information.

* * *

Custom responses that don't hit origin servers

/en/guides/custom-responses-that-dont-hit-origin-servers

Fastly can send custom responses for certain requests that you don't want to hit your origin servers.

Creating a quick response

Fastly provides features that allow you to quickly enable and configure responses for a robots.txt file and 404 and 503 errors. For more information, check out our guides on creating and customizing a robots.txt file and creating error pages with custom responses.

Creating an advanced response

You can create an advanced response to specify the HTTP status code, MIME type, and content of the response. For example, if you wanted to restrict caching to a URL subtree that contains images and scripts, you could configure Fastly to return an `HTTP 404 Not Found` response to requests for anything other than `/Content/*` or `/Scripts/*`. To illustrate how to implement this example, we'll show you how to create a response and corresponding request condition.

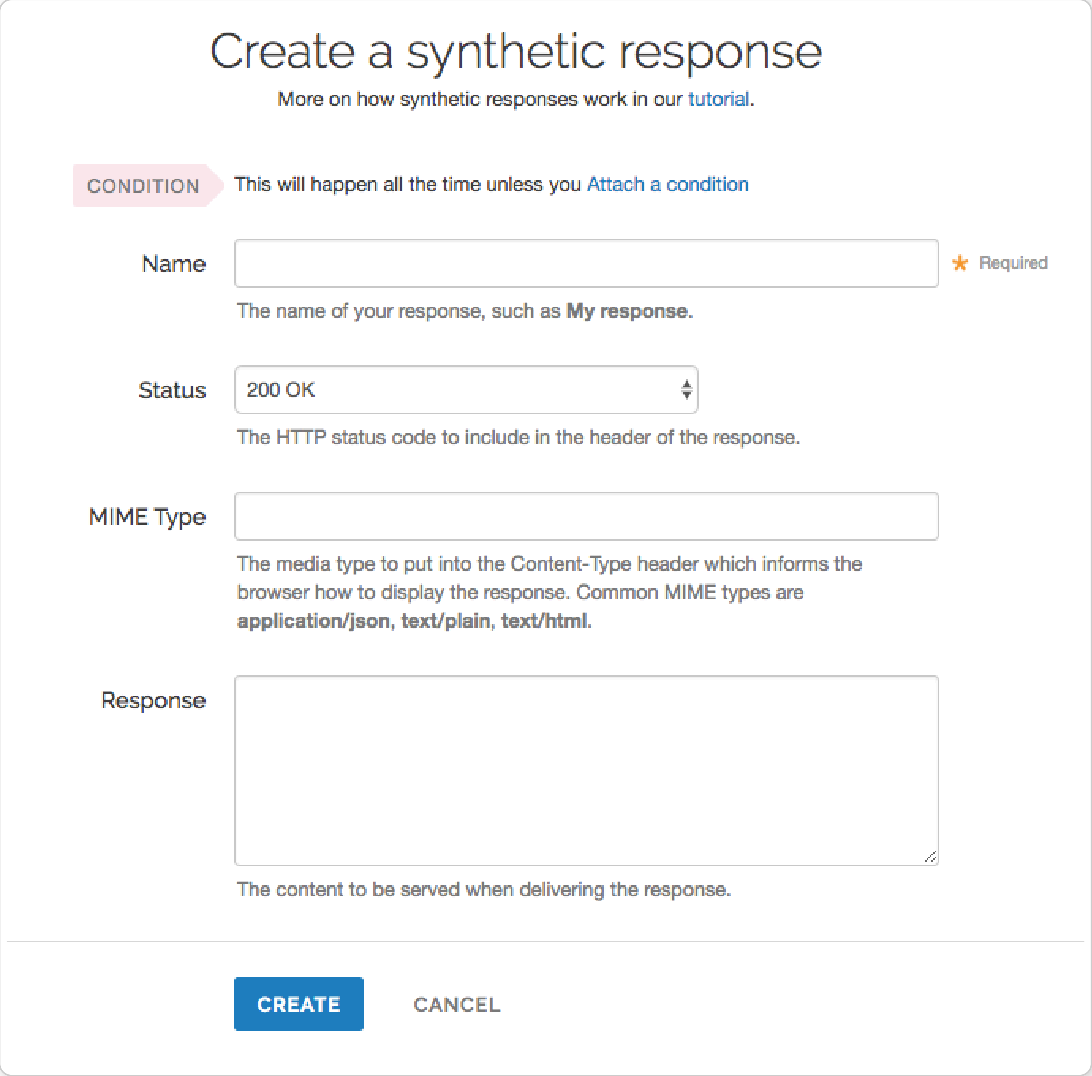
Follow these instructions to create an advance response:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Content**.

5. Click **Set up advanced response**.



Create a synthetic response

More on how synthetic responses work in our tutorial.

CONDITION This will happen all the time unless you Attach a condition

Name ★ Required

The name of your response, such as **My response**.

Status 200 OK

The HTTP status code to include in the header of the response.

MIME Type

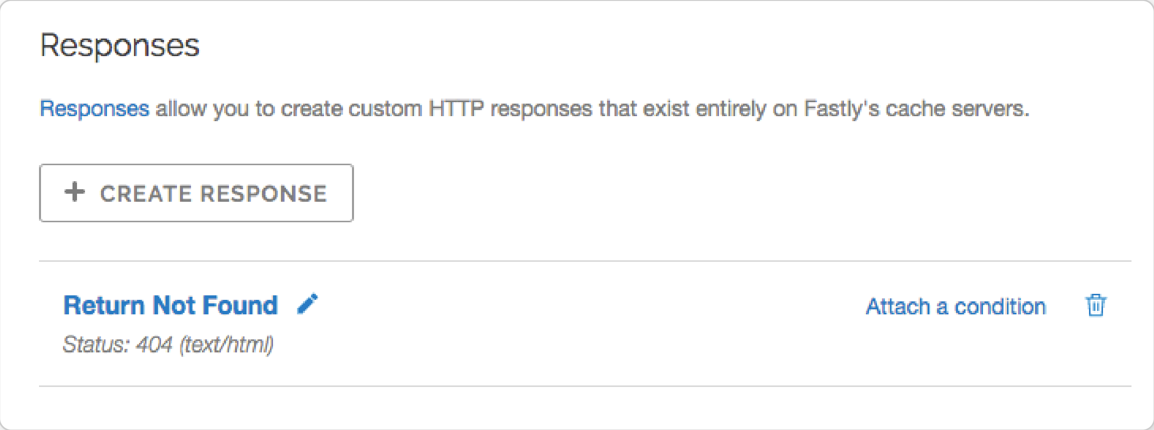
The media type to put into the Content-Type header which informs the browser how to display the response. Common MIME types are **application/json**, **text/plain**, **text/html**.

Response

The content to be served when delivering the response.

6. Fill out the **Create a synthetic response** fields as follows:
    - In the **Name** field, enter a human-readable name for the response. For example `Return Not Found`.
    - From the **Status** menu, select an HTTP code to return to the client. For example, `404 Not Found`.
    - In the **MIME Type** field, enter the MIME type of the response. For example, `text/html`.
    - In the **Response** field, enter the plaintext or HTML content to return to the client. For example `Page not found`.

7. Click **Create** to create the response. The new response appears in the Responses area of the Content page.



Responses

Responses allow you to create custom HTTP responses that exist entirely on Fastly's cache servers.

+ CREATE RESPONSE

Return Not Found

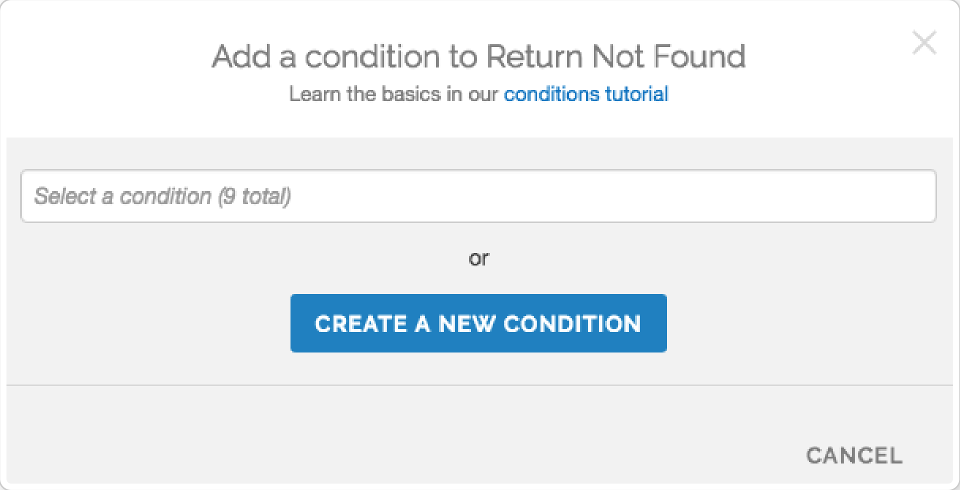
Status: 404 (text/html)

Attach a condition

Creating the request condition

Follow these instructions to attach a request condition to the response you just created:

1. Click the **Attach a condition** link next to the response that you just created.

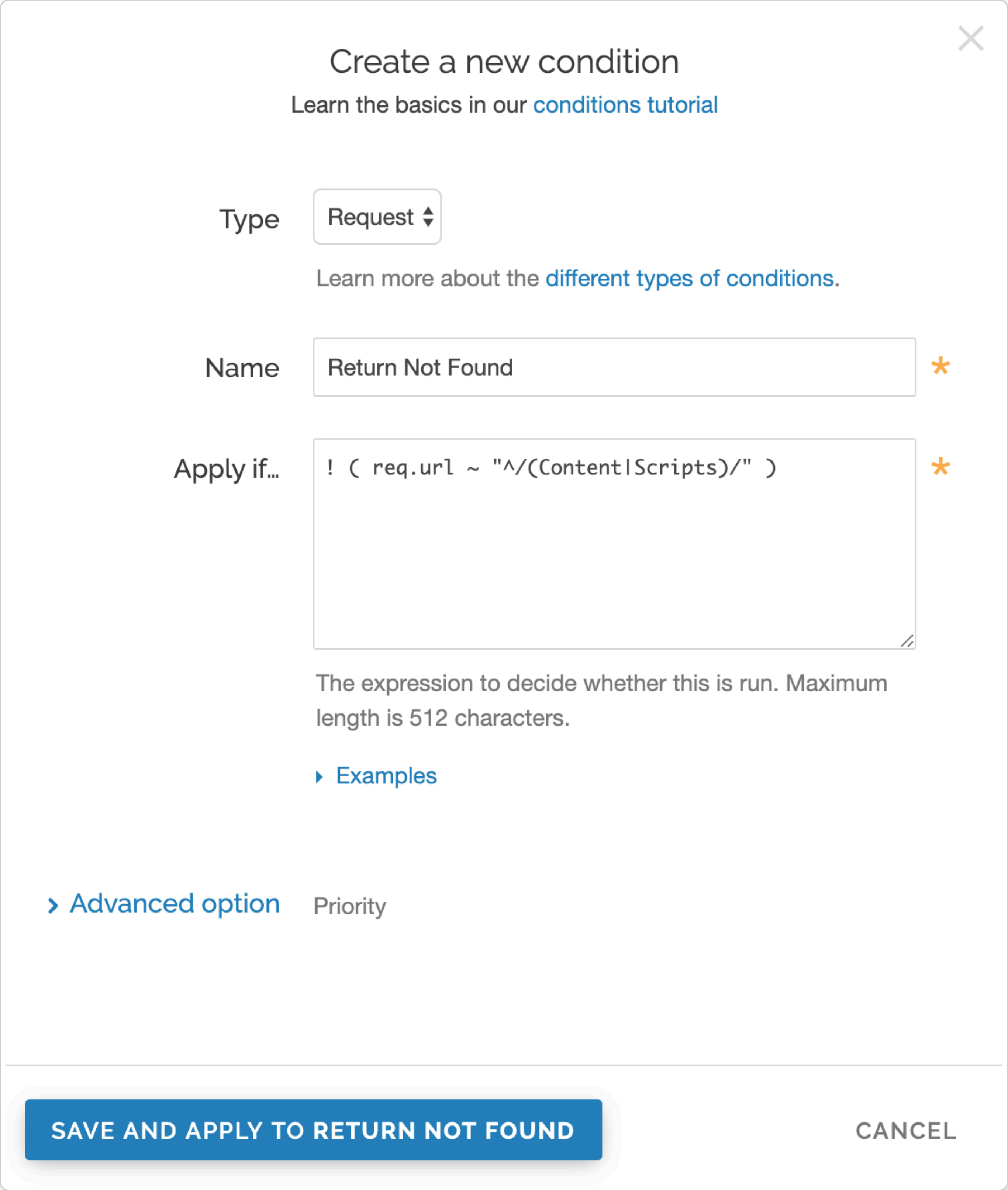


Add a condition to Return Not Found

Learn the basics in our conditions tutorial

Select a condition (9 total)
or
CREATE A NEW CONDITION
CANCEL

2. Click **Create a new condition**.



Create a new condition

Learn the basics in our conditions tutorial

Type: Request

Learn more about the different types of conditions.

Name: Return Not Found

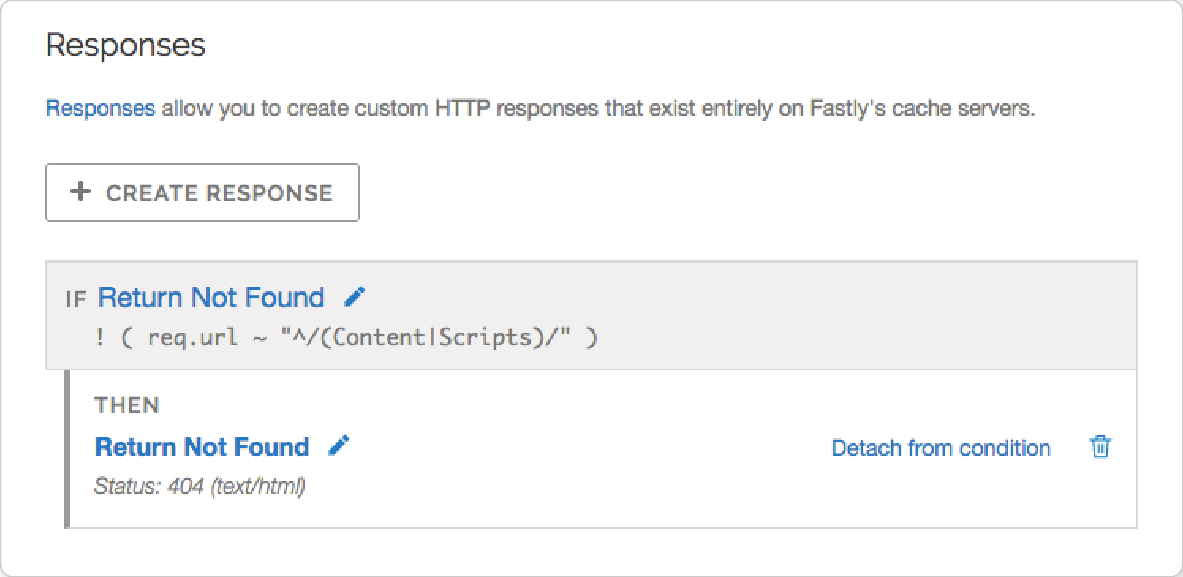
Apply if…

```
! ( req.url ~ "^/(Content|Scripts)/" )
```

The expression to decide whether this is run. Maximum length is 512 characters.

3. Fill out the **Create a new condition** fields as follows:
   - From the **Type** menu, select the type of condition you want to create.
   - In the **Name** field, enter a human-readable name for the condition. For example, `Return Not Found`.
   - In the **Apply if** field, enter the request condition you want inserted into a VCL if statement. For example, `! ( req.url ~ "^/(Content|Scripts)/" )`. See below for more examples of request conditions.

4. Click **Save and apply to**. The Responses area now displays the condition that must be met in order for your response to begin being used.



Responses

Responses allow you to create custom HTTP responses that exist entirely on Fastly's cache servers.

```
! ( req.url ~ "^/(Content|Scripts)/" )
```

Status: 404 (text/html)

5. Click **Activate** to deploy your configuration changes.

Example request conditions

Respond only if URLs don't match a certain mask, in this case `/Content/*` or `/Scripts/*`:

```
! (req.url ~ "^/(Content|Scripts)/")
```

Respond only if URLs match `/secret/*` or are Microsoft Word or Excel documents (`*.doc` and `*.xls` file extensions):

```
! (req.url ~ "^/secret/" || req.url ~ "\.(xls|doc)$")
```

Ignore POST and PUT HTTP requests:

```
req.method == "POST" || req.method == "PUT"
```

Deny a spider or crawler using `user-agent` `"annoying_robot"`:

```
req.http.user-agent ~ "annoying_robot"
```

Prevent a specific IP from connecting, in this case the IP `225.0.0.1`:

```
client.ip == "225.0.0.1"
```

Use geographic variables to block traffic from a specific location (e.g., China):

```
client.geo.country_code == "CN"
```

Match the `client.ip` against a CIDR range, such as `240.24.0.0/16` (this requires first creating an ACL object in VCL):

```
client.ip ~ ipRangeObject
```

---

* * *

Developer guide: Best practices for VCL

/en/guides/developer-guide-best-practices-for-vcl

---

* * *

Developer guide: Clustering with VCL

/en/guides/developer-guide-clustering-with-vcl

---

* * *

Developer guide: Getting started with VCL

/en/guides/developer-guide-getting-started-with-vcl

---

* * *

Developer guide: Synthetic errors in VCL

/en/guides/developer-guide-synthetic-errors-in-vcl

---

* * *

Developer guide: Using VCL

🔗 /en/guides/developer-guide-using-vcl

* * *

Guide to VCL

🔗 /en/guides/guide-to-vcl

Fastly's Edge Cloud services use the Fastly Varnish Configuration Language (VCL), a scripting language used to configure and add logic to Varnish caches. Fastly VCL allows you to generate and compile changes to your Fastly services. These changes can be distributed to all Fastly caches worldwide and then loaded and activated without requiring maintenance windows or service downtime. Fastly VCL is generated automatically per your service configurations specified via the web interface.

VCL and what you can do with it

We allow you to create your own VCL files with specialized configurations. Your custom VCL files can be uploaded into Fastly caches and activated.

You can also mix and match custom VCL and Fastly VCL, using them together at the same time. You will never lose the options on the Fastly web interface when you use custom VCL, but keep in mind that custom VCL always takes precedence over any VCL generated by the web interface. Be mindful of where your custom VCL sits in the default VCL.

> IMPORTANT
>
> Personal data should not be incorporated into VCL. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

Embedding inline C code in VCL

Currently, we don't provide embedded C access to our users. Fastly is a shared infrastructure. By allowing the use of inline C code, we could potentially give a single user the power to read, write to, or write from everything. As a result, our varnish process (i.e., files on disk, memory of the varnish user's processes) would become unprotected because inline C code opens the potential for users to do things like crash servers, steal data, or run a botnet.

We appreciate feedback from our customers. If you are interested in a feature that requires C code, contact support. Our engineering team looks forward to these kinds of challenges.

Where to learn more about VCL and Varnish

Fastly's Developer Hub provides a reference of Fastly VCL for programming custom edge logic on VCL services. You can also discover more about learning to build on the Fastly platform using VCL and the current best practices involved.

The official Varnish documentation is a good place to start when looking for online information. In addition, Varnish Software, who provides commercial support for Varnish, has written a free online book.

* * *

Isolating header values without regular expressions

/en/guides/isolating-header-values-without-regular-expressions

---

Fastly supports the ability to extract header subfield values without regular expressions in a human-readable way. *Headers subfields* are headers with a body syntax style similar to `value1=123value123; testValue=asdf_true; staff_user=true;` or `max-age=0, surrogate-control=3600` These headers include Cookie, Set-Cookie, Cache-Control, or a custom header. Fastly allows you to isolate these key values with the following syntax:

```
req.http.Header-Name:key-name
```

For example, if a `Cookie` request from a client was `value1=123value123; testValue=asdf_true; staff_user=true;`, you could isolate the `staff_user` value using this logic:

```
set req.http.Staff-User = req.http.Cookie:staff_user;
```

The same can be accomplished by using the subfield function:

```
set req.http.Staff-User = subfield(req.http.Cookie, "staff_user", ";");
```

You can add this logic using VCL Snippets or using a custom header.

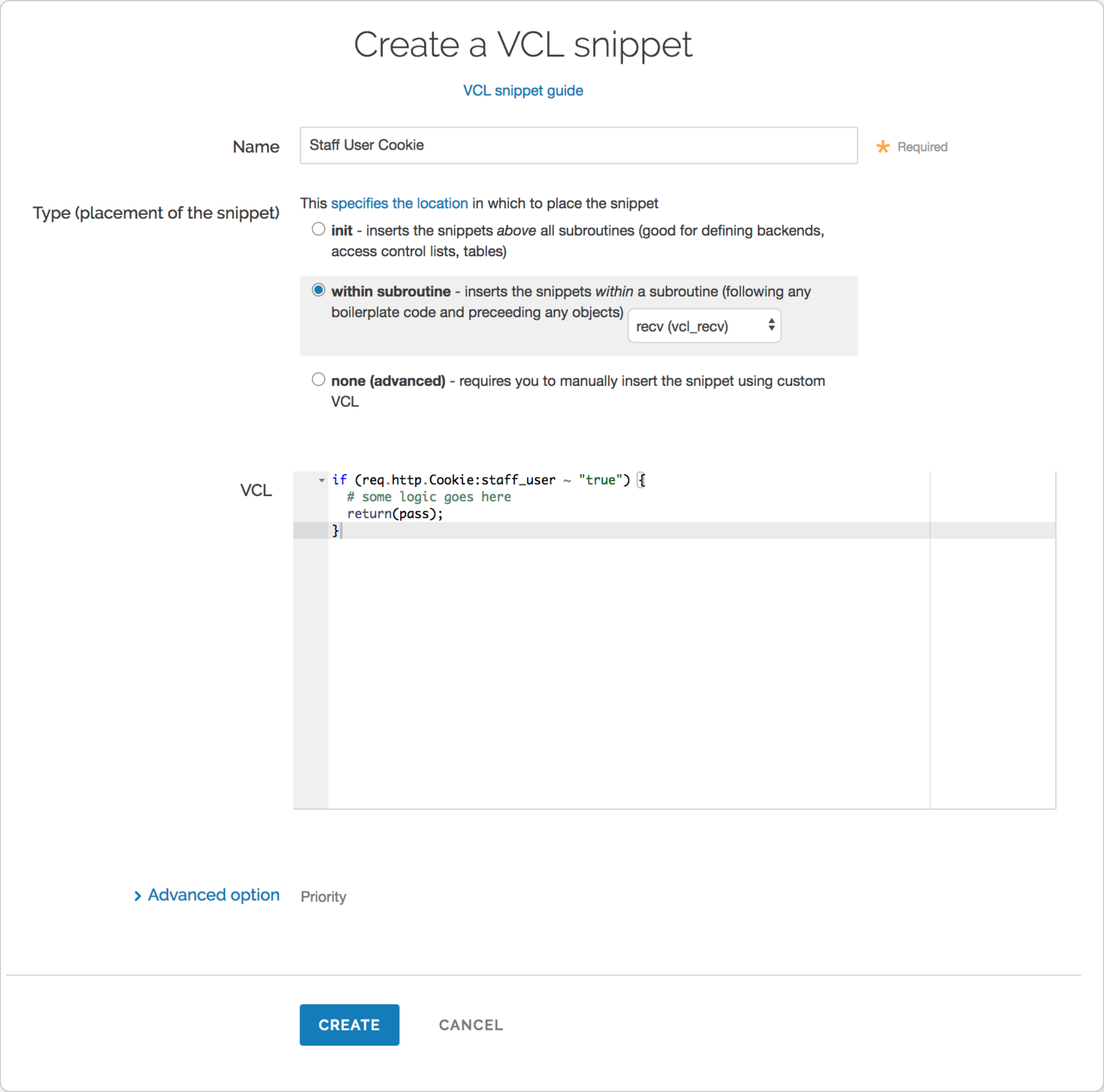
⚠ WARNING

The subfield function as well as the `:` accessor cannot be used for `Set-Cookie` headers.

Using VCL Snippets

To execute this logic based on the value of `staff_user` within `req.http.Cookie` using a VCL Snippet, you would:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create Snippet**.



Create a VCL snippet

Name: Staff User Cookie

Type (placement of the snippet)

This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects) recv (vcl_recv)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL

```
if (req.http.Cookie:staff_user ~ "true") {
  # some logic goes here
  return(pass);
}
```

6. In the **Name** field, enter an appropriate name (e.g., `Staff User Cookie`).

7. From the **Type** controls, select **within subroutine**.

8. From the **Select subroutine** menu, select **recv (**`vcl_recv`**)**.

9. In the **VCL** field, add the following condition:

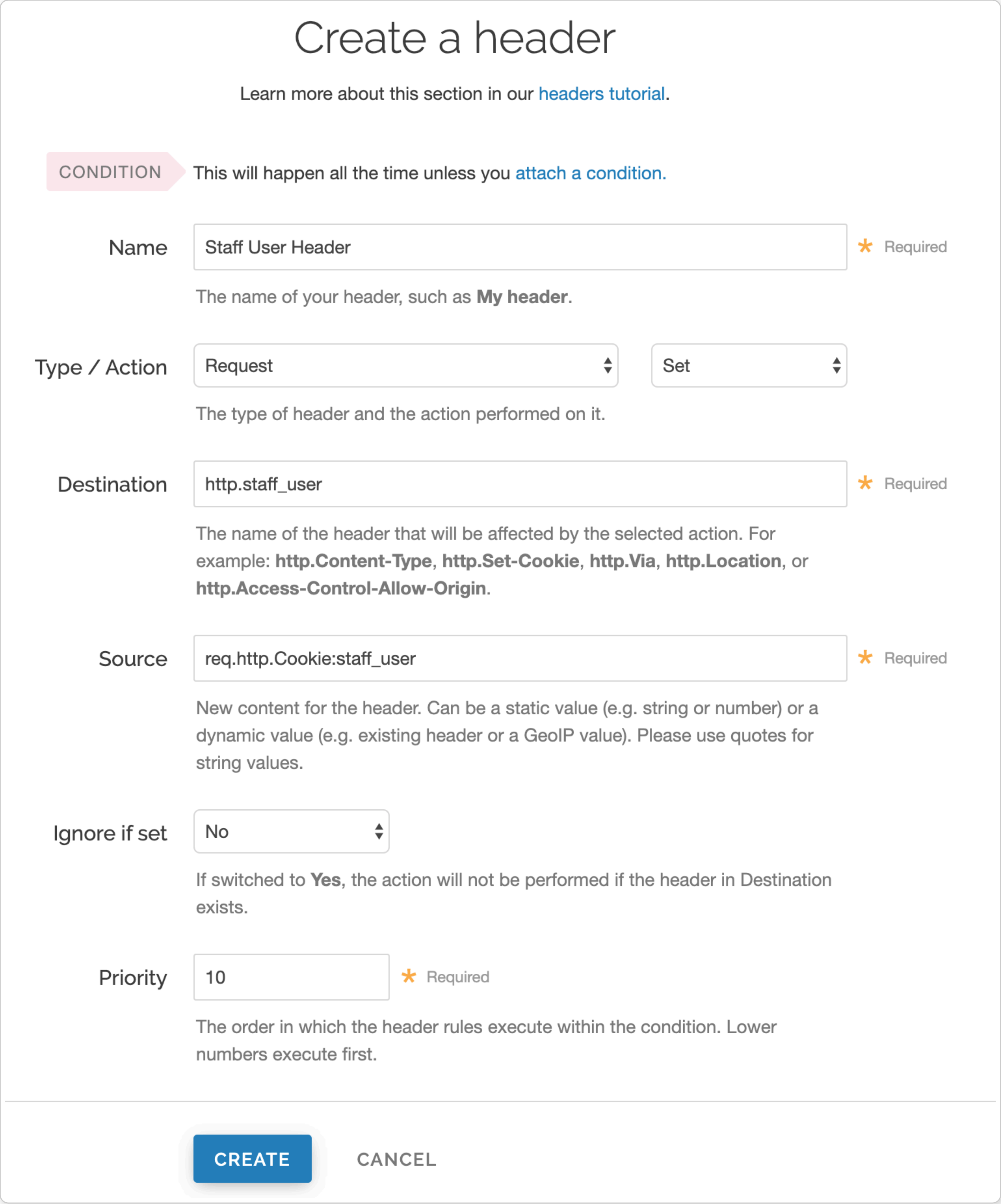
```
# in vcl_recv
if (req.http.Cookie:staff_user ~ "true") {
  # some logic goes here
  return(pass);
}
```

10. Click **Create** to create the snippet.

11. Click **Activate** to deploy your configuration changes.

Using a custom header

You can isolate the value of `staff_user` from `Cookie` to the header `req.http.staff_user` by creating a custom header with the following settings:



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Staff User Header * Required

The name of your header, such as **My header**.

Type / Action: Request | Set

The type of header and the action performed on it.

Destination: http.staff_user * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: req.http.Cookie:staff_user * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

Fill out the **Create a header** fields as follows:

- In the **Name** field, enter `Staff User Header`.
- From the **Type** menu, select **Request**, and from the **Action** menu select **Set**.

- In the **Destination** field, enter `http.staff_user`.
- In the **Source** field, enter `req.http.Cookie:staff_user`.
- From the **Ignore if set** menu, select **No**.
- In the **Priority** field, enter `10`.

This will send the `staff_user` header in every inbound request.

NOTE

You can use the `Attach a condition` link to only create this header when it's needed. See our Using Conditions docs for more information.

* * *

Manipulating the cache key

/en/guides/manipulating-the-cache-key

Before you begin

If your origin uses special values (e.g., request headers) to select content for users or to otherwise direct requests to appropriate security domains, consider including those values in your cache key or Vary header. Doing so will prevent you from accidentally caching content across security domains and could prevent malicious attackers from poisoning your cache.

Redefining the cache key

WARNING

By default, Fastly uses the URL and the Host of a request (plus a special, internal Fastly variable for purging purposes) to create unique HTTP objects. Although Fastly allows you to explicitly set the cache key to define this more precisely, changing the default behavior risks the following:

1. If you add too much information to the cache key, you can significantly reduce your hit ratio.
2. If you make a mistake when explicitly setting the cache key, you can cause all requests to get the same object.
3. If you add anything to the hash, you will need to send a purge for each combination of the URL and value you add in order to purge that specific information from the cache.

To avoid these dangers, consider using the Vary header instead of following the instructions below.

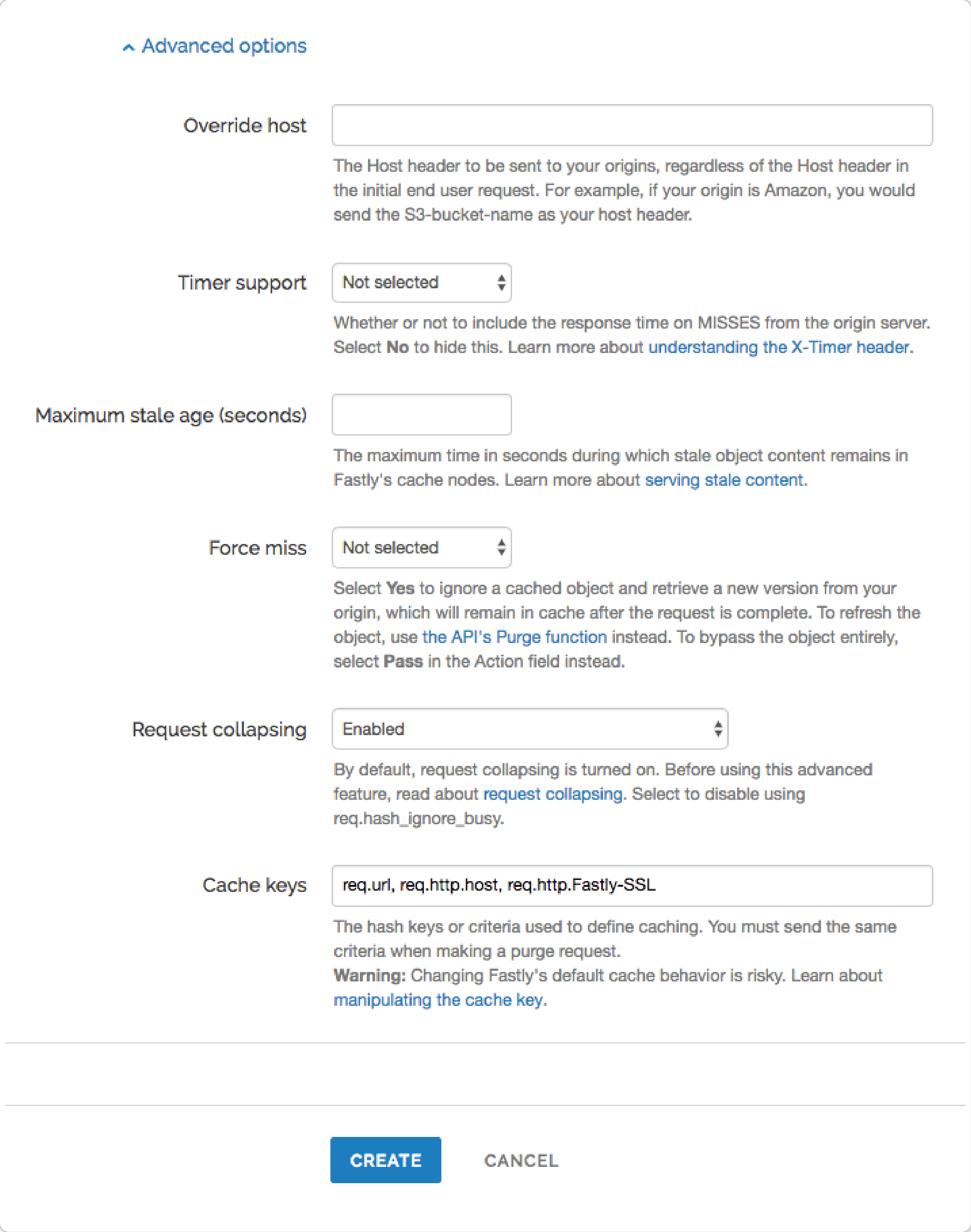
Explicitly setting the cache key

You can set the cache key explicitly (including attaching conditions) by adding a request setting via the Settings page in the configuration controls and including a comma-separated list of cache keys. The values of the cache keys listed are combined to make a single hash, and each unique hash is considered a unique object.

For example, if you don't want the query string to be part of the cache key, but you don't want to change `req.url` so that the query string still ends up in your logs, you could use the following text for the hash keys:

```
req.url.path, req.http.host
```

In the web interface, the text would appear in the Cache keys field:



Override host

The Host header to be sent to your origins, regardless of the Host header in the initial end user request. For example, if your origin is Amazon, you would send the S3-bucket-name as your host header.

Timer support: Not selected

Whether or not to include the response time on MISSES from the origin server. Select **No** to hide this. Learn more about understanding the X-Timer header.

Maximum stale age (seconds)

The maximum time in seconds during which stale object content remains in Fastly's cache nodes. Learn more about serving stale content.

Force miss: Not selected

Select **Yes** to ignore a cached object and retrieve a new version from your origin, which will remain in cache after the request is complete. To refresh the object, use the API's Purge function instead. To bypass the object entirely, select **Pass** in the Action field instead.

Request collapsing: Enabled

By default, request collapsing is turned on. Before using this advanced feature, read about request collapsing. Select to disable using req.hash_ignore_busy.

Cache keys: req.url, req.http.host, req.http.Fastly-SSL

The hash keys or criteria used to define caching. You must send the same criteria when making a purge request.
Warning: Changing Fastly's default cache behavior is risky. Learn about manipulating the cache key.

As a general rule, you should always have `req.url` as one of your cache keys or as part of one.

Purging adjustments when making additions to cache keys

Because purging works on individual hashes, additions to cache keys can complicate purging URLs. However, it can also be simplified.

For example, if you were to change your cache key to just `req.url` and not the default `req.url, req.http.host`, then purging `http://foo.example.com/file.html` would also purge `http://bar.example.com/file.html`. Keep in mind this is because they're actually the same object in the cache!

On the other hand, if you were to change your cache key to `req.url, req.http.host, req.http.Fastly-SSL`, you would have to purge `http://example.com/` and `https://example.com/` individually.

In the latter case, if you were to use the Vary header instead of changing the cache key, you could still have different content on the two URLs, yet purge them with a single purge. In this case you would add a new Cache Header, use `http.Vary` as the Destination, and use the following as the Source:

```
if(beresp.http.Vary, beresp.http.Vary ",", "") "Fastly-SSL"
```

Using a cookie as a cache key

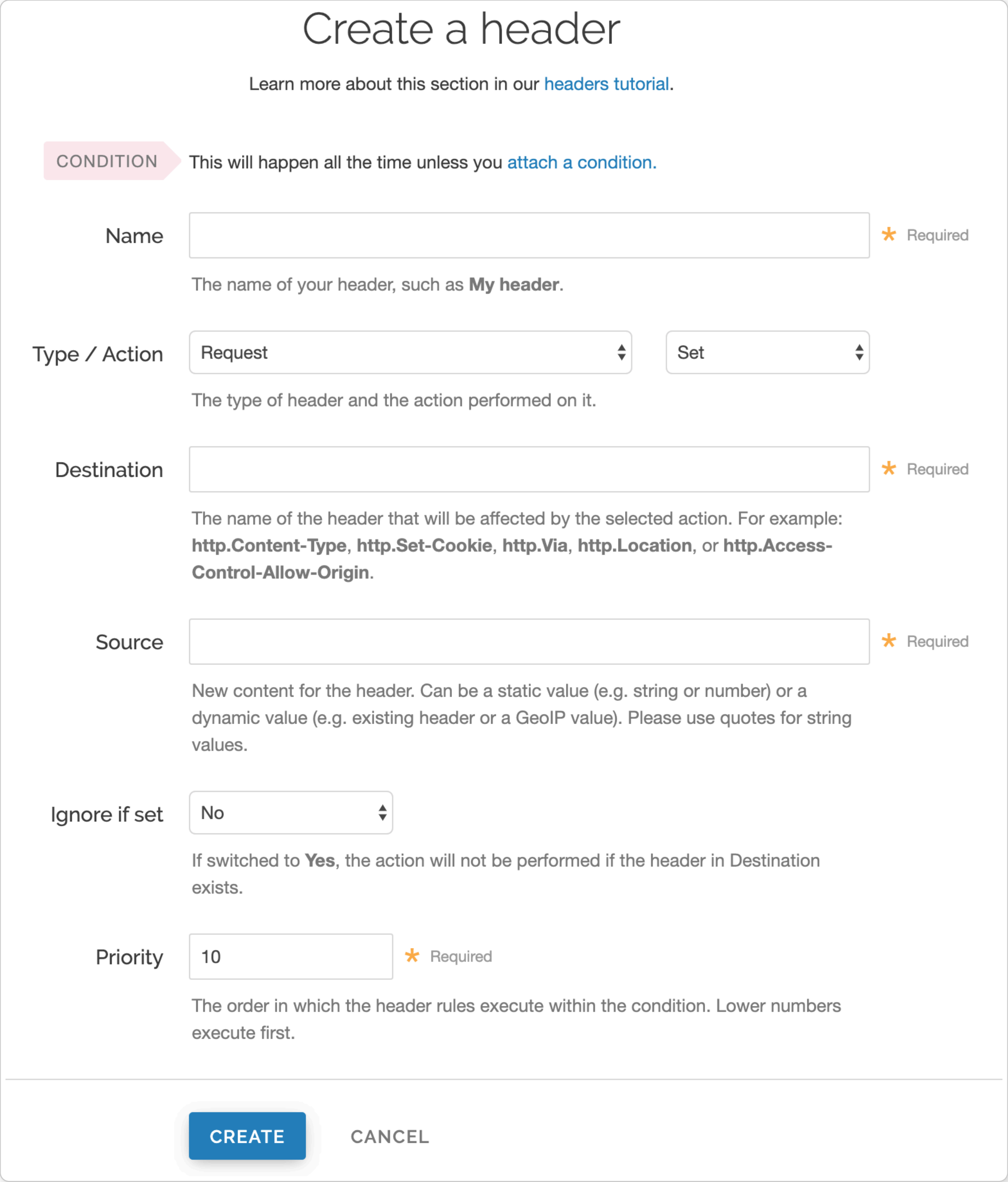
You can use a cookie as a cache key or just check for the presence of a cookie set to a specific value by controlling its request conditions. Both methods are simple and shown in the steps below.

To use a cookie as a cache key

Using a cookie as a cache key looks complicated but it's actually quite simple. Let's say your cookie is called "MyCookie" and it looks like `mycookie=`.

Create new headers

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name ☐ * Required

The name of your header, such as **My header**.

Type / Action: Request, Set

The type of header and the action performed on it.

Destination ☐ * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source ☐ * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

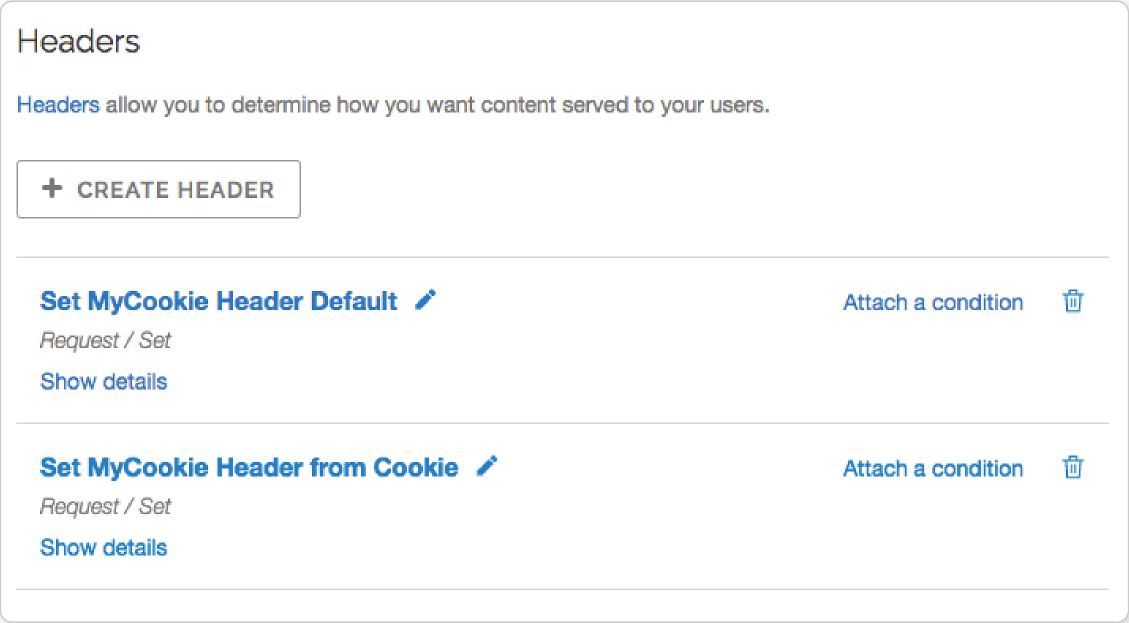
If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Set MyCookie Header Default`.
   - From the **Type** menu, select **Request**, and from the **Action** menu select **Set**.
   - In the **Destination** field, enter `http.X-MyCookie`.
   - In the **Source** field, enter `"0"` (with quotes).

  - Leave the **Ignore if set** menu set to the default, **No**.
  - In the **Priority** field, enter a number representing the order in which the header rule should execute. The default is set to `10` for new headers.
7. Click **Create**. The new header appears in the Headers area of the Content page.
8. Click **Create header** again and create a second new header by filling out the fields as follows:
   - In the **Name** field, enter `Set MyCookie Header from Cookie`.
   - From the **Type** menu, select **Request**, and from the **Action** menu select **Set**.
   - In the **Destination** field, enter `http.X-MyCookie`.
   - In the **Source** field, enter `req.http.cookie:mycookie`.
   - Leave the **Ignore if set** menu set to the default, **No**.
   - In the **Priority** field, enter a larger number than the priority of previous header you just created. For example, if you left the default priority set to `10`, enter `20`.
9. Click **Create**. The second header appears in the Headers area of the Content page.


Headers

Headers allow you to determine how you want content served to your users.

Set MyCookie Header Default

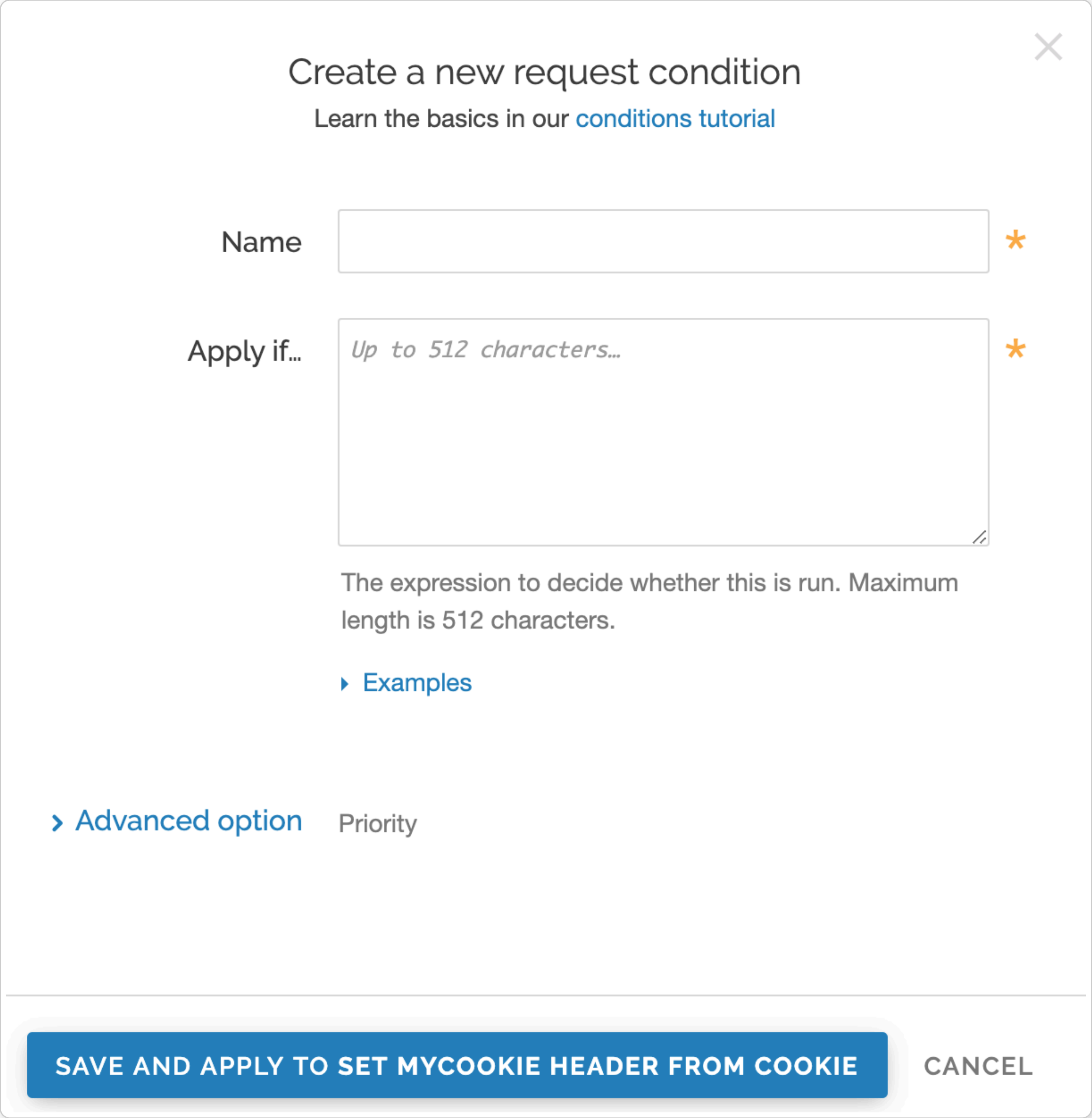
Request / Set

Set MyCookie Header from Cookie

Request / Set

Attach conditions to the new headers

1. Click **Attach a condition** next to the `Set MyCookie Header from Cookie` header.
2. Click **Create a new request condition**.



Create a new request condition

Learn the basics in our conditions tutorial

Name

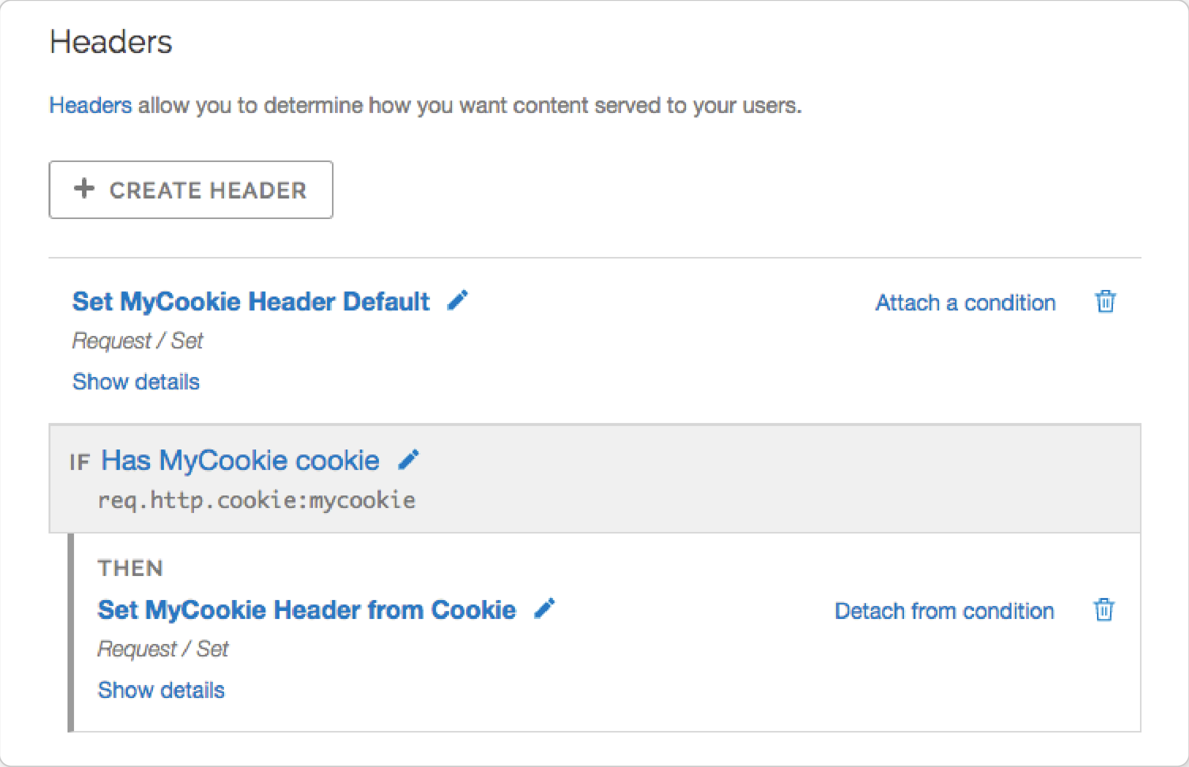
Apply if...

Up to 512 characters...

The expression to decide whether this is run. Maximum length is 512 characters.

▸ Examples
› Advanced option
Priority
SAVE AND APPLY TO SET MYCOOKIE HEADER FROM COOKIE
CANCEL

3. Fill out the fields of the **Create a new request condition** page as follows:
   - In the **Name** field, enter `Has MyCookie cookie`.
   - In the **Apply if** field, enter `req.http.cookie:mycookie`.
4. Click **Save and apply to**. The Headers area now displays the condition that must be met in order for your header to begin being used.

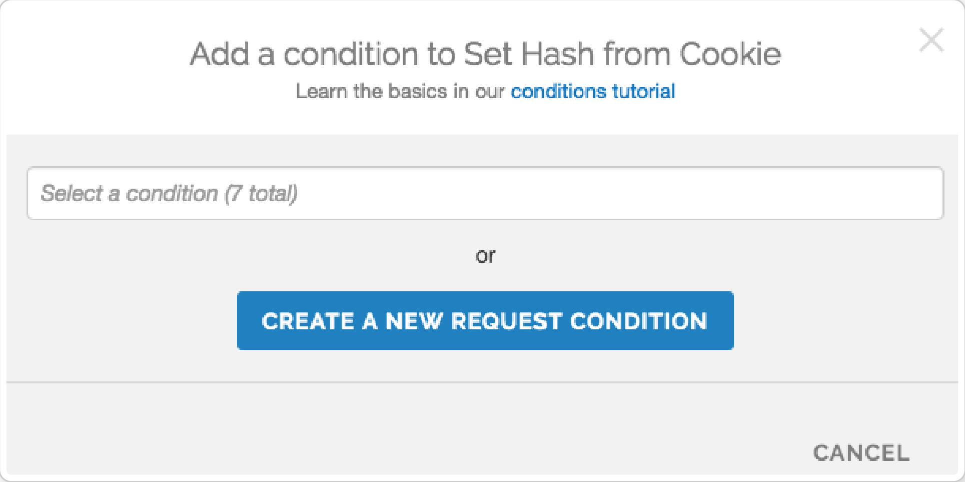


Headers

Headers allow you to determine how you want content served to your users.

+ CREATE HEADER
Set MyCookie Header Default
Attach a condition
Request / Set
Show details
IF Has MyCookie cookie
req.http.cookie:mycookie
THEN
Set MyCookie Header from Cookie
Detach from condition
Request / Set
Show details

5. Click **Settings**.
6. Click **Create request setting**.
7. In the **Name** field, enter `Set Hash from Cookie`.
8. Click **Advanced options** at the bottom of the page.
9. In the **Cache keys** field, enter `req.url, req.http.host, req.http.X-MyCookie`
10. Click **Create**.
11. Click **Attach a condition** next to the new request.

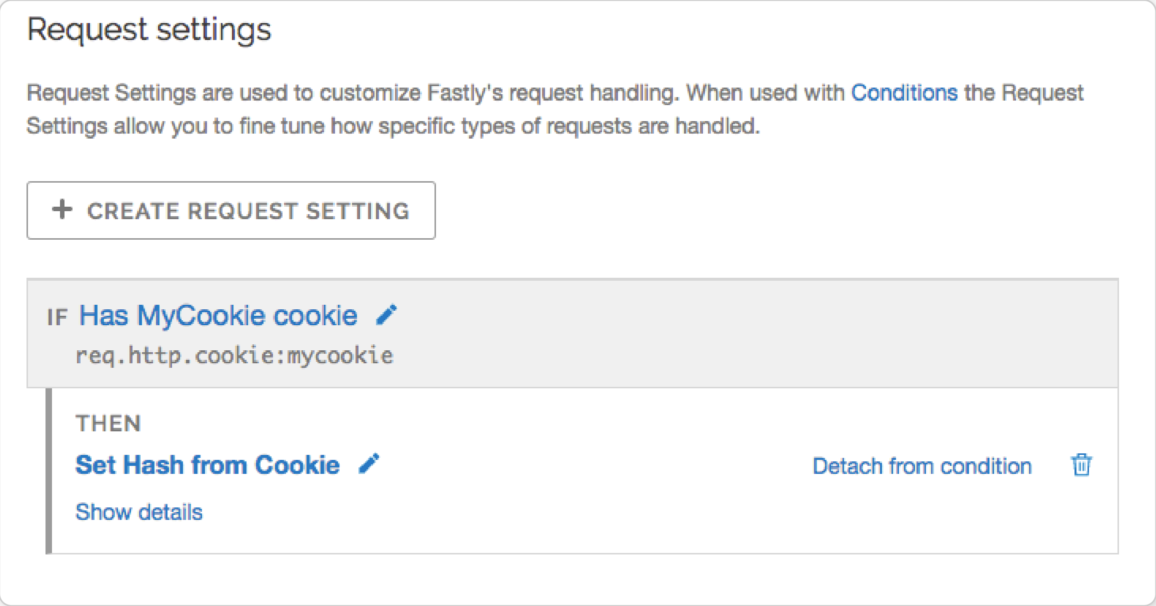


Add a condition to Set Hash from Cookie

Learn the basics in our conditions tutorial

Select a condition (7 total)
or
CREATE A NEW REQUEST CONDITION
CANCEL

12. From the **Select a condition** menu, select `Has MyCookie cookie`. The Request settings area now displays the condition that must be met in order for your request to begin being used.


Request settings

Request Settings are used to customize Fastly's request handling. When used with Conditions the Request Settings allow you to fine tune how specific types of requests are handled.

req.http.cookie:mycookie

13. Click **Activate** to deploy your configuration changes.

To check for the presence of a cookie set to a specific value

An alternative way if you're just checking for the presence of the cookie set to some specific value (e.g., 1):

1. Add a new Request setting where the Cache key field is set to `req.url, req.http.host, "Has mycookie"`.
2. Add a condition to that Request setting where the Apply if field contains `req.http.cookie:mycookie`.

* * *

Response Cookie handling

/en/guides/response-cookie-handling

The traditional way to read response cookies in VCL is to inspect either the `beresp.http.Set-Cookie` or the `resp.http.Set-Cookie` variables and then extract values using regular expressions. However this is not ideal since attempting to parse potentially complicated or quoted strings with regular expressions is brittle and prone to being tripped up by edge cases. It also doesn't allow for reading multiple headers with the same name such as when an origin sends multiple `Set-Cookie` headers. Because of these two reasons Fastly supports a method for extracting a named value out of `Set-Cookie` headers no matter how many there are.

To access a named value simply use the function with either `beresp` or `resp` depending on what part of the request you're in - so either

```
setcookie.get_value_by_name(beresp, "name")
```

or

```
setcookie.get_value_by_name(resp, "name")
```

as appropriate, replacing `"name"` with whatever the name of the value is. So for example, given this HTTP response from an origin:

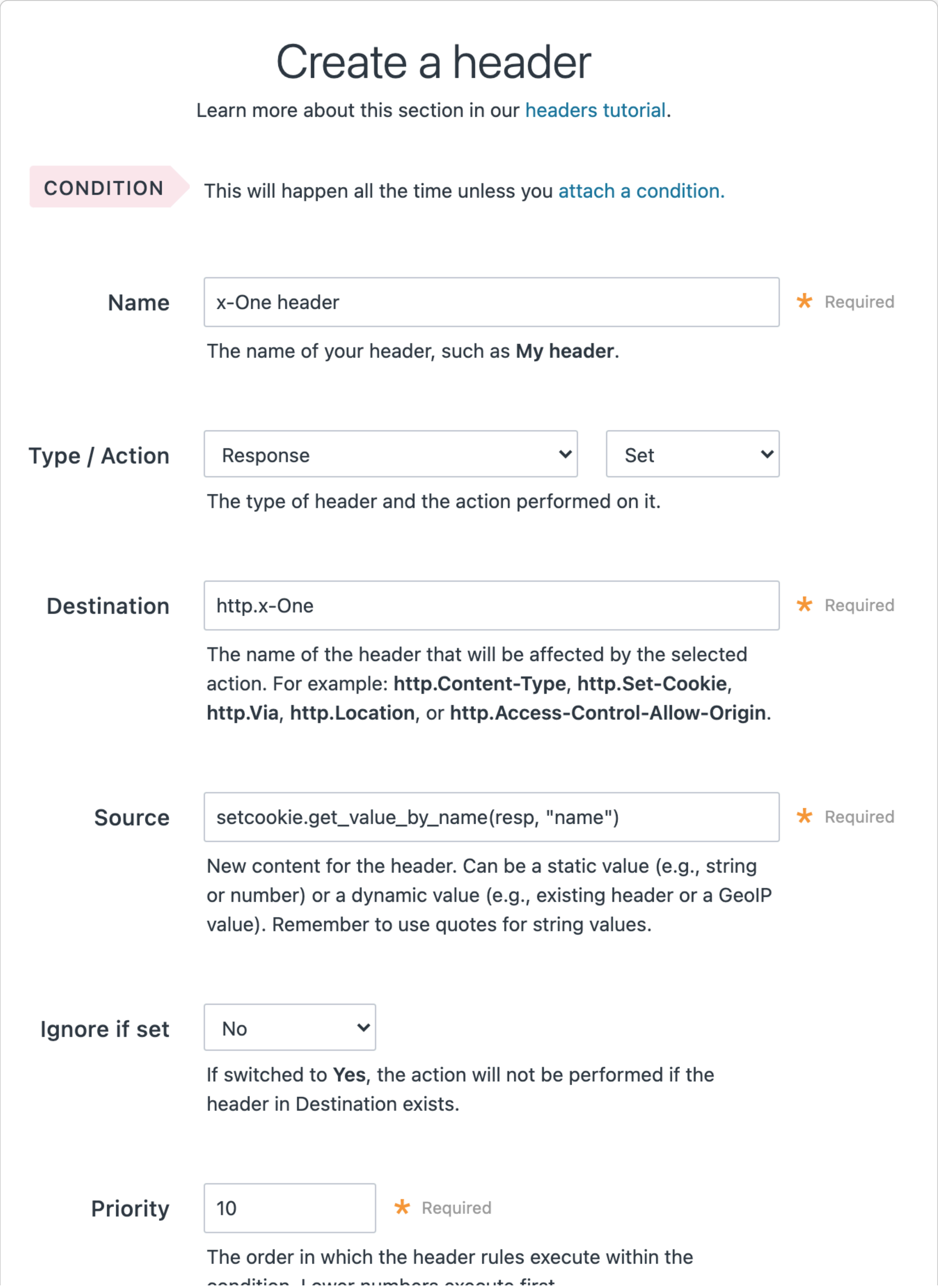
```
HTTP/1.1 200 OK
Cache-Control: max-age=60
Content-Type: text/html; charset=utf-8
Content-Length: 80806
Accept-Ranges: bytes
Date: Tue, 11 Aug 2015 19:00:04 GMT
Age: 123
Connection: keep-alive
Set-Cookie: one=a; httponly; secure
Set-Cookie: two=b or not to b; httponly
```

then using the function like this:

```
set resp.http.X-One = setcookie.get_value_by_name(resp, "one");
set resp.http.X-Two = setcookie.get_value_by_name(resp, "two");
```

will set `resp.http.X-One` to be "a" and `resp.http.X-Two` to "b or not to b".

This logic can be used in uploaded custom VCL, as well as throughout the web interface. For example:



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name x-One header ✱ Required

The name of your header, such as **My header**.

Type / Action Response Set

The type of header and the action performed on it.

Destination http.x-One ✱ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source setcookie.get_value_by_name(resp, "name") ✱ Required

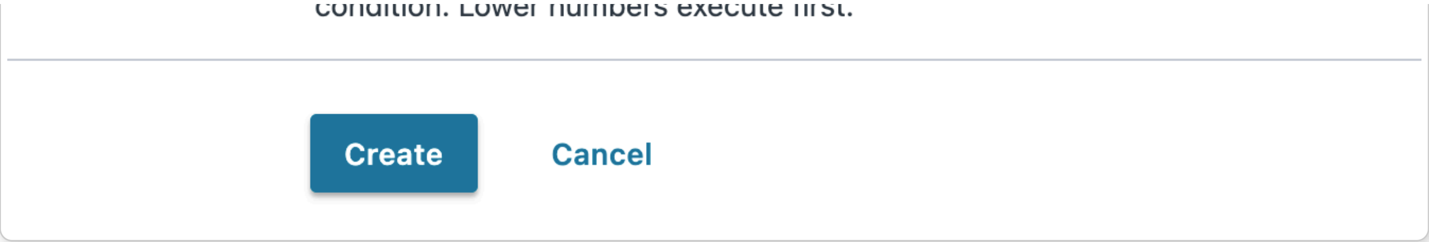
New content for the header. Can be a static value (e.g., string or number) or a dynamic value (e.g., existing header or a GeoIP value). Remember to use quotes for string values.

Ignore if set No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority 10 ✱ Required

The order in which the header rules execute within the condition. Lower numbers execute first.



condition. Lower numbers execute first.

* * *

Tracking your origin's name, IP, and port

/en/guides/tracking-your-origins-name-ip-and-port

Being able to track information related to your origin can be helpful in troubleshooting errors and making sure requests are processed as expected. Fastly provides three VCL variables that allow you to see and track origin information:

- `beresp.backend.name`
- `beresp.backend.port`
- `beresp.backend.ip`

You can create VCL Snippets that use these variables to capture information about an origin and then set up remote log streaming to save that information.

Retrieve the origin information

Create a regular VCL Snippet to retrieve the origin information:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL snippets**.
5. Click **Create snippet**.
6. Fill out the Create a VCL snippet fields as follows:
    - In the **Name** field, enter an appropriate name (for example, `Retrieve-Origin-Information`).
    - From the **Type** menu, select **within subroutine**.
    - From the **Select subroutine** menu, select **fetch (**`vcl_fetch`**)**.
    - In the **VCL** field, add the following VCL logic:

```
# save the variables for access in deliver
set beresp.http.Your-Backend-Name = beresp.backend.name;
set beresp.http.Your-Backend-IP-Port = beresp.backend.ip ":" beresp.backend.port;
```

7. Click **Create** to create the snippet.

Change the response header to a request header

Create another regular VCL Snippet that changes the response header to a request header before logging the information:

1. Click **VCL snippets**.
2. Click **Create snippet**.
3. Fill out the Create a VCL snippet fields as follows:
    - In the **Name** field, enter an appropriate name (for example, `Remove-Origin-Information`).
    - From the **Type** controls, select **within subroutine**.
    - From the **Select subroutine** menu, select **deliver (**`vcl_deliver`**)**.
    - In the **VCL** field, add the following VCL logic:

```
if (fastly_info.state ~ "(MISS|PASS)") {
   # only on a miss or pass
   # save the responses back to req.request because
   # request headers are not sent back to the client
   set req.http.Your-Backend-Name = resp.http.Your-Backend-Name;
   set req.http.Your-Backend-IP-Port = resp.http.Your-Backend-IP-Port;
   }

# remove the identifying information from the response
unset resp.http.Your-Backend-Name;
unset resp.http.Your-Backend-IP-Port;
```

4. Click **Create** to create the snippet.

NOTE

If a response header like `beresp.http.Your-Backend-Name` exists in `vcl_fetch` when VCL is processed, then a corresponding response header `resp.http.Your-Backend-Name` will exist in `vcl_deliver` and will be logged by default. This means that response headers will be included in response output, exposing origin information (e.g., IP address, port number, or origin name) in the process. Fortunately, request headers are not passed back to the client, but their information remains accessible from `vcl_log`.

Set up remote log streaming

Once your snippets are created, you can set up remote log streaming for tracking purposes. Do this by adding `req.http.Header-Name` to the **Log format** field in the logging endpoint you configured in the remote log streaming guide. Using the example above, you would add `req.http.Your-Backend-Name` and `req.http.Your-Backend-`

`IP-Port`. Once you've reviewed the changes you've made, click the **Activate** button to deploy these configuration changes for your service.

NOTE

When you configure remote log streaming to capture this information, remember the VCL is executed twice, once on the shield node and again on the edge node. This means that the first log (from the shield node) will have the origin's information and the second log (from the edge node) will have the shield's information.

* * *

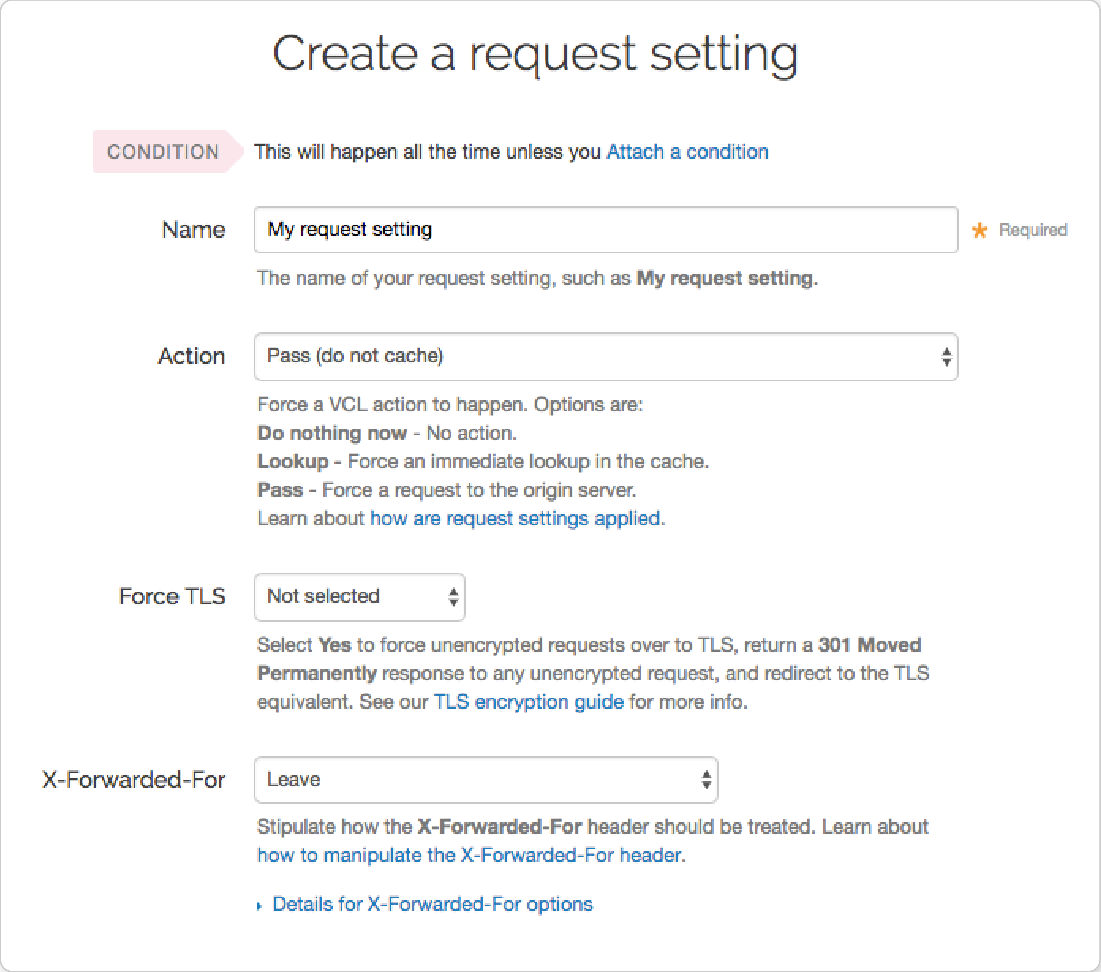
Understanding the different PASS action behaviors

/en/guides/understanding-the-different-pass-action-behaviors

Passing with a request setting and with a cache setting triggers very different behavior in Varnish. Within VCL, passing with a request setting is the same as `return(pass)` in `vcl_recv`. Passing with a cache setting is the same as `return(pass)` in `vcl_fetch`. If you are familiar with Varnish 3+, passing with a cache setting is equivalent to `return(hit_for_pass)`.

Using a request setting

Passing with a request setting translates within your generated VCL to `return(pass)` in `vcl_recv`. Varnish will not perform a lookup to see if an object is in cache and the response from the origin will not be cached.



Create a request setting

CONDITION This will happen all the time unless you Attach a condition

Name: My request setting ★ Required

The name of your request setting, such as **My request setting**.

Action: Pass (do not cache)

Force a VCL action to happen. Options are:
Do nothing now - No action.
Lookup - Force an immediate lookup in the cache.
Pass - Force a request to the origin server.
Learn about how are request settings applied.

Force TLS: Not selected

Select **Yes** to force unencrypted requests over to TLS, return a **301 Moved Permanently** response to any unencrypted request, and redirect to the TLS equivalent. See our TLS encryption guide for more info.

X-Forwarded-For: Leave

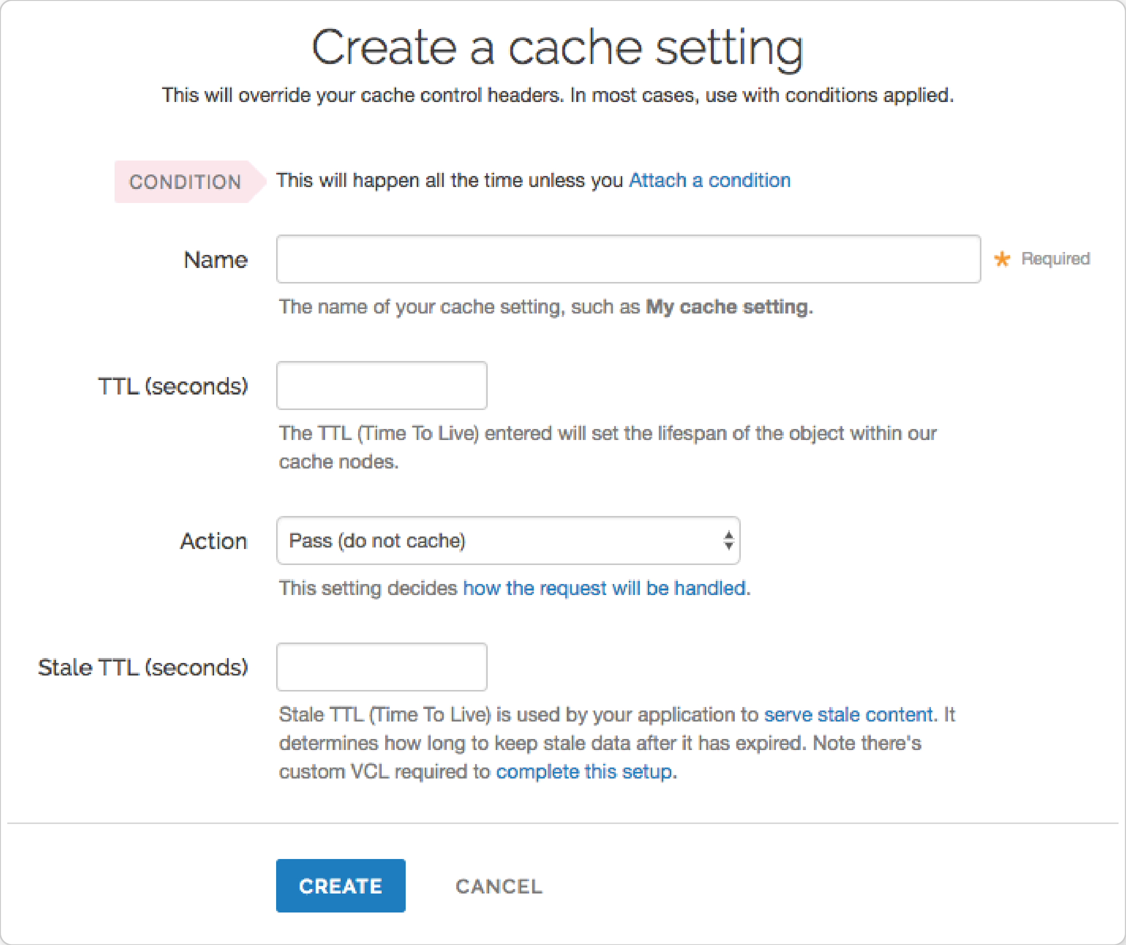
Stipulate how the **X-Forwarded-For** header should be treated. Learn about how to manipulate the X-Forwarded-For header.

▸ Details for X-Forwarded-For options

Passing in this manner disables request collapsing. Normally simultaneous requests for the same object that result in cache misses will be collapsed down to a single request to the origin. While the first request is sent to the origin, the other requests for that object are queued until a response is received. When requests are passed in `vcl_recv`, they will all go to the origin separately without being collapsed.

Using a cache setting

Passing with a cache setting translates within your generated VCL to `return(pass)` in `vcl_fetch`. At this point in the flow of a request, Varnish has performed a lookup and determined that the object is not in cache. A request to the origin has been made; however, in `vcl_fetch` we have determined that the response is not cacheable. In Fastly's default VCL, this can happen based on the presence of a `Set-Cookie` response header from the origin.



Create a cache setting

This will override your cache control headers. In most cases, use with conditions applied.

This will happen all the time unless you Attach a condition

Name

★ Required

The name of your cache setting, such as **My cache setting**.

TTL (seconds)

The TTL (Time To Live) entered will set the lifespan of the object within our cache nodes.

Action: Pass (do not cache)

This setting decides how the request will be handled.

Stale TTL (seconds)

Stale TTL (Time To Live) is used by your application to serve stale content. It determines how long to keep stale data after it has expired. Note there's custom VCL required to complete this setup.

Passing in `vcl_fetch` is often not desirable because request collapsing is *not* disabled. This makes sense since Varnish is not aware in `vcl_recv` that the object is uncacheable. On the first request for an object that will be later passed in `vcl_fetch`, all other simultaneous cache misses will be queued. Once the response from the origin is received and Varnish has realized that the request should be passed, the queued requests are sent to the origin.

This creates a scenario where two users request an object at the same time, and one user must wait for the other before being served. If these requests were passed in `vcl_recv`, neither user would need to wait.

To get around this disadvantage, when a request is passed in `vcl_fetch`, Varnish creates what is called a hit-for-pass object. These objects have their own TTLs and while they exist, Varnish will pass any requests for them as if the pass had been triggered in `vcl_recv`. For this reason, it is important to set a TTL that makes sense for your case when you pass in `vcl_fetch`. All future requests for the object will be passed until the hit-for-pass object expires. Hit-for-pass objects can also be purged like any other object.

Even with this feature, there will be cases where simultaneous requests will be queued and users will wait. Whenever there is not a hit-for-pass object in cache, these requests will be treated as if they are normal cache misses and request collapsing will be enabled. Whenever possible it is best avoid relying on passing in `vcl_fetch`.

Using req.hash_always_miss and req.hash_ignore_busy

Setting `req.hash_always_miss` forces a request to miss whether it is in cache or not. This is different than passing in `vcl_recv` in that the response will be cached and request collapsing will not be disabled. Later on the request can still be passed in `vcl_fetch` if desired.

A second relevant variable is `req.hash_ignore_busy`. Setting this to true disables request collapsing so that each request is sent separately to origin. When `req.hash_ignore_busy` is enabled all responses will be cached and each response received from the origin will overwrite the last. Future requests for the object that are served from cache will receive the copy of the object from the last cache miss to complete. `req.hash_ignore_busy` is used mostly for avoiding deadlocks in complex multi-Varnish setups.

Setting both these variables can be useful to force requests to be sent separately to the origin while still caching the responses.

* * *

	Uploading custom VCL
🔗	/en/guides/uploading-custom-vcl

Fastly allows you create your own Varnish Configuration Language (VCL) files with specialized configurations. By uploading custom VCL files, you can use custom VCL and Fastly VCL together at the same time. Any time you upload VCL files, you can preview the VCL prior to activating a new version of your service. Keep in mind that your custom VCL always takes precedence over VCL generated by Fastly.

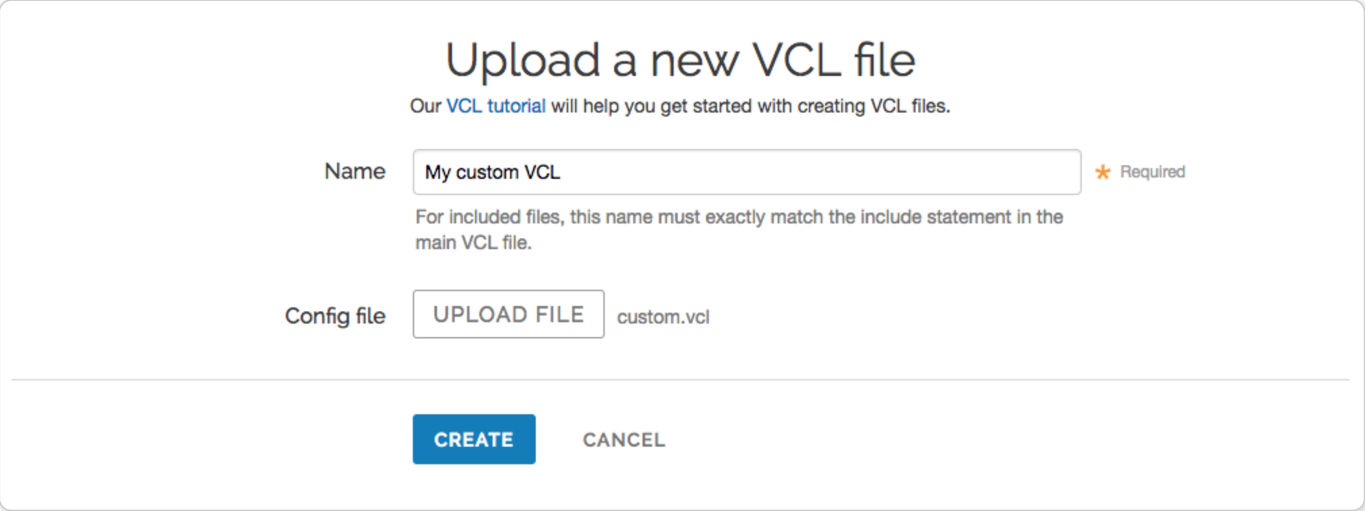
IMPORTANT

Personal data should not be incorporated into VCL. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

Uploading a VCL file

Follow these instructions to upload a custom VCL file:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Custom VCL**.
5. Click **Upload a new VCL file**.



Upload a new VCL file

Our VCL tutorial will help you get started with creating VCL files.

Name: My custom VCL

For included files, this name must exactly match the include statement in the main VCL file.

Config file: custom.vcl

6. In the **Name** field, enter the name of the VCL file. For included files, this name must match the include statement in the main VCL file. See how to include additional VCL configurations for more information.
7. Click **Upload file** and select a file to upload.

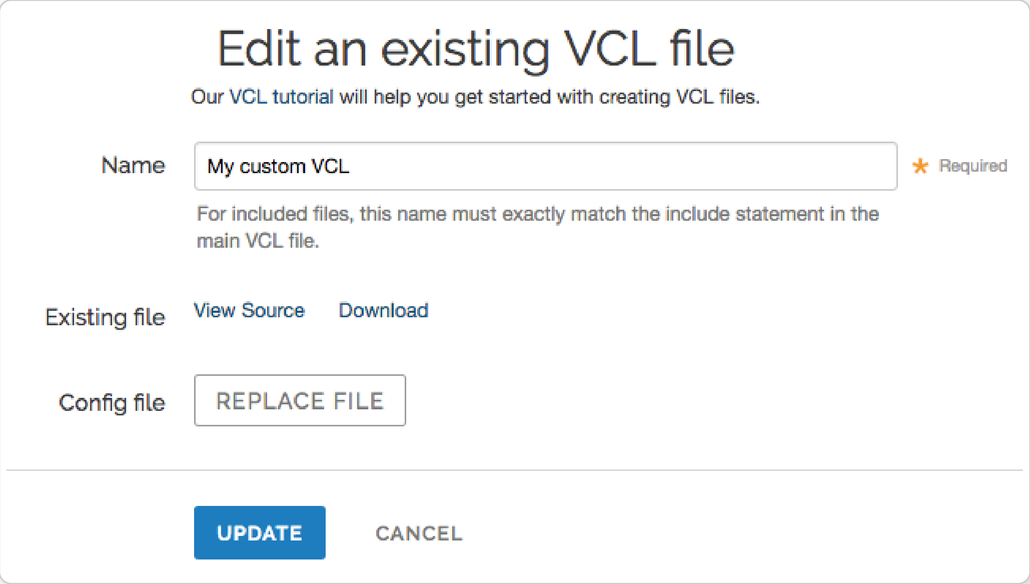
> IMPORTANT
>
> Don't upload generated VCL that you've downloaded from the Fastly web interface. Instead, edit and then upload a copy of Fastly's VCL boilerplate to avoid errors.

8. Click **Create**. The VCL file appears in the Varnish Configurations area.
9. Click **Activate** to deploy your configuration changes.

Editing a VCL file

To edit an existing VCL file, follow these instructions:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Custom VCL**.
5. In the **Varnish Configurations** area, click the VCL file you want to edit.


Edit an existing VCL file

Our VCL tutorial will help you get started with creating VCL files.

Name: My custom VCL

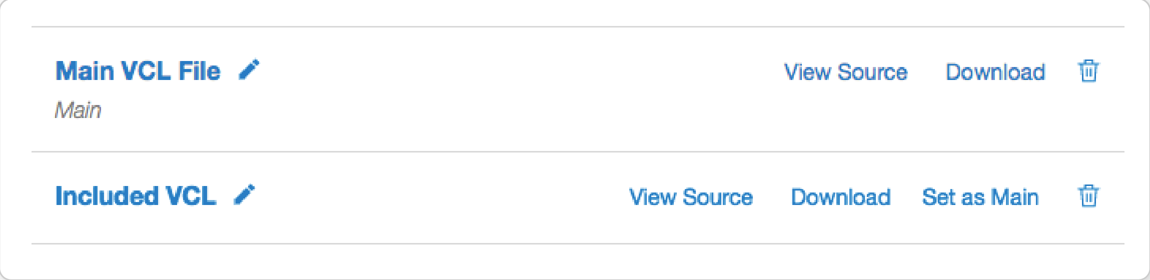
For included files, this name must exactly match the include statement in the main VCL file.

6. *(Optional)* In the **Name** field, enter a new name for the VCL file.
7. Click **Download** to download the appropriate file.
8. Make the necessary changes to your file and save them.
9. Click **Replace file** and select the file you updated. The selected file replaces the current VCL file and the file name appears next to the button.
10. Click **Update** to update the VCL file in the Fastly application.
11. Click **Activate** to deploy your configuration changes.

Including additional VCL configurations

To make your full VCL configuration easier to maintain, you can split it up into multiple files that are accessed by a main VCL file. This allows you to separate out chunks of logic (for example, logic that has a specific purpose or that might change frequently) into as many separate files as makes sense.

1. Start by isolating a portion of VCL and placing it in a separate file. The name of the file doesn't matter, nor does the file extension. A `foo.vcl` file will work just as well as a `bar.txt` file.
2. Upload the file to include it in your Varnish configurations and give it a unique name when you fill out the **Name** field at the time of upload (for example, you could call it `Included VCL`). The uploaded file will appear in the Varnish Configurations area along with your main VCL file.



3. Enter the name of the included VCL file on a separate line in the main VCL configuration file. For example, your **Included VCL** file would get added to the main VCL file in a single line like this:

```
include "Included VCL";
```

4. Continue uploading VCL files and then including them in your main VCL using the syntax `include "<VCL FILE>";` where `<VCL FILE>` exactly matches the name you entered in the **Name** field.

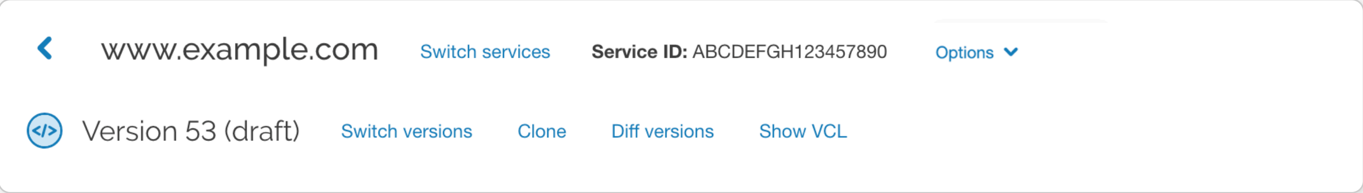
TIP

Our guide to manually creating access control lists demonstrates a common example of using included VCL.

Previewing VCL before activation

Follow these instructions to preview VCL prior to activating a service version:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Show VCL**.



Service ID: ABCDEFGH123457890

The VCL preview page appears.

* * *

Using dynamic VCL Snippets

/en/guides/using-dynamic-vcl-snippets

Dynamic VCL Snippets are one of two types of snippets that allow you to insert small sections of VCL logic into your service configuration without requiring custom VCL (though you can still include snippets in custom VCL when necessary).

You can only create dynamic snippets via the API. Because they are versionless objects (much like dictionaries or ACLs at the edge), dynamic snippets can be modified independently from changes to your Fastly service. This means you can modify snippet code rapidly without deploying a service version that may not be ready for production.

Creating and using a dynamic VCL Snippet

Using the curl command line tool, make the following API call in a terminal application:

```
$ curl -X POST -s https://api.fastly.com/service/<Service ID>/version/<Editable Version>
```

Fastly returns a JSON response that looks like this:

```
{
 "service_id": "<Service Id>",
 "version": "<Editable Version>",
 "name": "my_dynamic_snippet_name",
 "type": "recv",
 "priority": 100,
 "dynamic": 1,
 "content": null,
 "id": "decafbad12345",
 "created_at": "2016-09-09T20:34:51+00:00",
 "updated_at": "2016-09-09T20:34:51+00:00",
 "deleted_at": null
}
```

NOTE

The returned JSON includes `"content": null`. This happens because the content is stored in a separate, unversioned object.

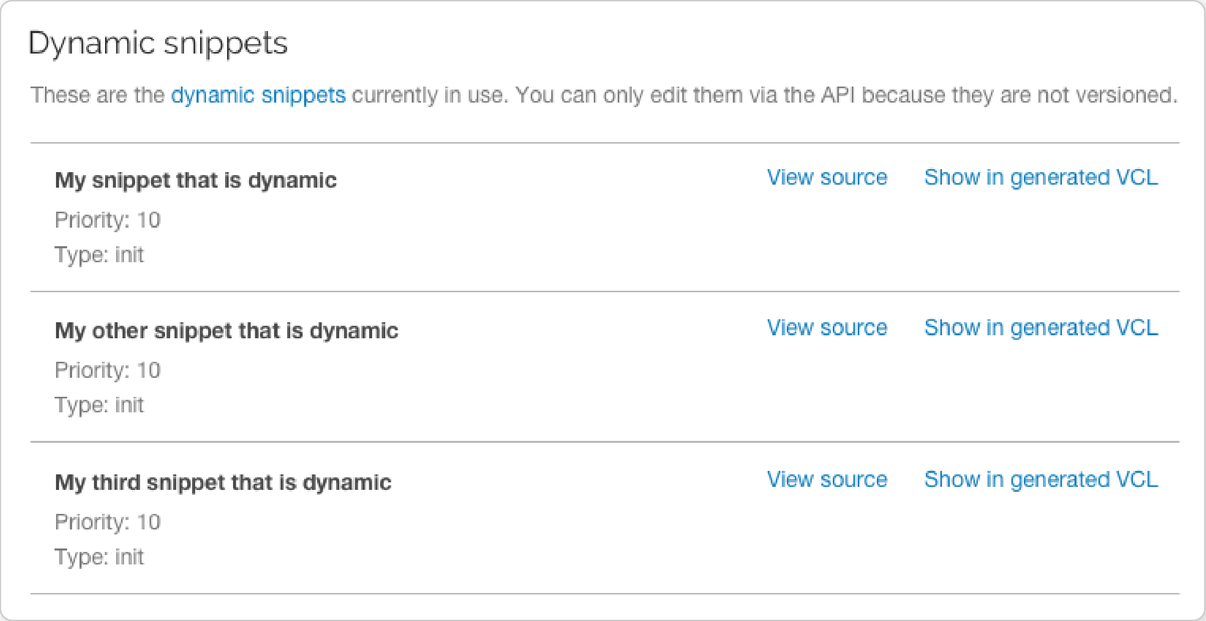
Viewing dynamic VCL Snippets in the web interface

You can view a list of dynamic VCL snippets. You can also view just the source of a specific snippet or a specific snippet's location in generated VCL.

Viewing a list of dynamic VCL Snippets

To view the entire list of a service's dynamic VCL Snippets directly in the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate VCL service. You can use the search box to search by ID, name, or domain.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**. The VCL Snippets page appears listing all dynamic VCL Snippets for your service in the Dynamic snippets area.



Dynamic snippets

These are the dynamic snippets currently in use. You can only edit them via the API because they are not versioned.

My snippet that is dynamic
Priority: 10
Type: init

My other snippet that is dynamic
Priority: 10
Type: init

My third snippet that is dynamic
Priority: 10
Type: init

Viewing the source of a specific snippet

You can view just the source of a specific snippet:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate VCL service. You can use the search box to search by ID, name, or domain.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **View Source** to the right of the name of the snippet.

Viewing the location of a specific snippet in generated VCL

You can view a specific snippet's location in generated VCL:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate VCL service. You can use the search box to search by ID, name, or domain.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Show in Generated VCL** to the right of the name of the snippet. The Generated VCL window appears.

Fetching a list of all dynamic VCL Snippets

To list all dynamic VCL Snippets attached to a service, make the following API call in a terminal application:

```
$ curl -X GET -s https://api.fastly.com/service/<Service ID>/version/<Version ID>/snippe -H
```

Fetching an individual dynamic VCL Snippet

To fetch an individual snippet, make the following API call in a terminal application:

```
$ curl -X GET -s https://api.fastly.com/service/<Service ID>/snippet/<Snippet ID> -H "Fa  ly
```

Unlike fetching regular VCL Snippets, you do not include the version in the URL and you must use the ID returned when the snippet was created, not the name.

Updating an existing dynamic VCL Snippet

To update an individual snippet, make the following API call in a terminal application:

```
$ curl -X PUT -s https://api.fastly.com/service/<Service ID>/snippet/<Snippet ID> -H "Fa  ly
```

Deleting an existing dynamic VCL Snippet

To delete an individual snippet, make the following API call in a terminal application:

```
$ curl -X DELETE -s https://api.fastly.com/service/<Service ID>/version/<Version ID>/sni  et
```

Including dynamic snippets in custom VCL

By specifying a location of `none` for the `type` parameter, snippets will not be rendered in VCL. This allows you to include snippets in custom VCL using the following syntax:

`include "snippet::<snippet name>"`

The same VCL Snippet can be included in custom VCL in as many places as needed.

Example use: blocking site scrapers

Say you wanted to implement some pattern matching against incoming requests to block someone trying to scrape your site. Say also that you've developed a system that looks at all incoming requests and generates a set of rules that can identify scrapers using a combination of the incoming IP address, the browser, and the URL they're trying to fetch. Finally, say that the system updates the rules every 20 minutes.

If, during system updates, your colleagues are also making changes to the rest of your Fastly configuration, you probably don't want the system to automatically deploy the latest version of the service since it might be untested. Instead you could generate the rules as a Dynamic VCL Snippet. Whenever the snippet is updated, all other logic remains the same as the currently deployed version and only your rules are modified.

* * *

Using regular VCL Snippets

/en/guides/using-regular-vcl-snippets

Regular VCL Snippets are one of two types of snippets that allow you to insert small sections of VCL logic into your service configuration without requiring custom VCL (though you can still include snippets in custom VCL when necessary).

Unlike dynamic snippets, regular snippets can be created via the web interface or via the API. They are considered *versioned* objects. They belong to a specific service and any modifications you make to the snippet are locked and deployed when you deploy a new version of that service. We continue to clone them and deploy them with a service until you specifically delete them.

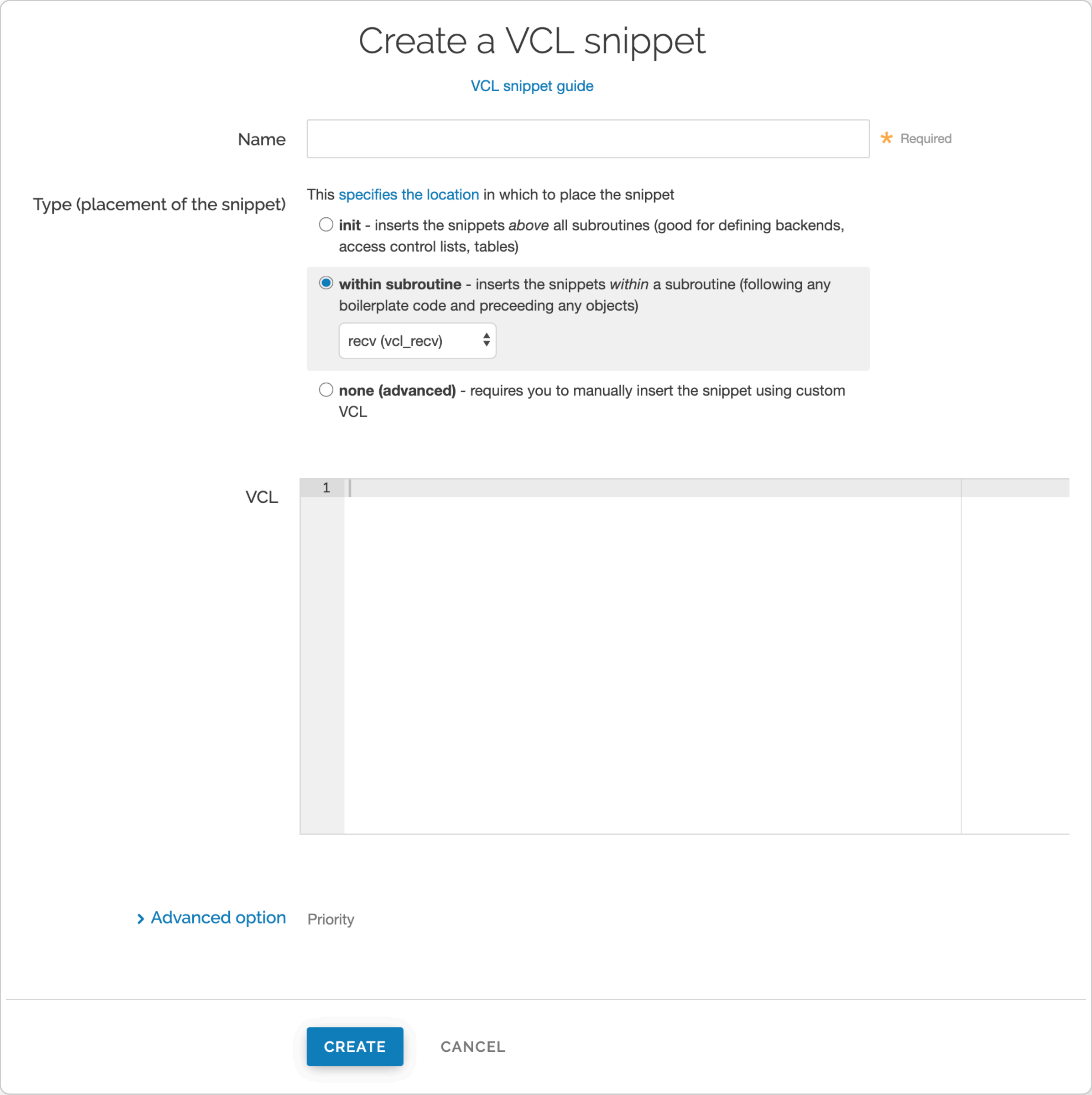
Creating a regular VCL Snippet

You can create regular VCL Snippets via the web interface or via the API.

Via the web interface

To create a regular VCL Snippet via the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create Snippet**.



Create a VCL snippet

VCL snippet guide

Name ___ * Required

Type (placement of the snippet)

This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

recv (vcl_recv)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL

6. In the **Name** field, enter an appropriate name (for example, `Example Snippet`).

7. Using the **Type** controls, select the location in which the snippet should be placed as follows:

   - Select `init` to insert it above all subroutines in your VCL.
   - Select `within subroutine` to insert it within a specific subroutine and then select the specific subroutine from the **Select subroutine** menu.
   - Select `none (advanced)` to insert it manually. See Including regular snippets in custom VCL for the additional manual insertion requirements if you select this option.

8. In the **VCL** field, enter the snippet of VCL logic to be inserted for your service version.

9. Click **Advanced options** at the bottom of the page.

10. *(Optional)* In the **Priority** field, enter the order in which you want the snippet to execute. Lower numbers execute first.

11. Click **Create** to create the snippet.

Via the API

To create a regular VCL Snippet via the API, make the following API call using the curl command line tool in a terminal application:

```
$ curl -X POST -s https://api.fastly.com/service/<Service ID>/version/<Editable Version>
```

Fastly returns a JSON response that looks like this:

```
{
 "service_id": "<Service Id>",
 "version": "<Editable Version>",
 "name": "my_regular_snippet",
 "type": "recv",
 "content": "if ( req.url ) {\n set req.http.my-snippet-test-header = \"true\";\n}",
 "priority": 100,
 "dynamic": 0,
 "id": "56789exampleid",
 "created_at": "2016-09-09T20:34:51+00:00",
 "updated_at": "2016-09-09T20:34:51+00:00",
 "deleted_at": null
}
```

NOTE

When regular VCL snippets get created, an `id` field will be returned that isn't used. The field only applies to dynamic VCL Snippets. In addition, the returned JSON includes a populated `content` field because the snippet content is stored in a versioned object.

Viewing regular VCL Snippets in the web interface

You can view a list of regular VCL snippets. You can also view just the source of a specific snippet or a specific snippet's location in generated VCL.

Viewing a list of regular VCL Snippets

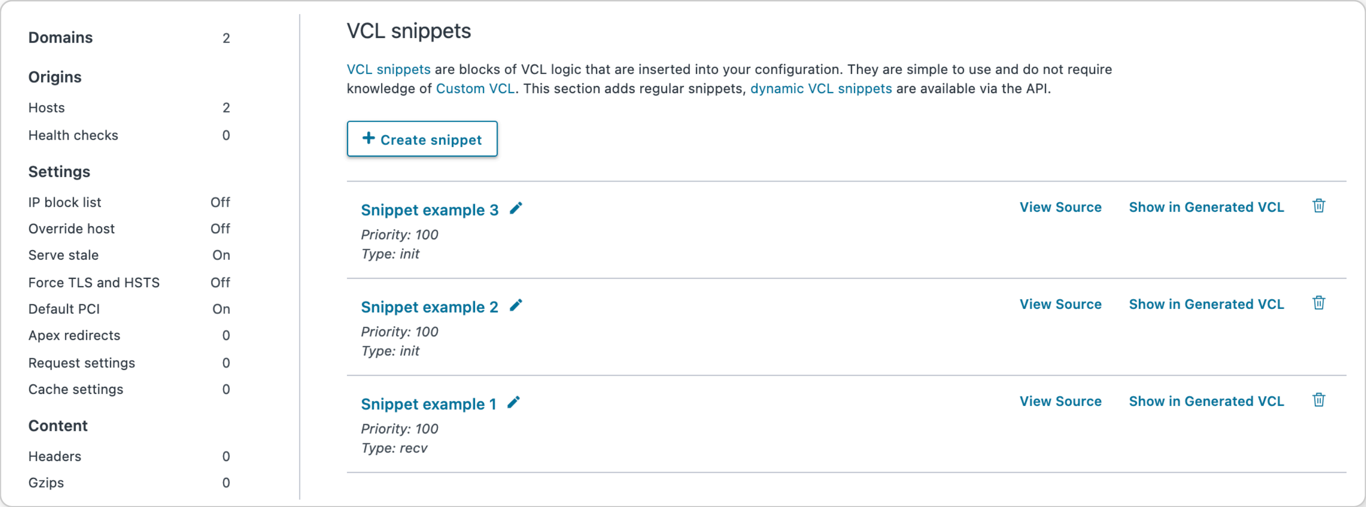
To view the entire list of a service's regular VCL Snippets directly in the web interface:

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **VCL Snippets**. The VCL Snippets page appears listing all available VCL snippets for your service.



Domains	2
Origins	
Hosts	2
Health checks	0
Settings	
IP block list	Off
Override host	Off
Serve stale	On
Force TLS and HSTS	Off
Default PCI	On
Apex redirects	0
Request settings	0
Cache settings	0
Content	
Headers	0
Gzips	0

VCL snippets

VCL snippets are blocks of VCL logic that are inserted into your configuration. They are simple to use and do not require knowledge of Custom VCL. This section adds regular snippets, dynamic VCL snippets are available via the API.

Snippet example 3
Priority: 100
Type: init

Snippet example 2
Priority: 100
Type: init

Snippet example 1
Priority: 100
Type: recv

Viewing the source of a specific snippet

You can view just the source of a specific snippet:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **View Source** to the right of the name of the snippet. A view source window appears.

Viewing the location of a specific snippet in generated VCL

You can view a specific snippet's location in generated VCL:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Show in Generated VCL** to the right of the name of the snippet. The Generated VCL window appears.

Fetching regular VCL Snippets via the API

You can fetch regular VCL Snippets for a particular service via the API either singly or all at once.

Fetching an individual regular VCL Snippet

To fetch an individual snippet, make the following API call in a terminal application:

```
$ curl -X GET -s https://api.fastly.com/service/<Service ID>/version/<Editable Version>/ip
```

Unlike fetching dynamic VCL Snippets you include the version in the URL and you must use the name of the snippet, not the ID.

Fetching a list of regular VCL Snippets

To list all regular VCL Snippets attached to a service, make the following API call in a terminal application:

```
$ curl -X GET -s https://api.fastly.com/service/<Service ID>/version/<Editable Version>/ip
```

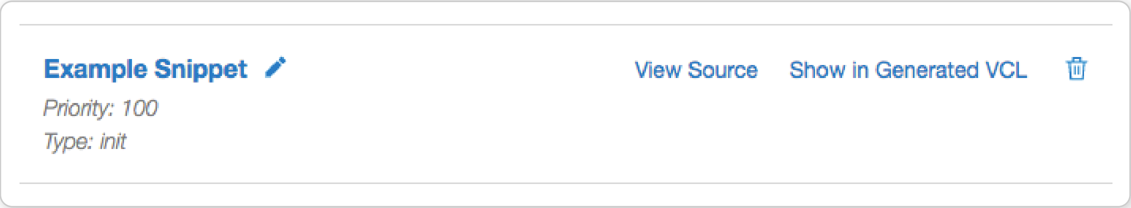
Updating an existing regular VCL Snippet

You can update existing regular VCL Snippets via the web interface or via the API.

Via the web interface

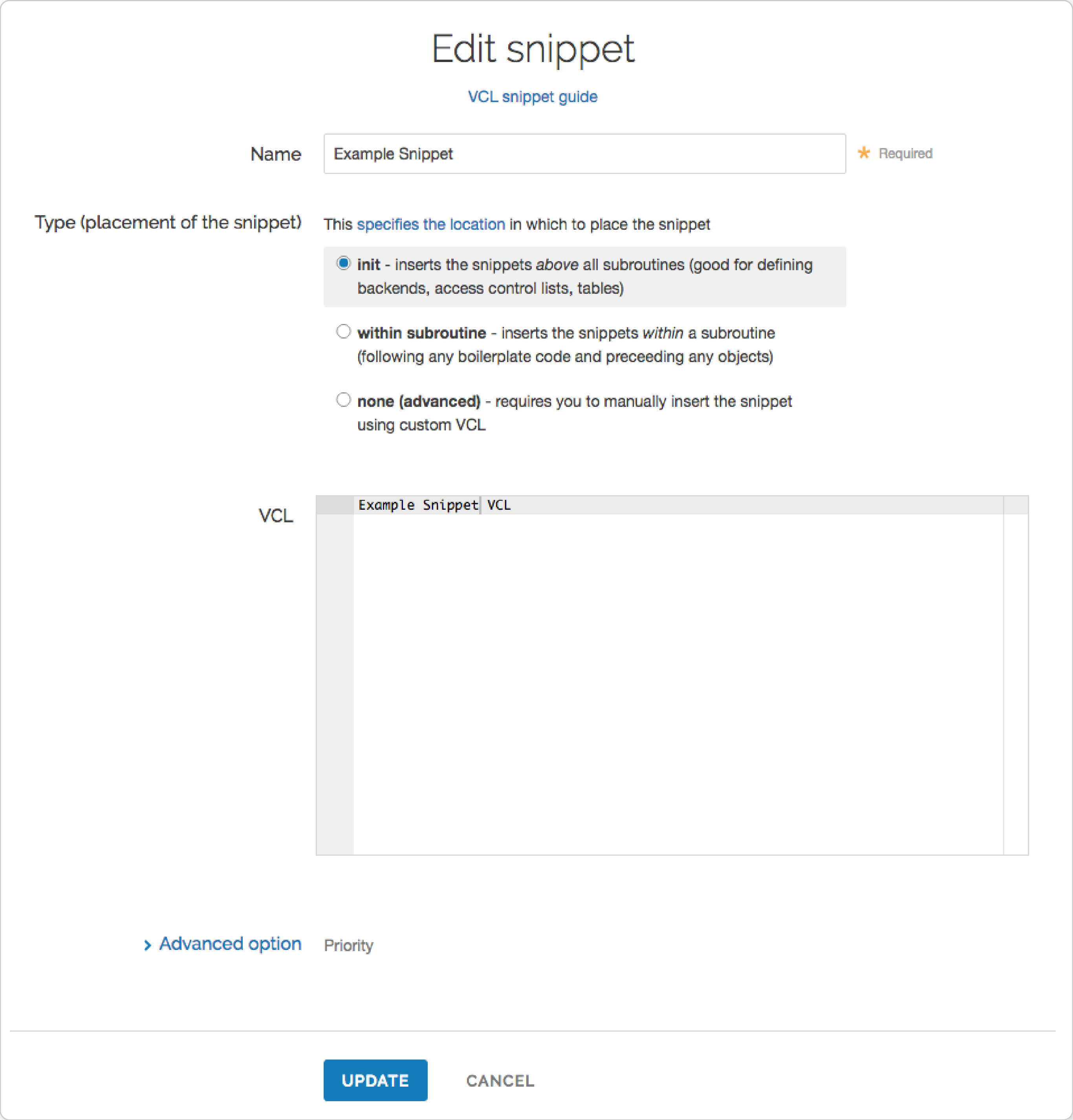
To update an individual snippet via the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click the pencil next to the name of the snippet to be updated.



Priority: 100
Type: init

The Edit snippet page appears.



Edit snippet

VCL snippet guide

Name: Example Snippet ★ Required

Type (placement of the snippet): This specifies the location in which to place the snippet

● **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

○ **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
Example Snippet VCL
```

› Advanced option Priority

6. Update the snippet's settings or VCL as appropriate.
7. Click **Update** to save your changes.

Via the API

To update an individual snippet via the API, make the following API call in a terminal application:

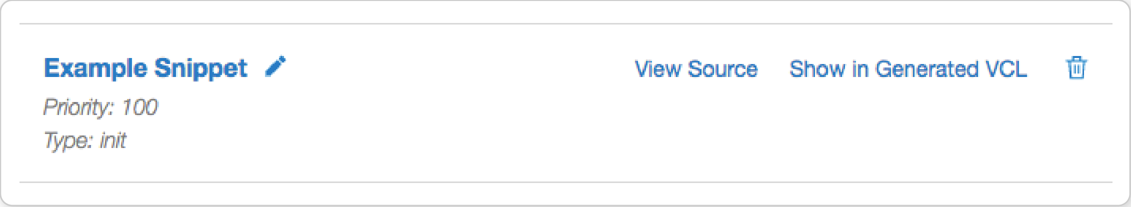
```
$ curl -X PUT -s https://api.fastly.com/service/<Service ID>/version/<Editable Version>/ ip
```

Deleting an existing regular VCL Snippet

You can update existing regular VCL Snippets via the web interface or via the API.

Via the web interface

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click the trash to the right of the name of the snippet to be updated.



Priority: 100
Type: init

6. Click **Confirm and Delete**.

Via the API

To delete an individual snippet via the API, make the following API call in a terminal application:

```
$ curl -X DELETE -s https://api.fastly.com/service/<Service ID>/version/<Editable Versio/s
```

Including regular snippets in custom VCL

Snippets will not be rendered in VCL if you select `none (advanced)` for the snippet type in the web interface or specify a location of `none` for the `type` parameter in the API. This allows you to manually include snippets in custom VCL using the following syntax:

`include "snippet::<snippet name>"`

The same VCL Snippet can be included in custom VCL in as many places as needed.

Example use: location-based redirection

Say that you work at a large content publisher and you want to redirect users to different editions of your publication depending on which country their request comes from. Say also that you want the ability to override the edition you deliver to them based on a cookie.

Using regular VCL snippets, you could add a new object with the relevant VCL as follows:

```
if (req.http.Cookie:edition == "US" || client.geo.country_code == "US") {
  set req.http.Edition = "US";
  set req.backend = F_US;
} elseif (req.http.Cookie:edition == "Europe" || server.region ~ "^EU-" ) {
  set req.http.Edition = "EU";
  set req.backend = F_European;
} else {
  set req.http.Edition = "INT";
  set req.backend = F_International;
}
```

This would create an Edition header in VCL, but allow you to override it by setting a condition. You would add the Edition header into Vary and then add a false condition (e.g., `!reg.url`) to your other backends to ensure the correct edition of your publication gets delivered (Remember: VCL Snippets get added to VCL before backends are set.)

* * *

VCL regular expression cheat sheet

/en/guides/vcl-regular-expression-cheat-sheet

Fastly VCL uses a subset of Perl Compatible Regular Expression (PCRE) syntax. This is case sensitive and forward slashes don't need to be escaped. To disable case sensitivity, add `(?i)` to the start of your expression.

Basic matching

```
req.url == "/phrase"
```

Matches only if `req.url` is exactly `/phrase`.

```
req.url ~ "phrase"
```

Matches `phrase` anywhere.

Matching at the beginning or end of a string

```
req.http.host ~ "^www"
```

Matches if `req.http.host` starts with `www`.

```
req.url ~ "\.jpg$"
```

Matches if `req.url` ends with `.jpg`.

Multiple matches

```
req.url ~ "\.(png|jpg|css|js)$"
```

Matches if `req.url` ends with either `.png`, `.jpg`, `.css`, or `.js`.

```
req.url ~ "\.php(\?.*)?$"
```

Matches if `req.url` ends with `.php`, `.php?foo=bar` or `.php?`, but not `.phpa`.

> **NOTE**
>
> You can also use `req.url.ext` to find the file extension specified in a URL. For example, in the request `www.example.com/1/hello.gif?foo=bar`, `req.url.ext` will contain `gif`.

```
req.url ~ "\.[abc]server$"
```

Matches if `req.url` ends with `.aserver`, `.bserver` or `.cserver`.

Matching wildcards

```
req.url ~ "jp.g$"
```

Matches if `req.url` ends with `jpeg`, `jpag`, and `jp0g`, but doesn't match if `req.url` ends with `jpg`. It also matches if any other character is between the `jp` and the `g`.

```
req.url ~ "jp.*g$"
```

Matches `jp` followed by 0 or more random characters ending with the letter `g` (`jpeg`, `jpg`, and `jpeeeeg` all match).

Capturing matches

```
set req.http.Foo = "abbbccccc";
if (req.http.Foo ~ "^(a+)(b+)(c+)") {
  set resp.http.match0 = re.group.0; # now equals 'abbbccccc'
  set resp.http.match1 = re.group.1; # now equals 'a'
  set resp.http.match2 = re.group.2; # now equals 'bbb'
  set resp.http.match3 = re.group.3; # now equals 'cccccc'
}
```

The `re.group.[0-9]` objects allow you to capture matches. The `re.group.0` object evaluates to the entire matched string even if no capture groups have been supplied. You can use these objects to replace this example:

```
if (req.url ~ "(?i)\?.*some_query_arg=([^&]*)") {
  set req.http.Thing-I-Want = regsub(req.url, "(?i)\?.*some_query_arg=([^&]*).*", "\1")
}
```

You can use `re.group` to greatly simplify the previous example:

```
if (req.url ~ "(?i)\?.*some_query_arg=([^&]*)") {
  set req.http.Thing-I-Want = re.group.1;
}
```

You could even get really fancy and do something like this:

```
set req.http.Thing-I-Want = if(req.url ~ "(?i)\?.*some_query_arg=([^&]*)", re.group.1, ";
```

Replacing content

```
set req.http.host = regsub(req.http.host, "^www\.","");
```

Removes a leading `www.` from the host, if present.

```
set req.http.api-test = regsub(req.http.host, "^www\.","api.");
```

Sets the `api-test` header to contain the host-header, but replaces a leading `www.` with `api.`:

```
Host: www.example.com ->
Host: www.example.com
api-test: api.example.com
Host: example.com ->
Host: example.com
api-test: example.com
```

```
set req.url = regsuball(req.url, "/+", "/");
```

Changes all occurrences of multiple slashes in the URL to a single slash. For example, `//docs///intro.html` will be transformed to `/docs/intro.html`.

* * *

Category: Dictionaries

These articles describe how to move rapid key/value pair decision logic to the edge using dictionaries.

About dictionaries

/en/guides/about-dictionaries

Dictionaries are a type of container that allow you to store data as key-value pairs that can be used in a service without being attached to a single version.

Using dictionaries, you can turn frequently repeated statements like this:

```
if (something == "value1") {
  set other = "result1";
} else if (something == "value2") {
  set other = "result2";
}
```

into a single function that acts as constant, like this:

```
table things {
  "value1": "result1",
  "value2": "result2",
  ...
}

set other = table.lookup(things, something);
```

This allows you to easily make changes to these statements without it being tied to a specific version.

When dictionaries might be useful

- Content sharing and social media outlets updating large referer block lists
- Mobile advertisers validating a key to prevent cache-bust guessing
- Customers authenticating valid user keys at the edge (see private dictionaries)
- Global publishers redirecting users to a specific country site based on geo-location
- Image providers performing token checks for certain objects
- Advertising technology companies blocking bad actors at edge
- Customers deploying web interface versions with simple value change via API

How dictionaries work

Dictionaries are made up of dictionary containers and the dictionary items within them. Once you attach a dictionary container to a version of your service and that service is activated, the data in it becomes *versionless*. This means you can add to and update the data a dictionary contains at any time after it is created, without ever incrementing a service's version.

For example, say you have a referer block list that changes frequently and you want to associate it with a service. Any time that service's configuration changes, especially if the configuration rolls back to a previous version, you would want the block-listed referer domains to continue to remain with the service configuration instead of being removed. Dictionaries would help you do this.

How to create and use dictionaries

To create a dictionary and use it within your service, start by creating an empty dictionary container and then add its entries in a working version of a service that's unlocked and not yet activated. You can create dictionaries:

- via the Dictionaries page.
- via VCL snippets.
- via the Fastly API.

> ✓ TIP
>
> You can create a private dictionary to store dictionary items that can't be listed or read via the web interface or the API.

Limitations and considerations

When creating dictionaries, keep the following things in mind:

- **Dictionaries created with custom VCL cannot be manipulated using the API or the Dictionaries page.** If you create a dictionary container using custom VCL, that dictionary must always be manipulated via custom VCL. Dictionaries uploaded via custom VCL aren't versionless.
- **Dictionary containers, item keys, and their values have specific limits.** Dictionary containers are limited to 1000 items. Dictionary item keys are limited to 255 characters and their values are limited to 8000 characters. If you find your dictionaries approaching these resource limits, contact support. We may be able to help you figure out more efficient ways to do things.
- **Dictionary item keys are case sensitive.** The names of dictionary items are case sensitive. When designing your dictionaries, be sure to take this into account.
- **The contents of dictionaries are stored as VCL.** Personal data should not be incorporated into VCL. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

When making changes to dictionaries, keep the following things in mind:

- **When you delete a dictionary container, you'll only delete it from the service version you're editing.** Dictionary containers are tied to versions and can be cloned and reverted. When using dictionaries, we want you to be able to do things like delete a dictionary container from a current version of your service in order to roll back your configuration to a previous version using as few steps as possible.
- **When you delete a dictionary container, we don't delete the dictionary items inside it.** The dictionary items in a dictionary container are versionless. When you change service versions, we want you to still be able to access the data.
- **Dictionary item deletions are permanent.** Because we don't store data, if you delete a dictionary item, the entry is gone forever from all service versions.
- **Event logs don't exist for dictionary changes.** If you add, update, or remove a dictionary item, there will be no record of it. The only record of a change will exist when you compare service versions to view the point at which the dictionary container was associated with the service version in the first place.

IMPORTANT

Personal information, secrets, or sensitive data should not be included in dictionaries or incorporated into VCL. In addition, we do not maintain version histories of your dictionaries. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

* * *

Working with dictionaries using the web interface

/en/guides/working-with-dictionaries-using-the-web-interface

Dictionaries are a type of container that allow you to store data as key-value pairs that can be used in a service without being attached to a single version. Dictionaries are made up of dictionary containers and dictionary items. You use dictionary items to create and store the key-value pairs, which are then added to a dictionary container. The dictionary container is attached to a service version but can be updated at any time after it is created, without ever incrementing a service's version.

You can work with dictionaries using the Dictionaries page, custom VCL snippets, or Fastly API. You can also create private dictionaries.

Before you begin

Before working with dictionaries, be sure to review our About dictionaries guide to familiarize yourself with how dictionaries work, common use cases, and limitations to using dictionaries.

Working with dictionaries via the Dictionaries page

You can create and interact with dictionaries using the Dictionaries page.

Viewing dictionaries created using the Dictionaries page

To view a dictionary via the Dictionaries page, navigate to the dictionary management area of your service:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Dictionaries** under **Data**. Existing dictionaries, if any, associated with the currently selected service version appear.

NOTE

Remember that dictionary containers are versioned. If you don't see an dictionary attached to your service, check the service version to make sure you're looking at the right one.

Creating a dictionary via the Dictionaries page

Creating a dictionary via the Dictionaries page requires you to create a dictionary container and then create the items that will exist in it.

Creating a dictionary container

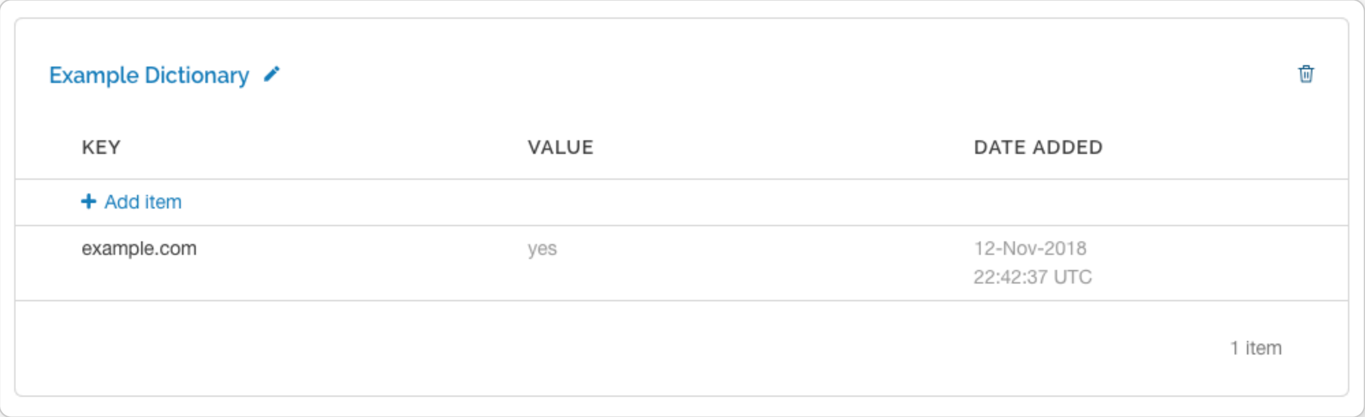
Start by creating a dictionary container using the following steps:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Dictionaries** under **Data**.
5. Click **Create a dictionary**.
6. In the **Name of dictionary** field, enter a descriptive name for the dictionary (e.g., `Example Dictionary`).
7. Click **Add**.
8. Click **Activate** to deploy your configuration changes to the service version you're editing.

Creating a dictionary item

Once you've created a dictionary container, add items into it:

1. Click **Add item**.
2. In the **Key** field, enter the unique identifier for some item of data (e.g., `example.com`).
3. In the **Value** field, enter the value associated with the unique identifier (e.g., `yes`)
4. Click **Add**. The key-value pair appears in the dictionary container. This addition will become effective immediately.



Example Dictionary

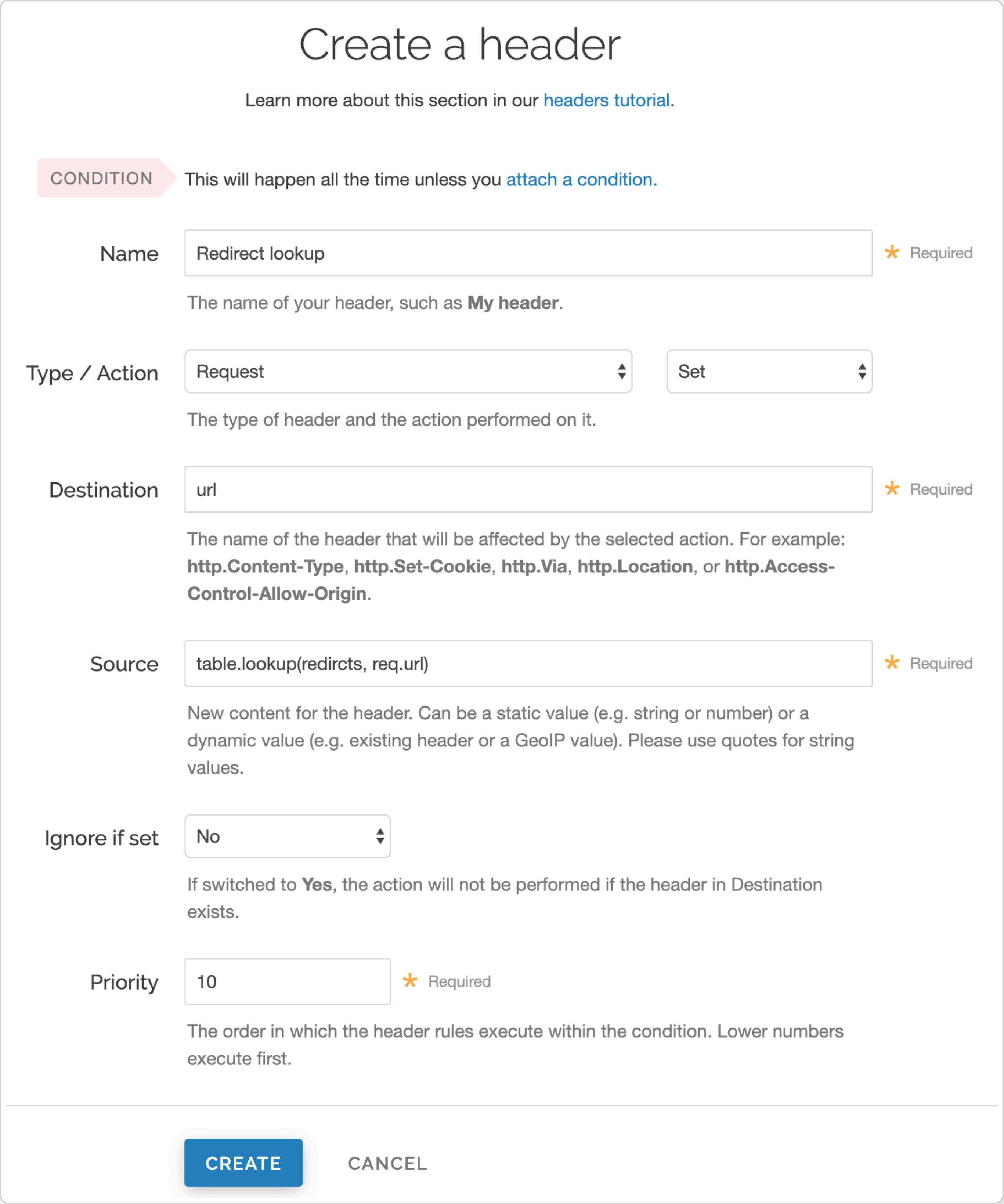
KEY	VALUE	DATE ADDED
+ Add item		
example.com	yes	12-Nov-2018 22:42:37 UTC

1 item

Using a dictionary

Once you've created a dictionary, you can start using it. To use a dictionary, you'll need to create and add a header. Follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Redirect lookup

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: url

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: table.lookup(redircts, req.url)

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

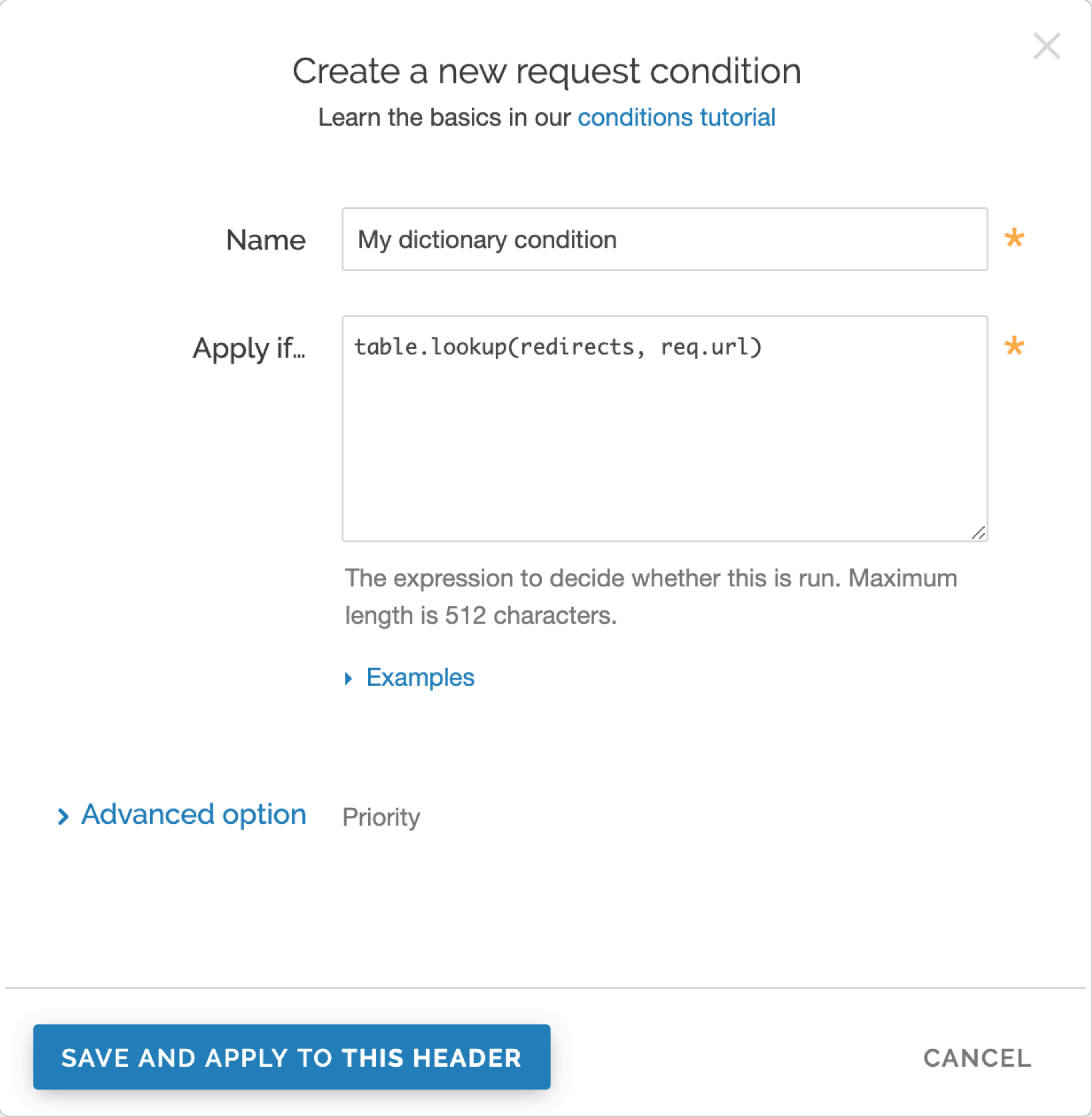
6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter the name of your header rule (for example, `Redirect lookup`).
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter the name of the header affected by the selected action (e.g., `url`).

  - In the **Source** field, enter where the content for the header comes from (e.g., `table.lookup(redirects, req.url)`).
  - From the **Ignore if set** menu, select **No**
  - Leave the **Priority** field as is.

7. Click **Create** to create the header. A new header appears on the Content page.

Once you've created a header, you can create a condition to specify when to use the dictionary:

1. Click **Attach a condition** next to new header you just created.
2. Click **Create a new request condition**.



Create a new request condition

Learn the basics in our conditions tutorial

Name: My dictionary condition

Apply if...: table.lookup(redirects, req.url)

The expression to decide whether this is run. Maximum length is 512 characters.

3. In the **Name** field, enter a name for your condition (e.g., `My dictionary condition`)

4. In the **Apply if** field, enter a condition (e.g., `table.lookup(redirects, req.url)`)
5. Click **Save and apply to**.
6. Click **Activate** to deploy your configuration changes.

Editing a dictionary container via the Dictionaries page

You can edit the name of a dictionary container that was created via the Dictionaries page in any unlocked service version:

1. Find a dictionary associated with an unlocked version of your service.
2. Click the pencil next to the dictionary container name.
3. Change the name, then click **Save**.

Editing a dictionary item via the Dictionaries page

You can edit the dictionary items within a container at any time. To edit the key-value pair in a dictionary container that was created via the Dictionaries page:

1. Find any dictionary associated with your service in which the key-value pairs appear. Because dictionary items are versionless, the service version you choose doesn't matter. Choose the one that makes the most sense to you.
2. Hover your cursor over a dictionary item, then click the pencil that appears.
3. Edit the key or value as necessary.
4. Click **Save**. The changes you make will be immediately applied to your configuration. If your dictionary container has already been associated with a deployed service version, those changes will happen live.

Deleting a dictionary via the Dictionaries page

Keeping in mind their limitations, dictionary containers and the items within them can be deleted via the Dictionaries page.

Deleting a dictionary container

You can delete a dictionary container that was created via the Dictionaries page in any unlocked service version:

1. Find a dictionary associated with an unlocked version of your service.
2. Click the trash in the top right corner of the dictionary.
3. Click **Confirm and delete**.
4. Click **Activate** to deploy your configuration changes to the service version you're editing.

Deleting a dictionary entry

You can delete the dictionary entries within a container at any time. To delete a key-value pair included in a dictionary container that was created via the Dictionaries page:

1. Find any dictionary associated with your service in which the key-value pairs appear. Because dictionary items are versionless, the service version you choose doesn't matter. Choose the one that makes the most sense to

you.

2. Hover your cursor over a dictionary item, then click the trash that appears.
3. Click **Confirm and delete**.

Working with dictionaries using VCL snippets

You can create and interact with dictionaries using custom VCL snippets. Dictionaries created with custom VCL cannot be manipulated using the API or the Dictionaries page. If you create a dictionary container using custom VCL, that dictionary must always be manipulated via custom VCL. Dictionaries uploaded via custom VCL aren't versionless.

Viewing dictionaries created by VCL snippets

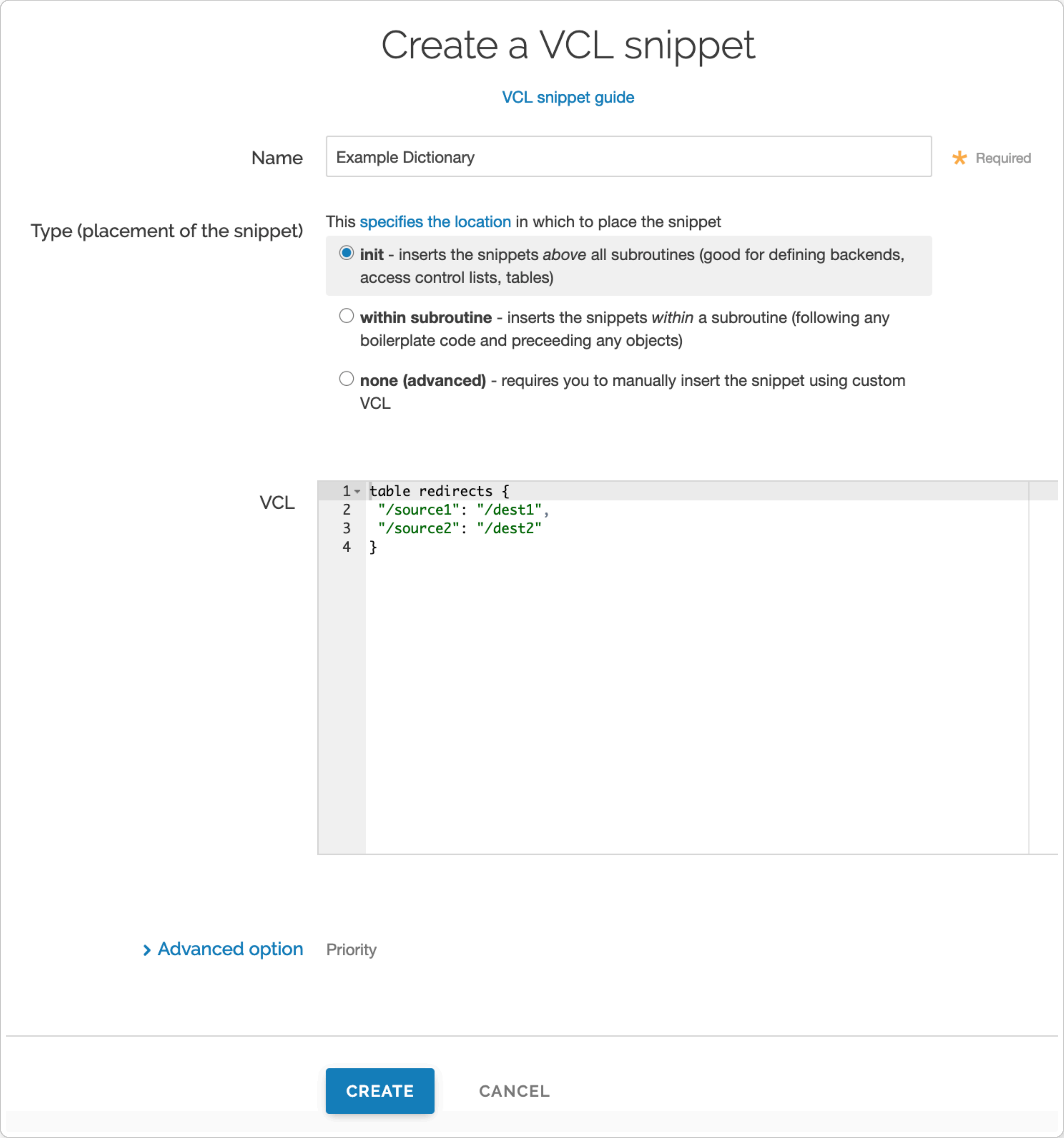
To view a dictionary under the VCL Snippets link, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. To view the contents of the dictionary, click **View source**.

Creating a dictionary using VCL snippets

To create a dictionary using VCL snippets, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL snippets**.
5. Click **Create snippet**.



Create a VCL snippet

VCL snippet guide

Name: Example Dictionary ★ Required

Type (placement of the snippet): This specifies the location in which to place the snippet

● **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

○ **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
table redirects {
 "/source1": "/dest1",
 "/source2": "/dest2"
}
```

6. In the **Name** field, enter an appropriate name (e.g., `Example Dictionary`).

7. From the **Type (placement of the snippet)**, select **init**.

8. In the **VCL** field, create a table and add key-value pairs. For example, if you want to create a table that redirects a URL to another path:

```
table redirects {
  "/source1": "/dest1",
  "/source2": "/dest2"
}
```

where the table is a set of key-value pairs that you can reference in your code. You can replace the contents of this table with different key-value pairs.

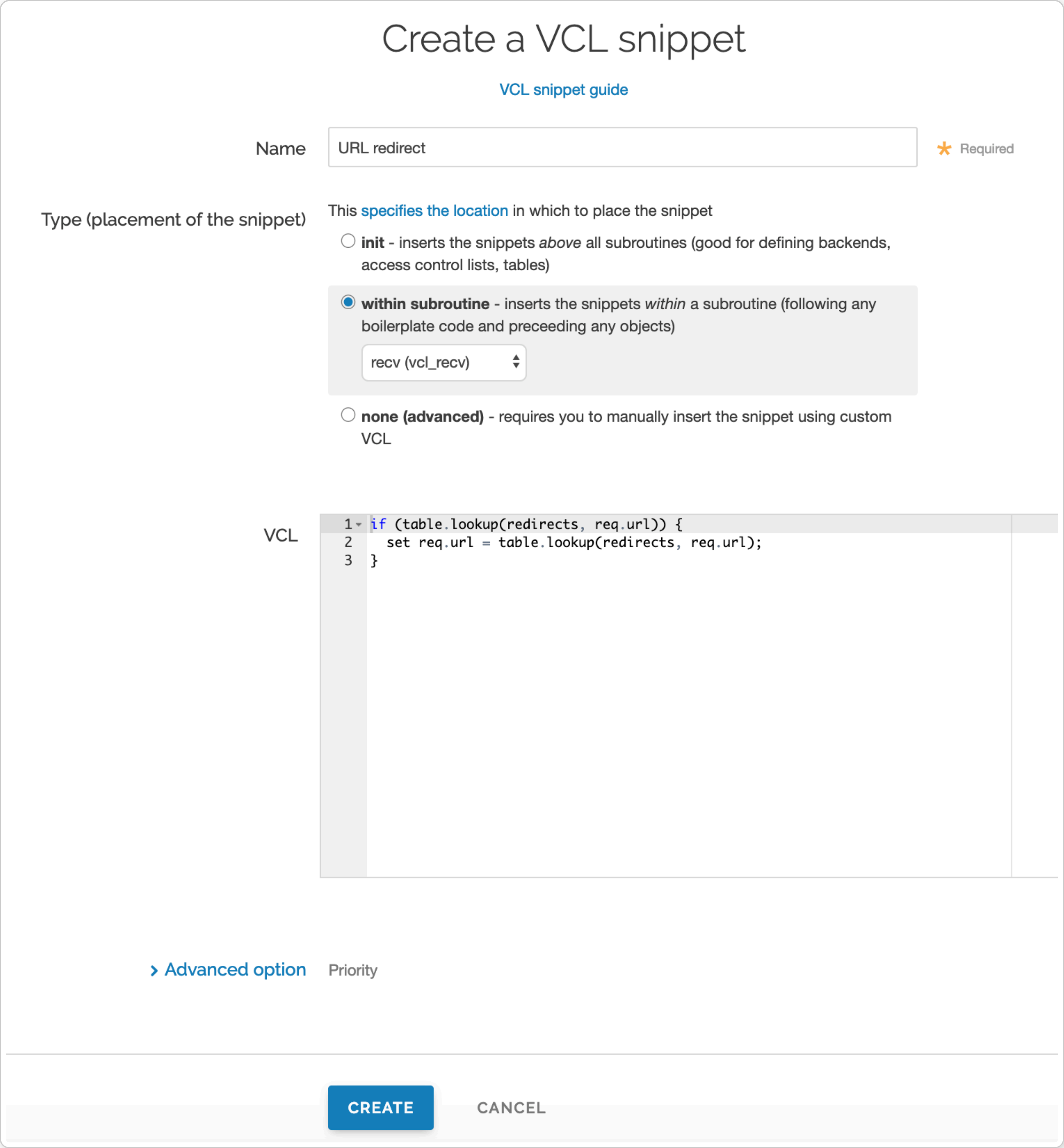
9. Click **Create** to create the snippet.

Using a dictionary with VCL snippets

Once you've created a dictionary, you can start using it.

To start using your dictionary with VCL Snippets, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL snippets**.
5. Click **Create snippet**.



Create a VCL snippet

VCL snippet guide

Name: URL redirect
* Required

Type (placement of the snippet): This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

recv (vcl_recv)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
if (table.lookup(redirects, req.url)) {
  set req.url = table.lookup(redirects, req.url);
}
```

6. Fill out the Create a VCL snippet fields as follows:
   - In the **Name** field, enter an appropriate name (e.g., `URL redirect`).
   - From the **Type (placement of the snippet)** controls, select **within subroutine**.
   - From the **Select subroutine** menu, select **recv (`vcl_recv`)**.
   - In the **VCL** field, add a condition to use the table you created in Creating a dictionary using VCL Snippets. For example, if you need to rewrite your URL destination, you could use:

```
if (table.lookup(redirects, req.url)) {
    set req.url = table.lookup(redirects, req.url);
}
```

where `table.lookup` checks the dictionary for the contents you want. The first parameter is the table being looked in and the second parameter is the key you're looking for. If the key exactly matches the second parameter, that value is returned. Be aware that regex doesn't work with a dictionary lookup.

7. Click **Create** to create the snippet.
8. Click **Activate** to deploy your configuration changes.

Editing a dictionary using VCL snippets

You can edit the dictionary name and the condition that was created using VCL Snippets in any unlocked service version:

1. Find a dictionary associated with an unlocked version of your service.
2. Click the pencil next to the dictionary. You can now make changes to the name and the condition.
3. Click **Update** once you’re finished with your changes.
4. Click **Activate** to activate the version you made the edits in and view the changes in your VCL.

Deleting a dictionary using VCL snippets

You can delete a dictionary using VCL Snippets by following the steps below. Keep in mind the limitations involved when deleting a dictionary.

1. Find a dictionary associated with an unlocked version of your service.
2. Click the trash on the top right corner of the snippet
3. Click **Confirm and delete**.
4. Click **Activate** to deploy your configuration changes to the service version you're editing.

Private dictionaries

Private dictionaries store dictionary items that can't be listed or read via the web interface or the API.

Limitations and considerations

When creating private dictionaries, keep the following things in mind:

- You can create, read, update, and delete a private dictionary. Private dictionaries can only be created via the API.
- You cannot update the `write_only` attribute of a dictionary after it has been created.
- You can create, update, and delete items that belong to a private dictionary. After you've activated a service version, you can only update the values of existing private dictionary items via API. You can add new items to a private dictionary via the web interface or the API.
- You cannot view items that belong to a private dictionary via the API. After you've activated a service version, you cannot use the web interface to view items that belong to a private dictionary.

- Depending on how your service is configured, data stored in private dictionaries can be sent in headers and to log streaming endpoints.

> WARNING
>
> To edit or delete dictionary items in a private dictionary, you'll need to remember the keys of the dictionary items.

Creating a private dictionary container

To use a private dictionary container, start by creating an empty one within an unlocked version of a service.

Before a private dictionary can be manipulated, its private dictionary container must be associated with at least one service version that is not locked and not active so that the service becomes aware of the private dictionary's existence.

For example, if you were creating a `my_example_dictionary` private dictionary via the API, you would make an API call by running this command:

```
$ curl -X POST -H 'Fastly-Key: FASTLY_API_TOKEN' -d 'name=my_example_dictionary&write_on=t
```

which would return:

```
{
  "created_at": "2017-05-03T16:11:41+00:00",
  "deleted_at": null,
  "id": "<dictionary_id>",
  "name": "my_example_dictionary",
  "service_id": "<service_id>",
  "updated_at": "2017-05-03T16:11:41+00:00",
  "version": <version_number>,
  "write_only": true
}
```

You can start adding dictionary items after you've created the dictionary. Don't forget to activate the service when you're finished.

Viewing private dictionaries in VCL

The contents of private dictionaries are hidden in VCL. The private dictionary's metadata is displayed, as shown below:

```
table my_example_dictionary {
     # REDACTED dictionary content
     # last_updated: 2017-10-16 20:44:43
     # item_count: 2
     # digest: 8f92141234567890da30ba9cea7d98ef614
 }
```

What's next

Explore custom VCL examples for using dictionaries on Developer Hub.

IMPORTANT

Personal information, secrets, or sensitive data should not be included in dictionaries or incorporated into VCL. In addition, we do not maintain version histories of your dictionaries. Our Compliance and Law FAQ describes in detail how Fastly handles personal data privacy.

* * *

Category: Domains and origins

These articles describe configuration settings and changes you can make to your domains and origins when setting up Fastly services.

Automatic load balancing

/en/guides/automatic-load-balancing

This guide describes how to automatically load balance between two or more origin servers. Load balancing distributes requests across multiple servers to optimize resource use and avoid overloading any single resource.

Before you begin

Before you configure load balancing, keep in mind the following:

- To prevent errors when shielding is enabled, all backends in the automatic load balancing group must use the same shielding location.
- Conditions on your origin server can directly change how automatic load balancing behaves. Be sure to review conditions behavior to ensure automatic load balancing works properly.
- Many customers configure failover at the same time they configure load balancing functionality. Our guide on configuring failover can show you how.

Enabling load balancing

To enable load balancing across two or more origin servers, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.

5. Click the name of the Host you want to edit.
6. From the **Auto load balance** menu, select **Yes**.
7. In the **Weight** field, enter the percentage of the total traffic to send to the origin server.

> TIP
>
> When you specify a whole number in the **Weight** field, you specify the percentage of the total traffic to send to a specific origin server. Each origin server receives the percentage (/) of the total traffic equal to the number you specify. For example, if you have two origin servers, A and B, setting the weight to 50 on both splits the traffic between them equally. Each origin server receives 50 percent of your total traffic. If you increase the weight on origin server A to 55 and decrease the weight on origin server B to 45, the percentage of traffic changes to 55 percent and 45 percent respectively.

8. Click **Update**.
9. Repeat steps 5, 6, 7, and 8 for each origin server you want to include in the automatic load balancing group.

> NOTE
>
> Each Fastly service can be configured with up to five origin servers. Contact sales@fastly.com to enable more than five origin servers per service in your account.

10. Click **Activate** to deploy your configuration changes.

Using conditions with load balancing

You can set conditions on origin servers to change how load balancing works. The load balancing autodirector groups servers together based on like conditions, giving you the flexibility to effectively create subsets of the autodirector by assigning a condition to one group of origins and another condition to another set of origins. If each group of origin servers has a different condition that affects load balancing, the auto load function will not randomly load balance between the different servers.

What's next

For more advanced load balancing scenarios, refer to our load balancing documentation.

* * *

Changing connection timeouts to your origin

/en/guides/changing-connection-timeouts-to-your-origin

Connection timeouts to your origin server control how long Fastly will wait for a response from your origin server before exiting with an error. Changing the connection timeout is a good way to start troubleshooting 503 backend read errors. Follow the steps below to change the connection timeouts to your origin server:

1. Log in to the Fastly web interface.

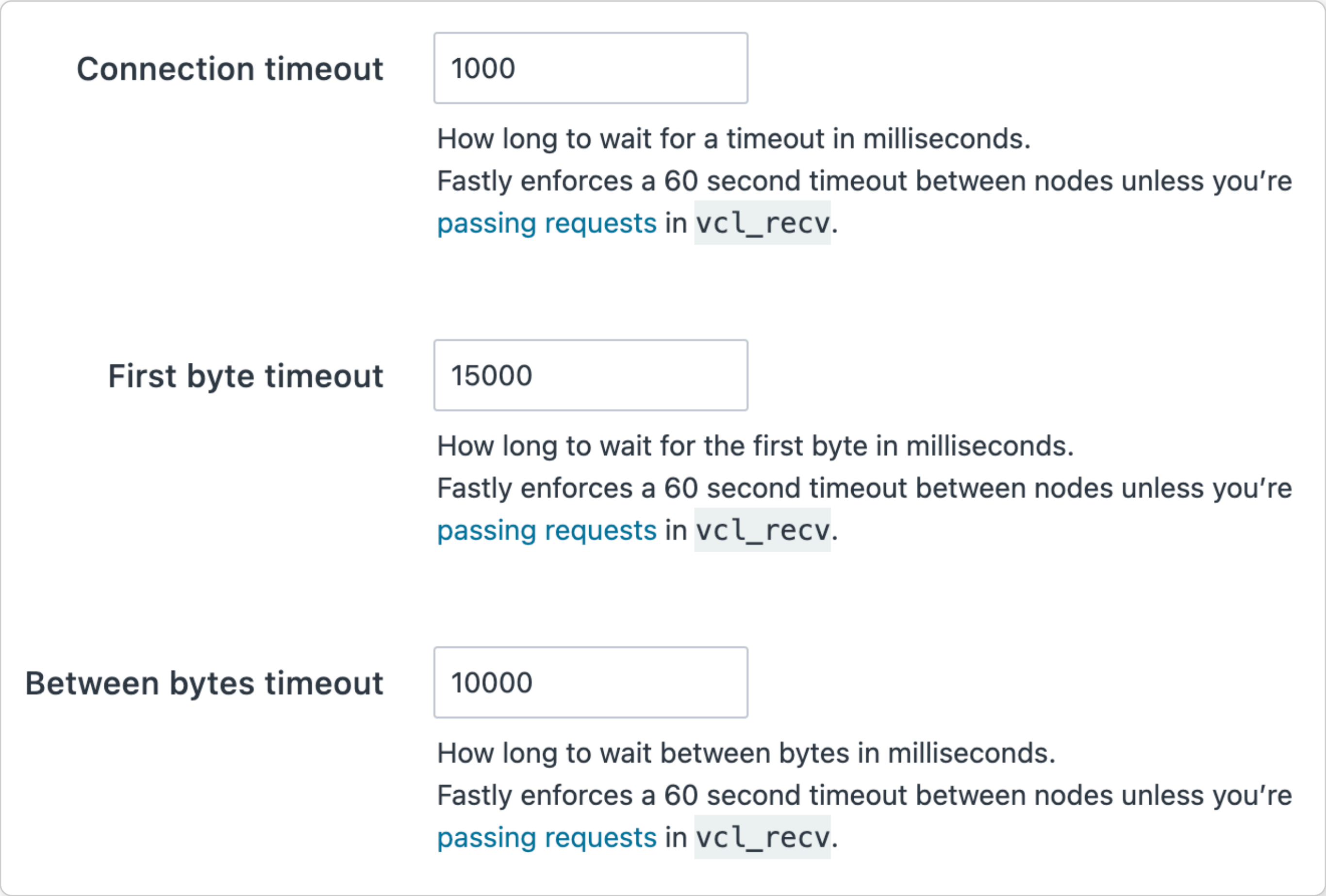
2. From the Home page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > CDN Services or **Compute** > Compute Services to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Origins**.

5. Click the name of the Host that you want to edit.

6. Click **Advanced options**.



Connection timeout: 1000

How long to wait for a timeout in milliseconds.
Fastly enforces a 60 second timeout between nodes unless you're passing requests in `vcl_recv`.

First byte timeout: 15000

How long to wait for the first byte in milliseconds.
Fastly enforces a 60 second timeout between nodes unless you're passing requests in `vcl_recv`.

Between bytes timeout: 10000

How long to wait between bytes in milliseconds.
Fastly enforces a 60 second timeout between nodes unless you're passing requests in `vcl_recv`.

7. Type the new timeout in the appropriate field of the **Timeouts** section.

8. Click **Update**.

* * *

Changing origins based on user location

/en/guides/changing-origins-based-on-user-location

Fastly allows you to change origin servers based on the user's geographic location. This is useful when you need to serve different content to users who are in different locations. For example, you could change origin servers to serve a restricted version of your website to users in a different country.

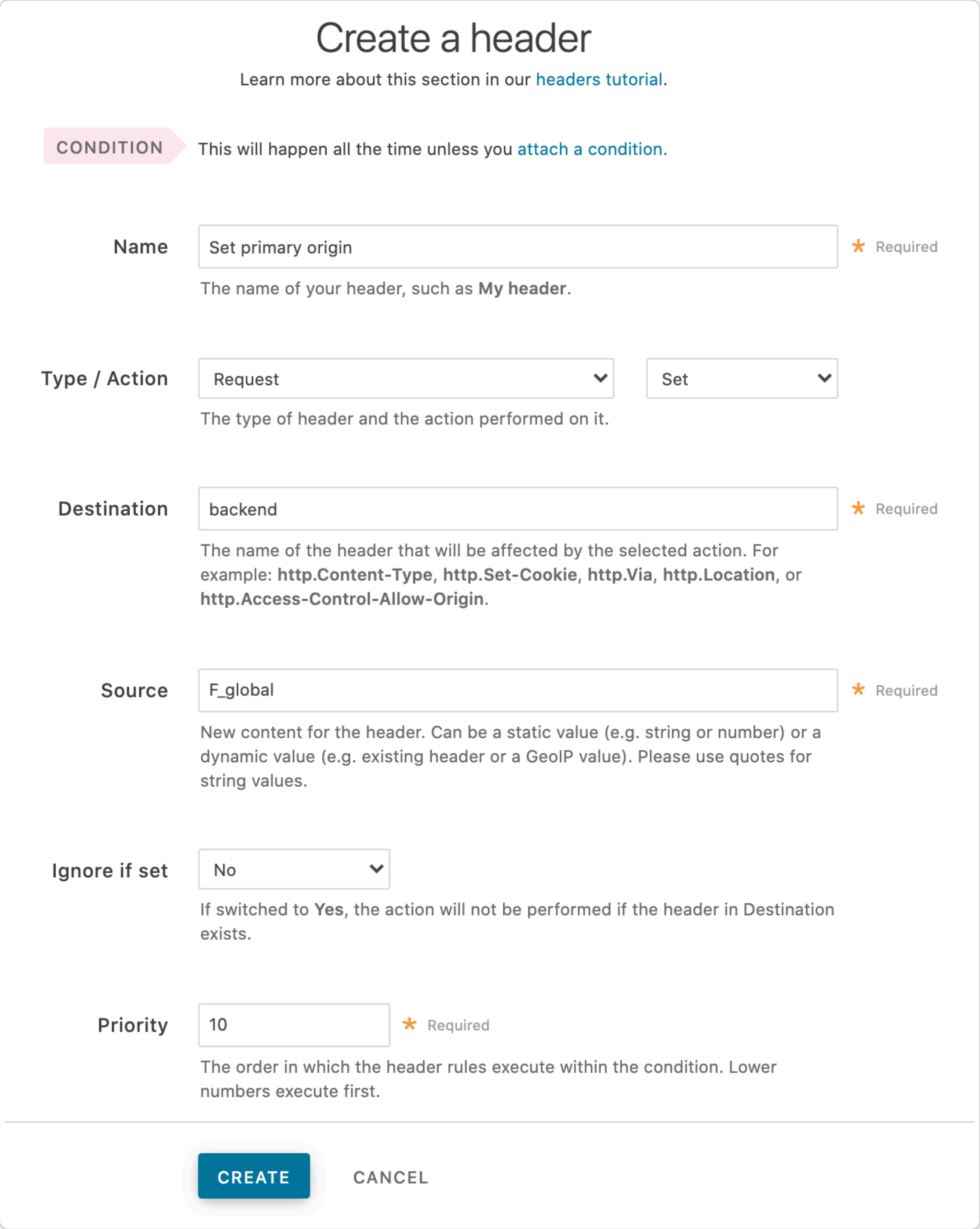
Using the web interface

You can use the web interface to create the headers and the condition.

Creating the header for the default origin server

First, create a header for the default origin server to serve content to the majority of users. Follow these instructions to create the header:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Set primary origin

The name of your header, such as **My header**.

Type / Action: Request, Set
The type of header and the action performed on it.

Destination: backend

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: F_global

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No
If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter the name of your header rule (for example, `Set default origin`).
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.

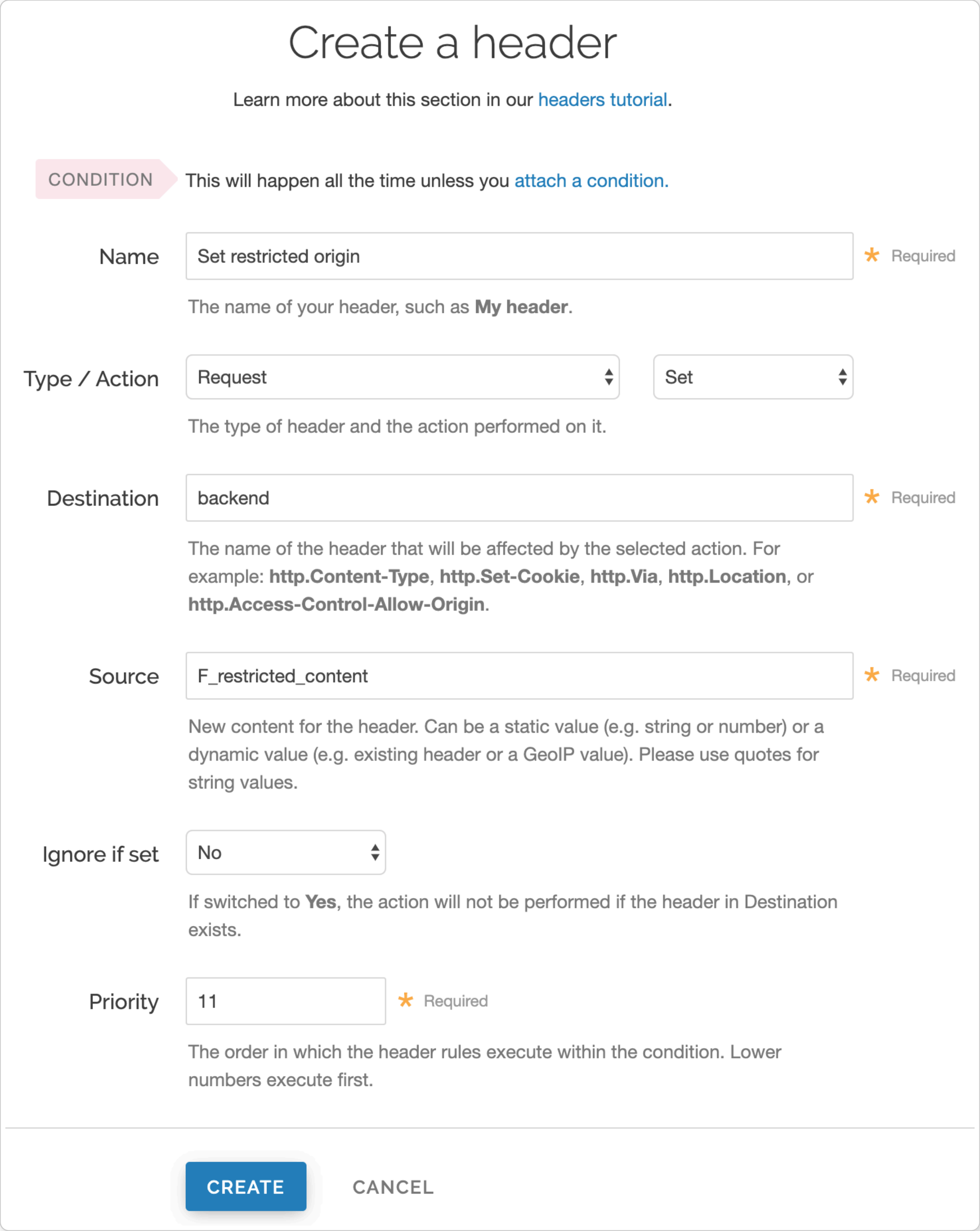
  - In the **Destination** field, enter `backend`.
  - In the **Source** field, enter the name of the origin server you want to serve content to the majority of users (here it's `F_global`). Preview the VCL to find the name of the origin server.
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**.

Creating the header for the restricted origin server

Now, create a header for the restricted origin server to serve content to the users residing in the countries specified in the condition. Follow these instructions to create the header:

1. Click **Content**.
2. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Set restricted origin
The name of your header, such as **My header**.

Type / Action: Request / Set
The type of header and the action performed on it.

Destination: backend
The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: F_restricted_content
New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No
If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 11
The order in which the header rules execute within the condition. Lower numbers execute first.

3. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter the name of your header rule (for example, `Set restricted origin`).
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.

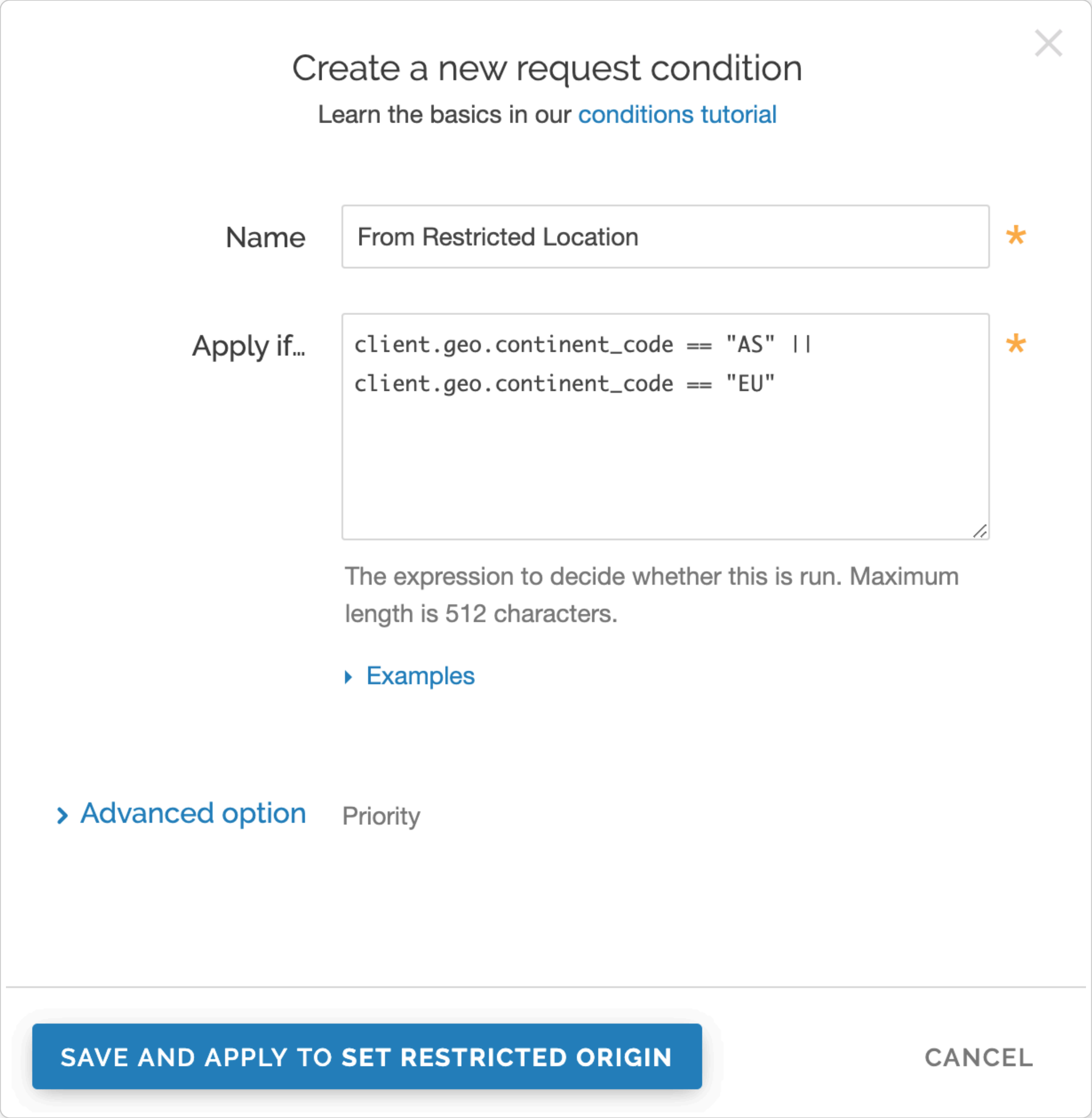
- In the **Destination** field, enter `backend`.
- In the **Source** field, enter the name of the restricted origin server you want to serve content to the users residing in the countries specified in the condition (here it's `F_restricted_content`). Preview the VCL to find the name of the origin server.
- From the **Ignore if set** menu, select **No**.
- In the **Priority** field, enter `11`.

4. Click **Create**.

Creating a condition for the restricted origin header

Finally, create a condition for the restricted origin header. The condition checks the geolocation header. If the user's geolocation matches a location specified in the condition, Fastly uses the restricted origin server. Follow these instructions to create the condition:

1. Click **Content**.
2. In the Headers section, click **Attach a condition** next to the **Set restricted origin** header.


Create a new request condition
Learn the basics in our conditions tutorial
Examples
Advanced option
SAVE AND APPLY TO SET RESTRICTED ORIGIN
CANCEL

Name: From Restricted Location

Apply if...

```
client.geo.continent_code == "AS" ||
client.geo.continent_code == "EU"
```

The expression to decide whether this is run. Maximum length is 512 characters.

Priority

3. Fill out the **Create a new request condition** fields as follows:
    - In the **Name** field, enter a descriptive name for the new condition (for example, `From Restricted Location`).
    - In the **Apply if** field, enter a request condition. For example, to send all users in Asia and Europe to the restricted origin server, enter `client.geo.continent_code == "AS" || client.geo.continent_code == "EU"`. See Geolocation-related VCL features for more information.
4. Click **Save and apply to**.
5. Click **Activate** to deploy your configuration changes.

Using custom VCL

If you'd prefer not to use the web interface, you can use custom VCL to configure your service to change origin servers based on the user's geographic location. Use the following VCL as a starting point:

```
# default conditions
  set req.backend = F_global;

  # Use restricted content if the user is in Asia, France or Germany
  if (client.geo.continent_code == "AS" || client.geo.country_code == "FR" || client.ge
    set req.backend = F_restricted_content;
  }
```

* * *

Failover configuration

/en/guides/failover-configuration

This guide describes how to configure failover origin servers. Failover (backup) servers ensure you can maintain availability of your content if your primary server is not available.

Before you begin

To configure failover origin servers you must make sure you have health checks configured for your primary server. If you configure your failover servers but don't configure health checks on the primary server, the failover won't work properly if your primary server stops responding.

Configuring a failover origin server

Once you've confirmed health checks are configured, you must:

1. Enable automatic load balancing on all primary origin servers and any servers that will become your failover.
2. Attach a condition to the failover server that specifies exactly when to use is as a backup.

Enable automatic load balancing

To configure a failover origin server, turn on automatic load balancing for your primary and failover origin servers by following the steps in our guide on configuring load balancing.

Specify when to use the failover server

Once you've configured your primary and failover servers, attach a condition that specifies exactly when the failover servers should be used. The settings for the condition differ depending on whether you have a single primary origin server or multiple primary origin servers.

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. In the **Hosts** area, find your failover origin server and click **Attach a condition**.
6. Click **Create a new request condition**.
7. In the **Name** field, enter the name of your request condition (for example, `Primary origin down`).
8. In the **Apply if** field, enter the conditions under which to use the failover server:
    - If you have multiple primary servers, enter the following in the **Apply if** field: `backend.autodirector_.healthy == false`.
    - If you have a single primary server, enter the following in the **Apply if** field: `backend.{name of primary server}.healthy == false`.

    > NOTE
    >
    > Preview the VCL to find the name of the primary server. Typically the name is prefixed with `F_` and spaces are replaced with underscores. For example, a backend called `Host 1` has the VCL name `F_Host_1`.
9. Click **Save and apply**.
10. For any other failover origin servers, click **Attach a condition** and select the condition you already created to apply it.

* * *

IPv6 support

/en/guides/ipv6-support

Fastly has integrated IPv6 into its technology stack. By enabling IPv6, visitors on IPv6 connections can access your websites and applications. This can be done without any changes to your backend infrastructure.

Enabling IPv6

To enable traffic to be served over IPv6 and IPv4 addresses, follow the instructions below appropriate to your CNAME record.

If you're using one of our TLS products, you can find your CNAME records in the **HTTPS and network** tab, under **DNS details**. If the name ends in `map.fastly.net`, it is a customer-specific hostname. Otherwise, it is a Fastly-shared hostname.

> NOTE

Fastly doesn't support IPv6 connections to origin servers.

Switching to dualstack for Fastly-shared hostnames

You can enable IPv6 dualstack (IPv4 and IPv6) functionality for your hostname by prefixing your CNAME record with `dualstack`. For example, if you have traffic on an IP address pool nicknamed `j.sni` (which supports a minimum TLS version of 1.2, a maximum TLS version of 1.3, does not support 0RTT, and offers HTTP/2), then you could use the following dualstack options:

- `dualstack.j.sni.global.fastly.net` (dualstack global map)
- `dualstack.j.sni.us-eu.fastly.net` (dualstack NA/EU)

If you are using `m.ssl` (which supports a minimum and maximum TLS version of 1.2 and HTTP/1.x only), then you could use the following dualstack options:

- `dualstack.m.ssl.global.fastly.net` (dualstack global map)
- `dualstack.m.ssl.us-eu.fastly.net` (dualstack NA/EU)

✓ TIP

For more information on updating your CNAME record, see our instructions on updating your CNAME record with your DNS provider.

Enabling IPv6 on customer-specific hostnames

If you have a customer-specific hostname, contact support and we'll provide you with a parallel IPv6 map or enable dualstack on your current one. By default, maps will be HTTP/2 enabled and have a global billing region set. Be sure to specify any required changes when having a new map created.

Enabling Anycast IPv6 addresses for apex domains

If you use our Anycast IPv4 addresses for apex domains, contact Fastly support and we'll provide you with the appropriate Anycast IPv6 addresses.

Geolocation features for IPv6

Fastly's geolocation features work with IPv6 addresses.

VCL variable

You can track whether a request came in as an IPv6 request with the `req.is_ipv6` VCL variable as well as by the IPv6 format itself when logging `%h`.

Testing IPv6

Once you're up and running with IPv6, test IPv6 by entering a dig command in a terminal application to make sure your map returns AAAA records. For example, you can type something similar to this:

```
$ dig www.example.com AAAA +short
```

where `www.example.com` is the domain that you're testing.

Your output should appear similar to the following:

`2606:2800:220:1:248:1893:25c8:1946`

You can also use a tool like What's my DNS and choose the AAAA option to see how clients around the world are resolving to your CNAME record.

Performance implications

Enabling IPv6 shouldn't negatively impact performance. Most modern clients implement an approach called Happy Eyeballs to connect over either IPv4 or IPv6, whichever is faster. Happy Eyeballs chooses IPv6 over IPv4 when all else is equal.

* * *

Maintaining separate HTTP and HTTPS requests to origin servers

/en/guides/maintaining-separate-http-and-https-requests-to-backend-servers

---

It is common to use the same origin web application to serve both HTTP and HTTPS requests and let the application determine which actions to take to secure communications depending on the incoming protocol. Fastly allows users to set this up to preserve this functionality within their servers. To set Fastly up to send HTTP requests to the non-secure service and HTTPS requests to the secure service, configure two origins, one each for the secure and non-secure ports, then set up the conditions under which requests will be sent there.

Create multiple origins

Begin by configuring the same origin address with a different port as a separate origin server. Follow the instructions for working with hosts. You'll add specific details about the non-secure server (port `80`) when you fill out the **Create a host** fields:

- In the **Name** field, enter a name for the non-secure server (for example, `Server Name (plain)`).
- In the **Address** field, enter the address of the non-secure server (for example, `server.example.com`).
- In the **Transport Layer Security (TLS)** section, set **Enable TLS?** to **No**.

Follow the instructions for working with hosts to create another origin server, this time for your secure server. You'll add specific details about the secure server (port `443`) when you fill out the **Create a host** fields:

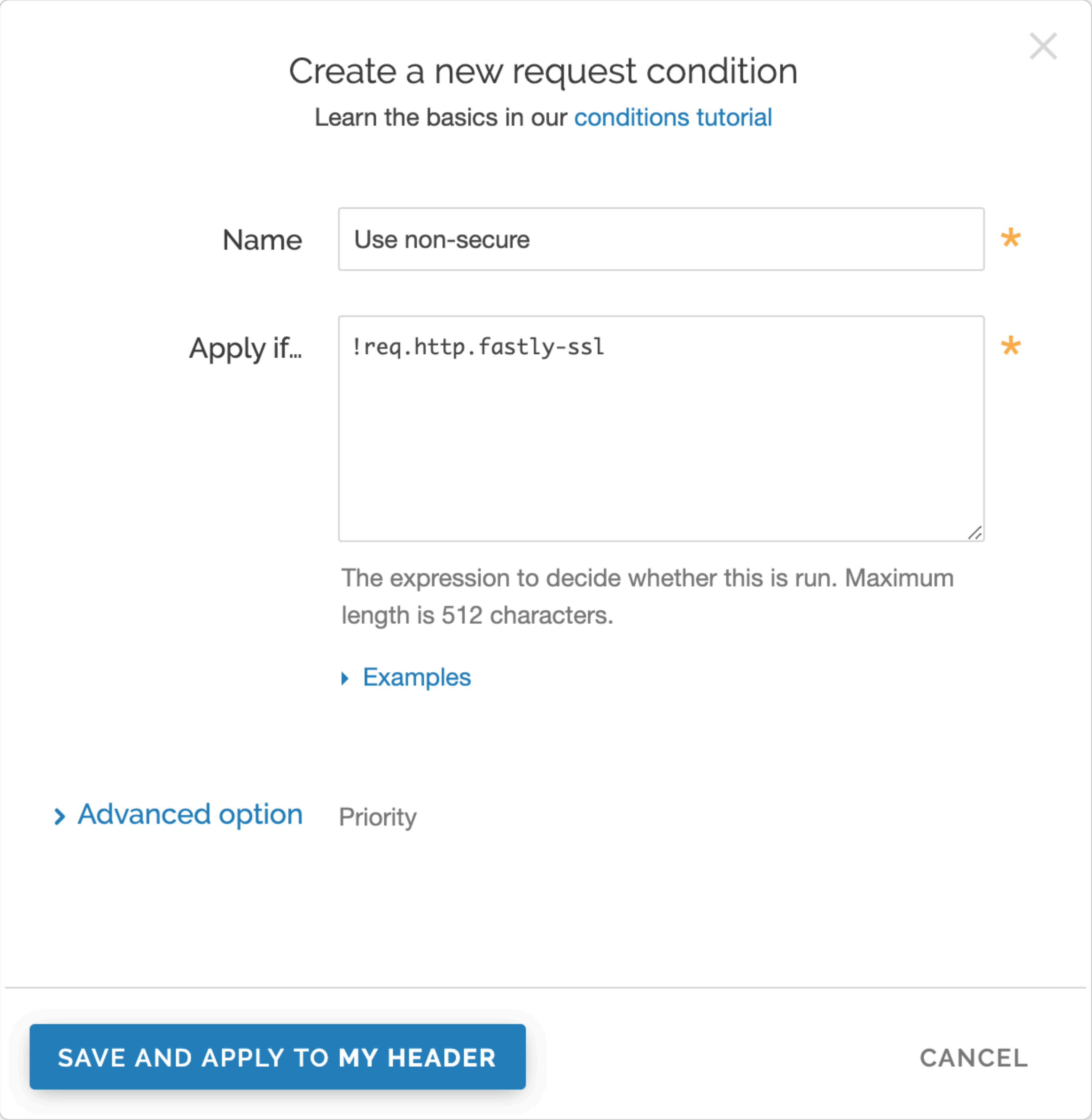
- In the **Name** field, enter a name for the non-secure server (for example, `Server Name (secure)`).
- In the **Address** field, enter the address of the non-secure server (for example, `server.example.com`).
- In the **Transport Layer Security (TLS)** section, leave the **Enable TLS?** default set to **Yes**.

Conditionally send traffic to origins

To conditionally determine which server receives secure and non-secure requests, Fastly relies on the presence or absence of a specific header when the backend is selected. When an incoming connection is received over TLS, Fastly sets the `req.http.fastly-ssl` header to determine which server to use.

Set a condition for this header on each origin by following the steps below.

1. On the **Origins** page, click **Attach a condition** next to the name of the non-secure server.



Create a new request condition
Learn the basics in our conditions tutorial

Name: Use non-secure

Apply if...: !req.http.fastly-ssl

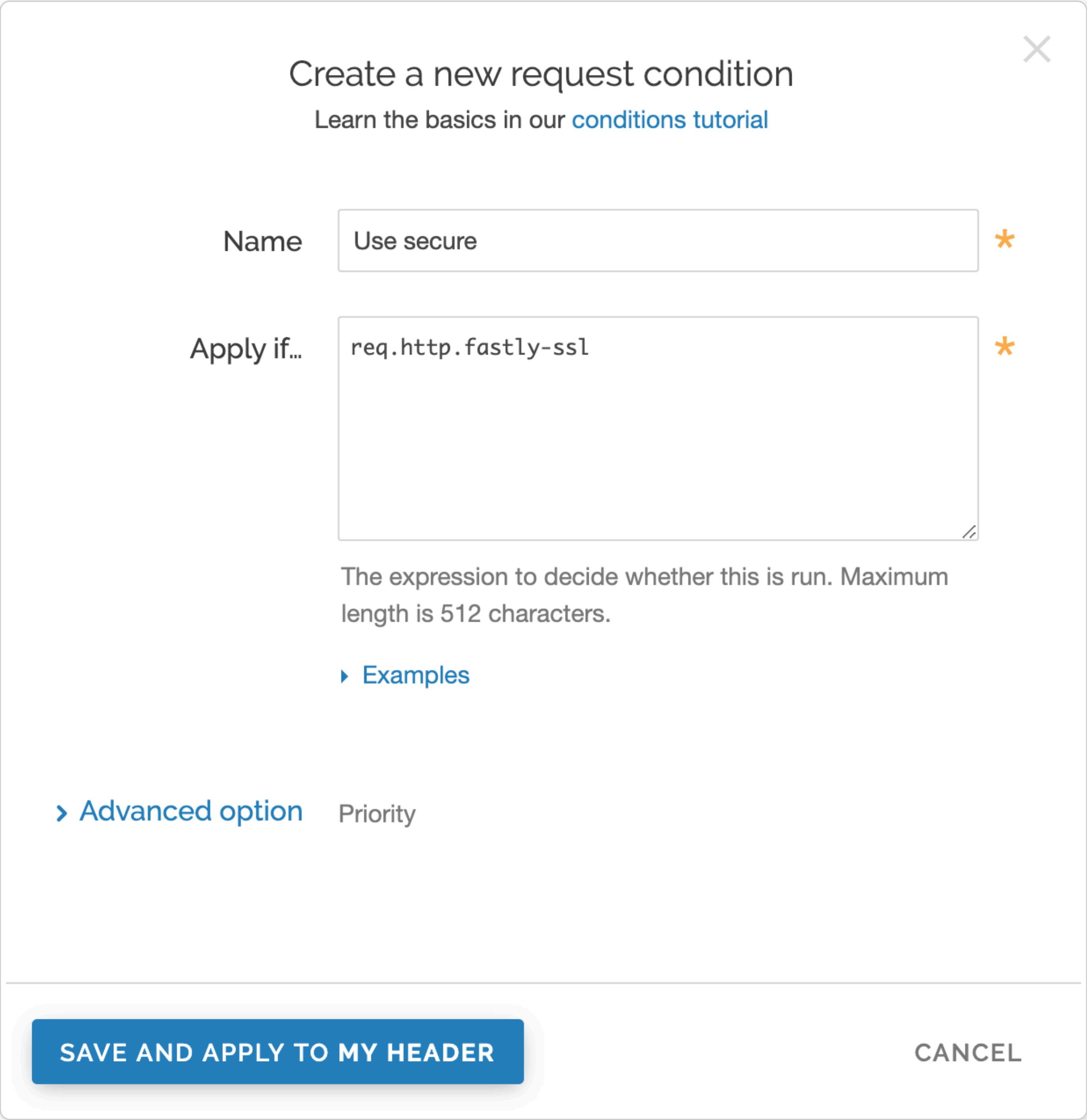
The expression to decide whether this is run. Maximum length is 512 characters.

Examples
Advanced option
Priority
SAVE AND APPLY TO MY HEADER
CANCEL

2. Fill out the **Create a new request** fields as follows:
    - In the **Name** field, enter the name of the condition specifying use of the non-secure server (for example, `Use non-secure`).
    - In the **Apply if** field, enter `!req.http.fastly-ssl`.
    - Leave the priority set to its default value.

3. Create the new condition by clicking **Save and apply to**.

4. On the **Origins** page, click **Attach a condition** next to the name of the secure server.



Create a new request condition

Learn the basics in our conditions tutorial

Name: Use secure

Apply if...: req.http.fastly-ssl

The expression to decide whether this is run. Maximum length is 512 characters.

5. Fill out the **Create a new request condition** window as follows:
   - In the **Name** field, enter the name of the condition specifying use of the secure server (for example, `Use secure`).
   - In the **Apply if** field, enter `req.http.fastly-ssl`.
   - Leave the priority set to its default value.

6. Create the new condition by clicking **Save and apply to**.

7. Click **Activate** to deploy your configuration changes.

* * *

Setting up redundant origin servers

/en/guides/setting-up-redundant-origin-servers

Sometimes you want to set up two different origin servers, one as a primary and one as a backup in case the primary becomes unavailable. You can do this via the web interface or using custom VCL.

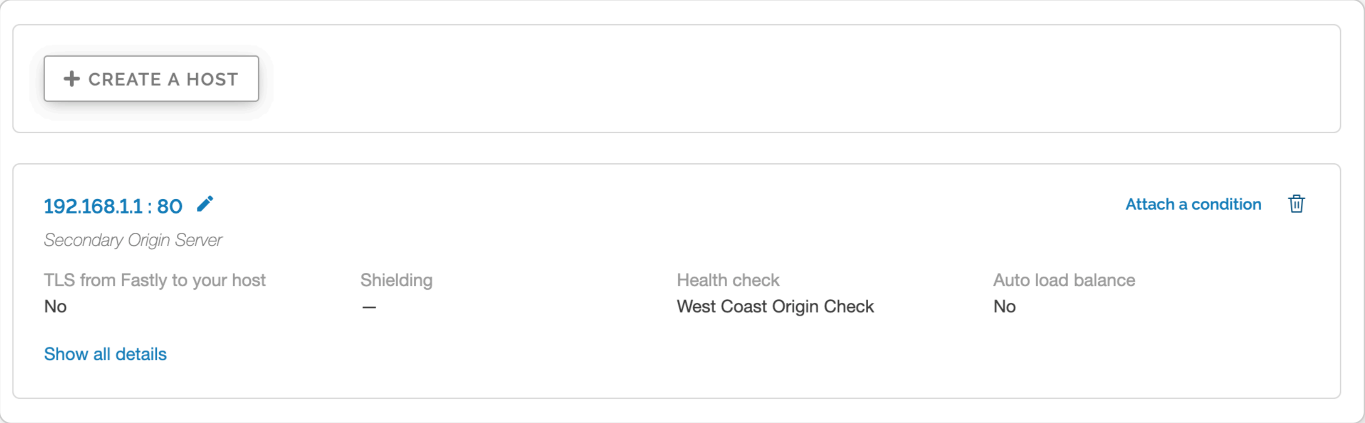
NOTE

Each Fastly service can be configured with up to five origin servers. Contact sales@fastly.com to enable more than five origin servers per service in your account.

Using the web interface

Set up redundant origins via the web interface using these steps.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. In the **Health Checks** area, define a health check and assign it to the primary origin server.
6. In the **Hosts** area, find your secondary origin server and click **Attach a condition**.

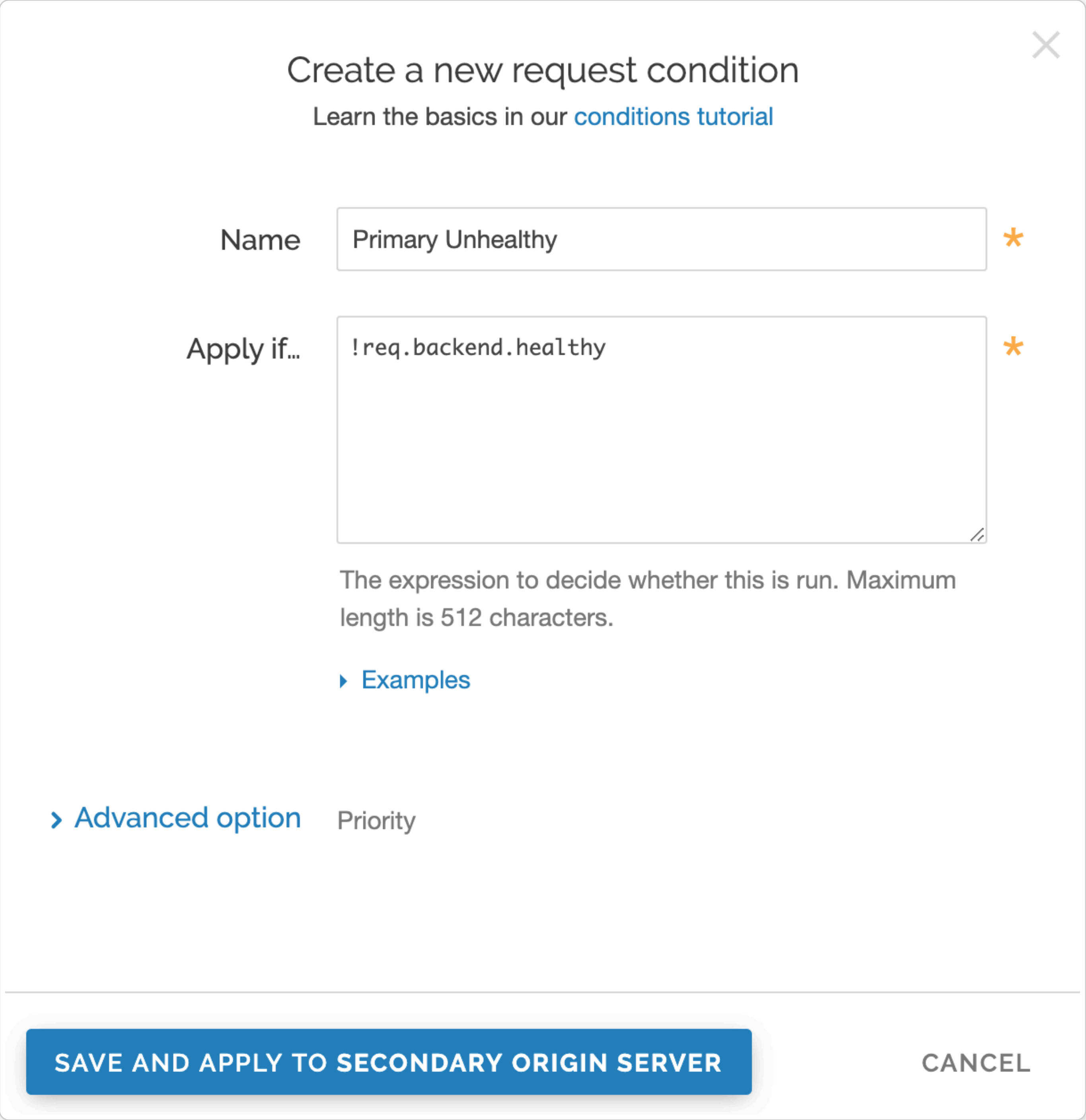


192.168.1.1 : 80

Secondary Origin Server

TLS from Fastly to your host: No
Shielding: —
Health check: West Coast Origin Check
Auto load balance: No

7. Click **Create a new request condition**.



Create a new request condition

Learn the basics in our conditions tutorial

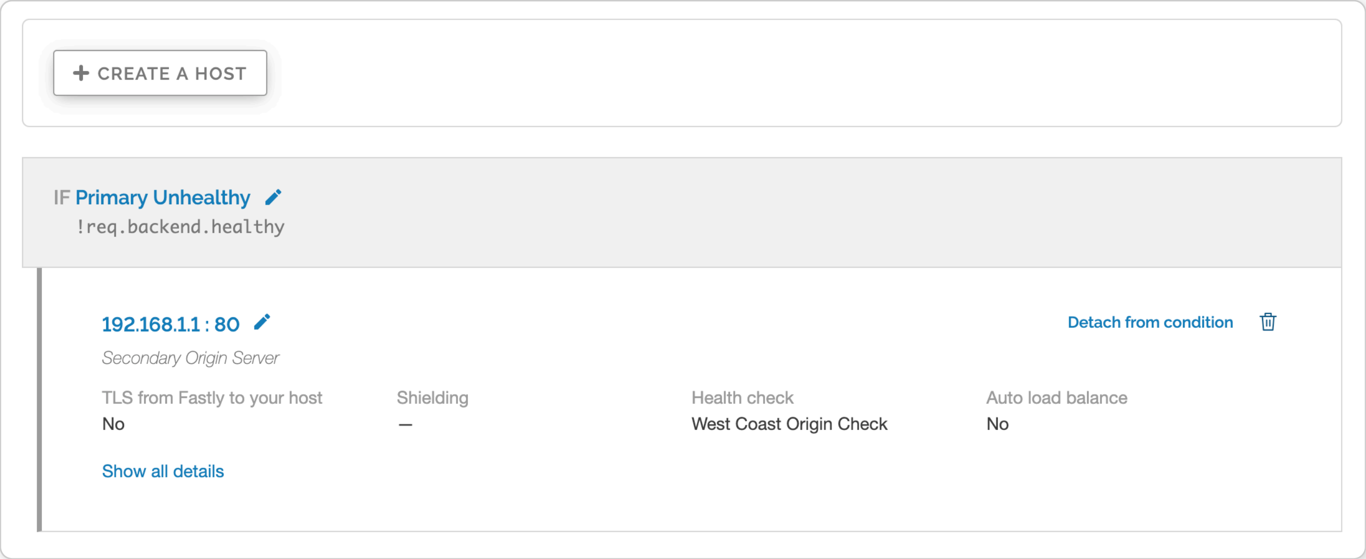
Name: Primary Unhealthy

Apply if...: `!req.backend.healthy`

The expression to decide whether this is run. Maximum length is 512 characters.

Examples
Advanced option
Priority
SAVE AND APPLY TO SECONDARY ORIGIN SERVER
CANCEL

8. Fill out the **Create a new request condition** fields as follows:
    - In the **Name** field, enter the name of your request condition (for example, `Primary Unhealthy`).
    - In the **Apply if** field, enter `!req.backend.healthy`.
9. Click **Save and apply to**. The Hosts area now displays the condition that must be met (Primary Unhealthy) in order for your secondary origin server to begin being used.



IF Primary Unhealthy

!req.backend.healthy

192.168.1.1 : 80

Secondary Origin Server

TLS from Fastly to your host	Shielding	Health check	Auto load balance
No	—	West Coast Origin Check	No

Once you've added the condition to your secondary origin server, the VCL generated by Fastly will reflect the new condition.

10. Preview the VCL, and confirm the following snippets appear in `vcl_recv`:

```
# default conditions
set req.backend = F_primary;
```

```
# Request Condition: primary unhealthy Prio: 10
if (!req.backend.healthy) {
  set req.backend = F_secondary;
}
#end condition
```

Using custom VCL

Set up redundant origins with custom VCL using these steps.

1. In the Fastly web interface, define a health check and assign it to the primary origin server.

2. Copy the boilerplate VCL from our guide on mixing Fastly VCL with custom VCL, and paste it into a new file.

3. Replace the `vcl_recv` sub with:

```
sub vcl_recv {
#FASTLY recv
  set req.backend = F_<primary_origin>;
  if (!req.backend.healthy) {
    set req.backend = F_<secondary_origin>;
  }
  if (req.method != "HEAD" && req.method != "GET" && req.method != "FASTLYPURGE") {
    return(pass);
  }
  return(lookup);
}
```

To find the exact backend names, view the generated VCL.

4. Upload your VCL file.

* * *

Specifying an override host

/en/guides/specifying-an-override-host

To rewrite the Host header being sent to your origin regardless of the Host used in the initial request, specify an override host. Use this if you have multiple domains tied to a service and want them all served by the same origin, if the domain your origin is expecting is different than one specified in your Fastly service, or if the Host header being sent to your origin is different from the Host used in the initial request.

You can override the Host header being sent to your origin by specifying the domain name of your override Host on the **Settings** page for a specific service or by specifying a host on the **Origins** page for a specific Host.

Here are some examples of when to use an override host:

- When using backends such as Amazon S3, Google Cloud Storage, or Heroku, you want to ensure you use the proper Host header so these providers know how to route requests directly to your content. Each provider uses the Host header to associate requests with your account's storage location. For example, if you set up your origin using Amazon S3, you send the name of your S3 bucket as your Host header. Amazon is set up so that it only accepts Host headers that have the same name as the bucket hosting your content. A request to `your-domain.com` must be re-written to `<BUCKET>.s3.<REGION>.amazonaws.com`, or else the request is denied.
- You have a service that contains three sites: `www.abc.com`, `www.myexample.com`, and `www.mysite.com` and you have one origin. You can have the same origin respond to each domain by overriding the Host header to one accepted by your origin, for example, `origin.example.com`. The result will be that a request to `www.abc.com`, `www.myexample.com`, or `www.mysite.com` returns content from `origin.example.com`.

NOTE

If you've initially set an override Host globally and then switch the configuration to set an override Host per origin, this will temporarily increase cache MISS and origin traffic because the Host header that Fastly was using for the cache lookup was changed from the client's original Host header. Once a new object is cached with the new host, cache HIT will be served.

Overriding a host at the origin level

You can add an override Host per origin if you're using an origin that requires a specific hostname to be passed to it. Once you’ve added a host, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by

type.

3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. In the **Hosts** area, click the pencil next to the Host you want to edit.
6. In the **Override host** field, enter the hostname of your override Host header based on the origin you're using. The value in this field will take precedence over anything you've set using the global override Host configuration. For example:
    - If you're using Amazon S3 as your origin, enter `<BUCKET>.s3.<REGION>.amazonaws.com`.
    - If you're using Google Cloud Storage as your origin, enter `<BUCKET>.storage.googleapis.com`.

    TIP

    To see other examples of Host headers for third-party services, refer to our developer documentation on overriding the Host header.

7. Click **Update**. The new override Host appears under the **Show all details** field of the **Override host** section and a code block is added to the origin definition in your VCL that will look similar to the following:

```
Backend F_Host_1 {
    .host = "..."; # IP or hostname
    .host_header = "example.com";
    .always_use_host_header = true;
     ...
}
```

8. Click **Activate** to deploy your configuration changes.

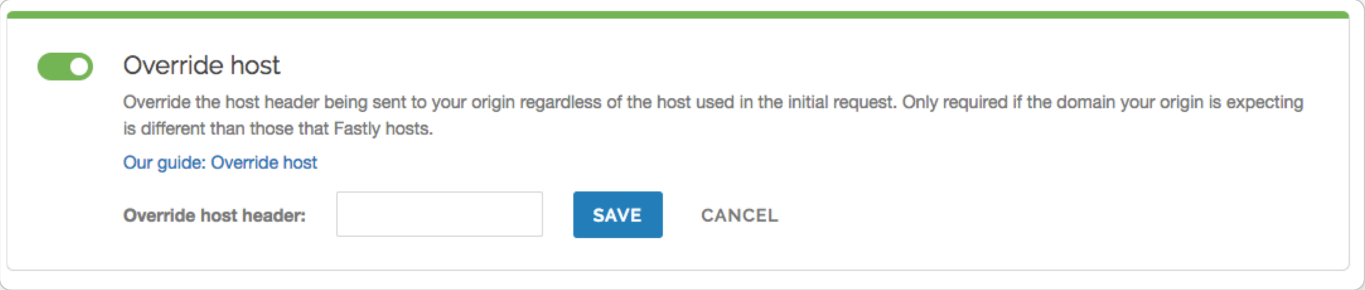
Overriding a host globally

NOTE

Use the global override if you only have one backend. If you do use the global override, be sure to read all the caveats below.

You can globally override a Host if your service has multiple domains to serve but they are all same assets (e.g., `assets1.example.com` and `assets2.example.com`) and you want to normalize them (e.g., `assets.example.com`). To globally override a host, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. Click the **Override host** switch.



Override host

Override the host header being sent to your origin regardless of the host used in the initial request. Only required if the domain your origin is expecting is different than those that Fastly hosts.

Our guide: Override host

6. In the **Override host header** field, enter the hostname of your override Host based on the origin you are using:
    - If you are using Amazon S3 as your origin, enter `<yourbucket>.s3.amazonaws.com`.
    - If you are using Google Cloud Storage as your origin, enter `<your bucket name>.storage.googleapis.com`.
7. Click **Save**. The new override host header appears in the Override host section.
8. Click **Activate** to deploy your configuration changes.

Caveats about using the global override host

There are situations when you may not want to use an override host:

- **Forcing TLS and enabling HSTS.** You may experience problems if you enable this setting along with the force TLS and enable HSTS setting. Instead of enabling this setting, create a new request setting and specify the override host in the advanced options.
- **Using multiple origins.** When you specify a Host override, you're specifying what hostname is actually sent to your origin. If you have a service with two different origins and each origin requires a different hostname, specifying a Host override for all requests results in one origin not returning valid responses. If you specify a default hostname that matches only one of the origins, then no content is returned from the other origin requests.

  NOTE

  To serve content from multiple backends, you should conditionally route to them. Check out our guide on URL path-based routing for more information.

- **Shielding is enabled.** If you enable a Host override along with shielding and the specified override host doesn't match to a domain within the service, the shield won't route the request properly and an error of 500 is expected. Refer to Shielding for more information.

  NOTE

  To ensure consistent behavior of Fastly customer services and origins, we normalize the host header's value to all lowercase in the `vcl_hash` function. This means that no matter how your site's domain name is capitalized in the request, the hash function will behave predictably. This does not apply to any other parts of the URL, which remain case-sensitive.

---

* * *

Using Fastly with apex domains

/en/guides/using-fastly-with-apex-domains

Some customers use only their second-level or apex domain (e.g., `example.com` rather than `www.example.com`) as their canonical domain. Due to limitations in the DNS specification, we don't recommend placing a CNAME record at the apex domain or using the CNAME Flattening features offered by some DNS providers (e.g., ALIAS or ANAME).

Instead, we offer anycast IP addresses for content that must be hosted on a second-level or apex domain. Our anycast options allow you to add A (IPv4) or AAAA (IPv6) records that point your apex domain at Fastly, prioritizing either performance or cost, depending on the option you choose.

> IMPORTANT
>
> Because anycast addressing methods don't offer Fastly as much flexibility in routing requests, our anycast options may not be as performant as our CNAME-based system. We recommend using our CNAME-based system for as much content as possible, particularly for large resources or streaming video.

Anycast options

Fastly offers the following anycast options to all paid customers.

Global anycast option

Fastly offers anycast IP addresses that allow you to use our entire global network to route requests to the nearest Fastly POP (from a network perspective), without regard to the billing region in which that POP resides.

Choose this option to prioritize traffic routing performance and to avoid restricting traffic to specific POPs. We'll provide you with a list of anycast IP addresses, which you then enter into your DNS records.

Billing Zone anycast options

> IMPORTANT
>
> Billing Zone anycast options are part of a limited availability release. For more information, see our product and feature lifecycle descriptions.

Fastly offers anycast IP addresses that allow you to prioritize where requests get routed based on the cost of its travel through groups of specific billing regions called *zones*.

Choose one of these options to prioritize cost savings over routing performance:

- **Maximum Billing Zone (MBZ) 100:** This option includes only the Fastly POPs located in the North America and Europe billing regions.
- **Maximum Billing Zone (MBZ) 200:** This option includes all Fastly POPs in MBZ 100 as well as those in the billing regions of Asia, Australia and New Zealand, and South America. It does not include POPs in the South Korea, India, and Africa billing regions.

Availability versus routing accuracy

IMPORTANT

Fastly prioritizes service availability over routing accuracy. Fastly will not offer invoice credits for traffic delivered from Fastly POPs outside of intended billing zones when that traffic is intentionally diverted to preserve the integrity or performance of Fastly services, for scheduled maintenance, or when the traffic routing changes are outside of Fastly’s control.

Because most anycast routing on the public internet is outside of our direct control, we cannot guarantee traffic routed to us will never arrive at POPs in higher billing zones. When routing changes on the public internet shift your traffic to higher priced Fastly POPs, however, we will make our best effort to investigate the cause and work with all parties involved to correct any problems discovered.

In addition to factors outside of our control, Fastly performs traffic engineering on a regular basis to ensure the availability of all Fastly services during incidents and scheduled maintenance to our network. When necessary, we may temporarily choose to route traffic to POPs outside of the chosen Billing Zones.

If you observe traffic being served from Fastly POPs outside of the intended Maximum Billing Zone, then before opening a support ticket:

- Check the Fastly Service Status page. Verify that there are no events that would explain the temporary rerouting of your traffic.
- Check the DNS records for your impacted domain names. Only the Fastly provided Billing Zone anycast IP addresses should be present in the DNS records of your impacted domain names.

If neither of the above issues explain your observed Fastly POP routing issue, open a support ticket within 30 days of the Fastly invoice date that reflects the billed traffic in question by contacting support.

Finding anycast IP addresses

When you choose a Billing Zone anycast option, we'll provide you with the list of anycast IP addresses for you to enter into your DNS records. Fastly will use these addresses to serve traffic only from the POPs included in the zones listed above, even if those POPs are unlikely to give the best performance for any given request.

When you have TLS configured

If TLS is configured for your Fastly service, you can obtain your global anycast IP addresses in the Fastly web interface by following these steps:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. Click **View details** for the domain you would like to route to Fastly via the global anycast option. The domain’s details page appears.

The A records section contains the global anycast IP addresses for you to enter into your DNS records of this domain.

When you don’t have TLS configured

If you don’t have TLS configured for the domain you would like to use with the global anycast option or would like to use one of the new Billing Zone anycast options, you can request your anycast IP addresses by contacting support. We'll provide you with the anycast IP addresses appropriate for the option you choose so you can enter those addresses into your DNS records. Be sure to enter all of them to ensure maximum availability and reliability. We don’t charge extra for these options, however, you must be using one of Fastly's paid plans (with or without a contract) to take advantage of them.

TIP

Fastly's billing regions can be found on our pricing page. We announce changes to these regions via our network status page.

Apex domain problems and their workarounds

The DNS instructions in RFC1034 (section 3.6.2) state that, if a CNAME record is present at a node, no other data should be present. This ensures the data for a canonical name and its aliases cannot be different. Because an apex domain requires NS records and usually other records like MX to make it work, setting a CNAME at the apex would break the "no other data should be present" rule.

In general, the problem with apex domains happens when they fail to redirect to their www equivalents (`example.com` points nowhere instead of pointing to `www.example.com`). Two workaround options exist:

- only use Fastly for API or AJAX calls, images, and other static assets (e.g., serve `example.com` yourself and CNAME to Fastly for assets at `assets.example.com`).
- redirect from the apex domain to the version proxied by Fastly (e.g., redirect any requests for `example.com` to `www.example.com`).

Neither workaround, however, is ideal.

IMPORTANT

To use TLS with an apex domain, explicitly add it to a certificate using one of our TLS products. For wildcard entries on our shared certificates, add an apex domain as a separate SAN entry.

* * *

Category: Headers

These articles describe configuration settings and changes you can make to your headers when setting up Fastly services.

Adding or modifying headers on HTTP requests and responses

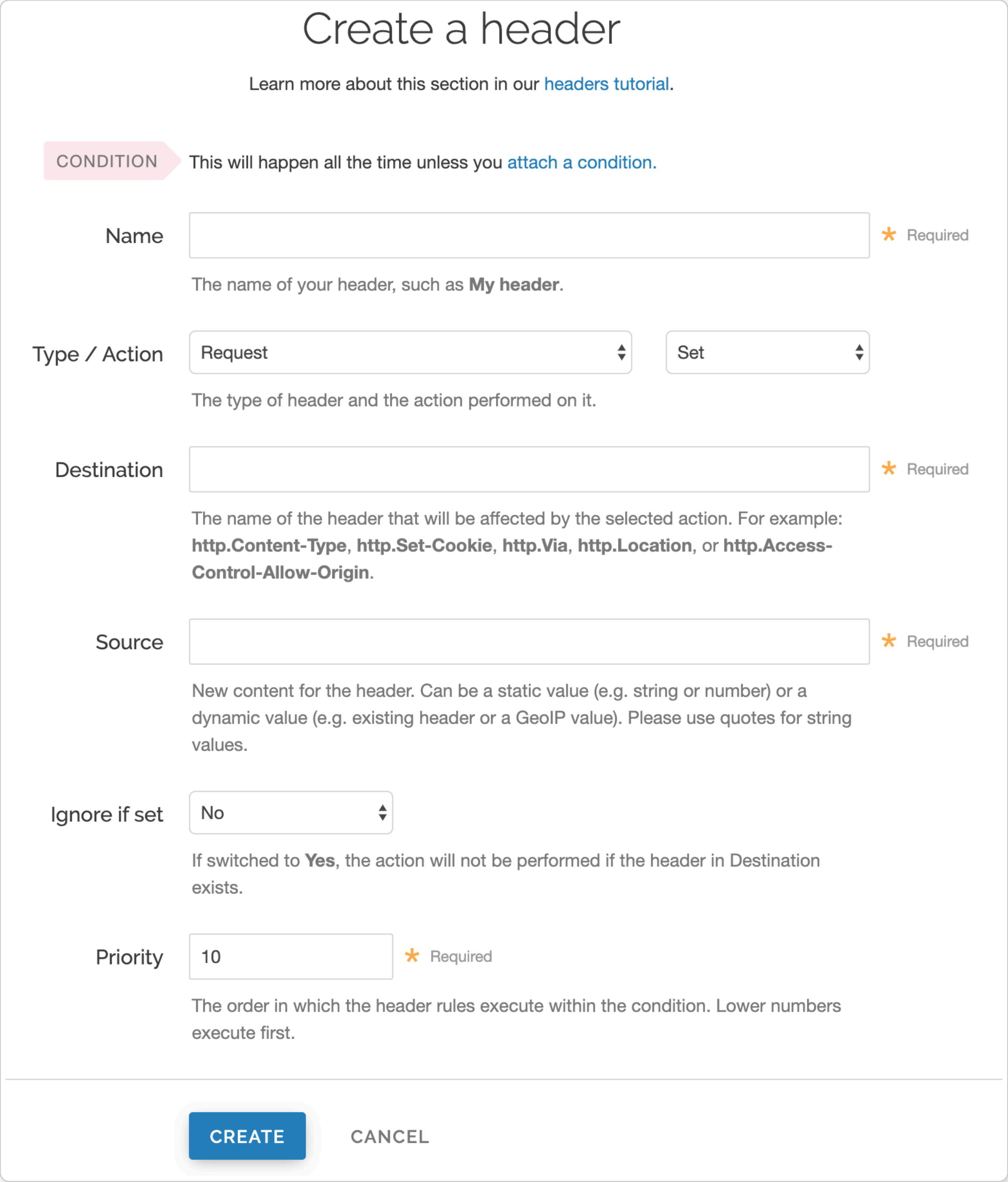
/en/guides/adding-or-modifying-headers-on-http-requests-and-responses

---

HTTP header fields are components of the header section of request and response messages in the Hypertext Transfer Protocol (HTTP). They define the operating parameters of an HTTP transaction. When you create and configure headers, you can determine how you want your content served to your users. The following steps show you how to add and edit headers.

Create new headers

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name ☐ * Required

The name of your header, such as **My header**.

Type / Action: Request, Set

The type of header and the action performed on it.

Destination ☐ * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source ☐ * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter the name of your header rule (for example, `My header`).
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter the name of the header affected by the selected action.
    - In the **Source** field, enter where the content for the header comes from.

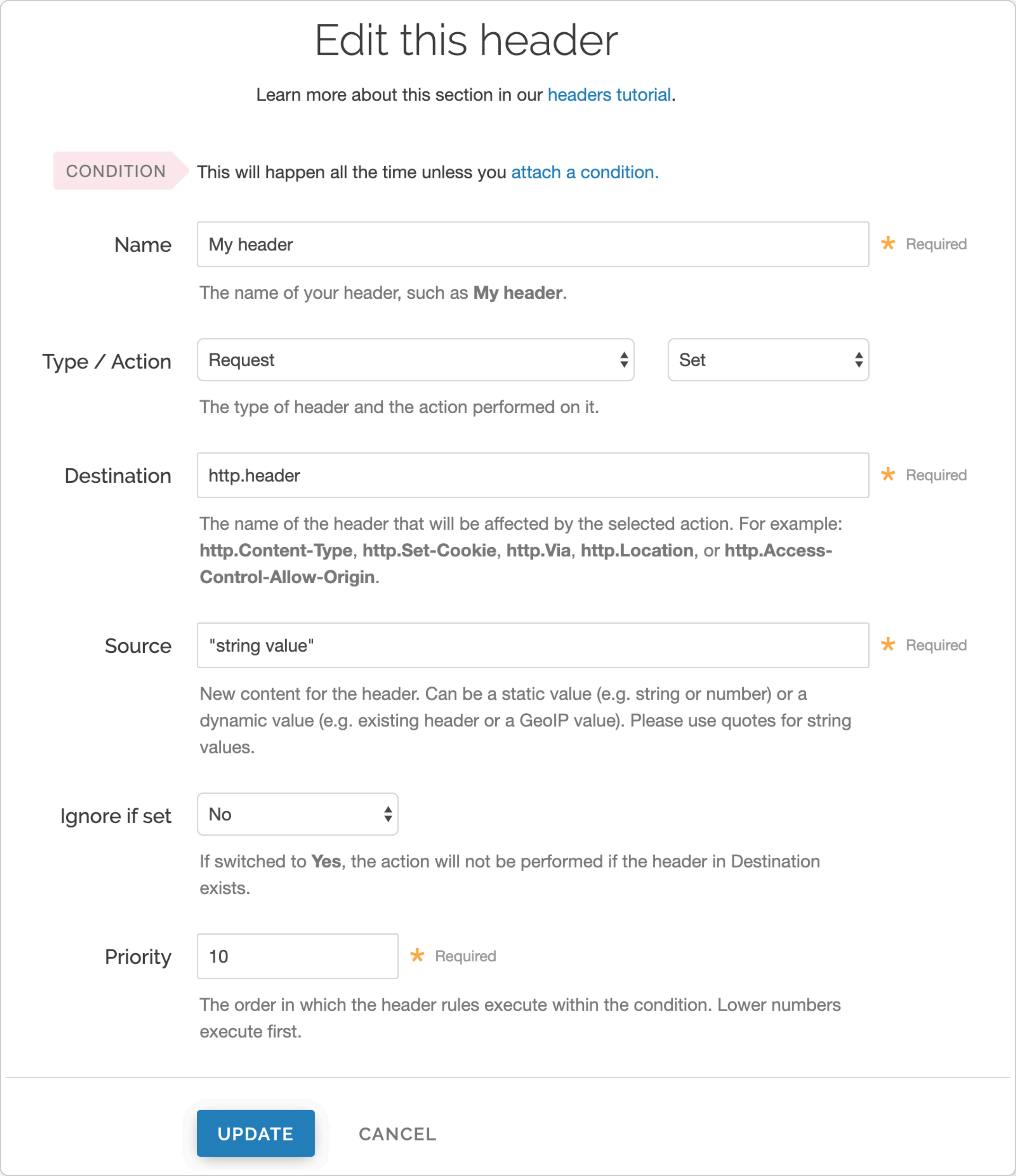
  - From the **Ignore if set** menu, select **No** if you want the header in the **Destination** field modified or select **Yes** if you don't want it modified.
  - In the **Priority** field, enter the order the header rules execute.

  The Field description table below provides additional details about each of these controls.

7. Click **Create**.
8. Click **Activate** to deploy your configuration changes.

Edit headers

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click the name of the header you want to edit.



Edit this header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: My header ★ Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.header ★ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "string value" ★ Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 ★ Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Edit this header** fields as follows:
    - In the **Name** field, enter the name of your header rule (for example, `My header`).
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter the name of the header affected by the selected action.
    - In the **Source** field, enter where the content for the header comes from.

- From the **Ignore if set** menu, select **No** if you want the header in the **Destination** field modified or select **Yes** if you don't want it modified.
- In the **Priority** field, enter the order the header rules execute.

7. Click **Update**.
8. Click **Activate** to deploy your configuration changes.

Field description table

This table describes what each field in the Header window means:

Field	Description
Name	The Name field specifies a memorable word or phrase that allows you to recognize and remember a particular Header rule.
Type	The Type menu can be set to **Request**, **Response**, or **Cache**. Selecting **Request** modifies the request coming from the user, and this will carry through to the request that gets sent to your origin server. Selecting **Response** affects the HTTP response that is sent back to the user. Selecting **Cache** affects the HTTP response that your origin server returns before it gets stored on Fastly servers, meaning whatever changes you make there will be remembered on a cache hit.
Action	The Action menu can be set to **Set**, **Append**, **Delete**, **Regex**, and **Regex All**. Selecting **Set** (the default) will write a value into the header (potentially overwriting it, if it already exists). Selecting **Append** will add a value onto the end of a header or set it if it doesn't exist. Selecting **Delete** will remove a header. When selected, it hides the Source field in the Header window. Selecting **Regex** allows you to perform a find and replace on specific text and is based on a regular expression you type in. When selected, the Regex and Substitution controls appear in the Header window. Selecting **Regex All** allows you to perform the same function as **Regex** but it performs a find and replace multiple times. When selected, the Regex and Substitution controls appear in the Header window.
Destination	The Destination field determines the name of the header that is going to be affected by our Action. Because header rules can be used to affect more than just HTTP headers, your input to this field should be formatted like this: `http.Header-Name`.
Source	The Source field is available on Set, Append, Regex, and Regex All actions. This field becomes hidden in the Header window when you select Delete from the Action menu. It determines where the new content for the header comes from. There are a plethora of options for Source. The simplest is a static string such as `"My Static String"` (including the quotes). Other options include `client.ip`, `req.http.Another-Header`, and `client.geo.city`. See the list of Common Sources below for more common sources of new content.
Regex	The Regex field only appears in the Header window when you select Regex or Regex All from the Action menu. It allows you to perform a find and replace on specific text and is based on a regular expression that you type in.
Substitution	The Substitution field only appears in the Header window when you select the Regex and Regex All from the Action menu. It replaces the text that was removed by the regex expression with the text you typed in the Substitution field.

Field	Description
Ignore if set	By default this is set to No, which means that if the header you are modifying already exists, it will be modified.
Priority	The Priority field determines the order in which the header rules execute (e.g., a priority of 1 means the header rule executes first). This can be important if you set headers and then set other headers based on the earlier ones.

Common sources of new content

Name	Valid Types	Description
`req.http.Fastly-Client-IP`	Request, Cache, Response	The true IP address of the client.
`client.ip` and `client.identity`	Request, Cache, Response	The client IP address. These variables are available, but may not always display the source IP address. For instance, they may show the edge node IP when shielding is enabled. For the true client IP address, use `req.http.Fastly-Client-IP`. **IMPORTANT:** In some cases, client IP data may be considered sensitive. Make sure you protect the sensitive IP data you stream or store.
`server.identity`	Request, Cache, Response	A unique identifier for the Fastly server processing the request.
`server.region`	Request, Cache, Response	The region in which the Fastly server resides.
`server.datacenter`	Request, Cache, Response	The data center in which the Fastly server resides.
`req.url`	Request, Cache, Response	The URL of the HTTP Request from the client.
`req.http.*`	Request, Cache, Response	The headers from the HTTP Request, access as: `req.http.HeaderName`
`beresp.status`	Cache	The status returned from the origin server.
`beresp.http.*`	Cache	The headers from the origin's HTTP Response, access: `beresp.http.HeaderName`
`resp.status`	Response	The status that is going to be returned to the client.

Name	Valid Types	Description
`resp.http.*`	Response	The headers in the HTTP Response to be returned to the client, access: `resp.http.HeaderName`
`client.geo.*`	Request, Cache, Response	Geolocation values for the client's IP.

* * *

Enabling cross-origin resource sharing (CORS)

/en/guides/enabling-cross-origin-resource-sharing

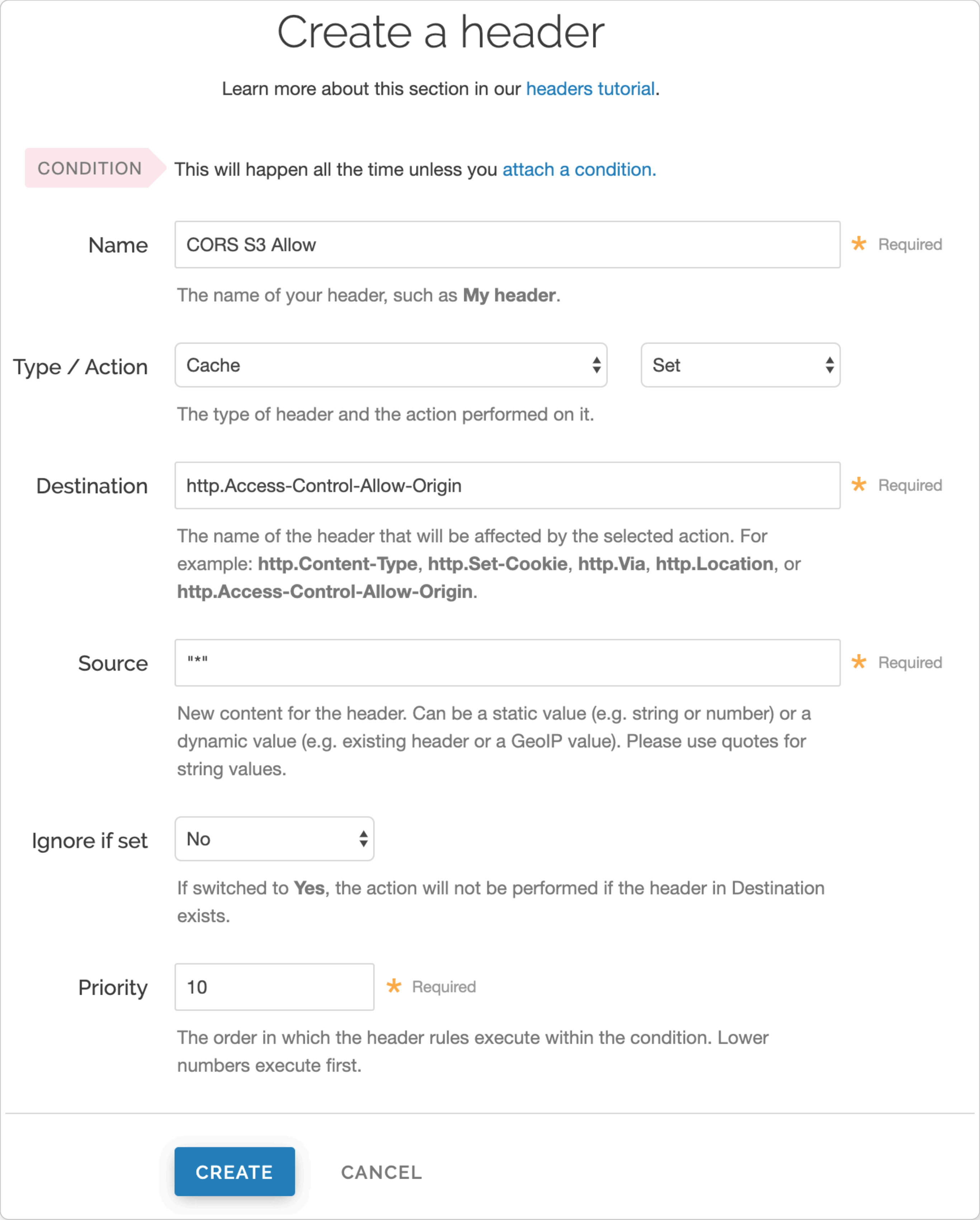
Enabling Cross-Origin Resource Sharing (CORS) allows a server to indicate that other origins can request sub-resources, like scripts and stylesheets, from it. These origins might use a different scheme (HTTP vs HTTPS) or an entirely different domain or port. This guide describes how to add an `Access-Control-Allow-Origin` header, which is sufficient for simple scenarios, and is often useful when using static bucket providers like Amazon S3 and Google Cloud Storage as an origin.

IMPORTANT

We recommend only enabling CORS when you have sub-resources on your server that you want other origins to load, especially if you allow requesting code from *any* origin, as it makes things less secure.

To enable CORS, set up a custom HTTP header for your service by following the steps below.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: CORS S3 Allow (Required)

The name of your header, such as **My header**.

Type / Action: Cache / Set

The type of header and the action performed on it.

Destination: http.Access-Control-Allow-Origin (Required)

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "*" (Required)

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 (Required)

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Do the following with the **Create a header** controls depending on whether you're creating a header for all origins or creating a header for a specific origin:

For this control	When creating a header for any origin	When creating a header for a specific origin
Name	Enter a descriptive name for the new header (e.g., `CORS S3 Allow`).	Enter a descriptive name for the new header (e.g., `CORS S3 Allow`).
Type	Select **Cache**.	Select **Response**
Action	Select **Set**.	Select **Set**.
Destination	Enter `http.Access-Control-Allow-Origin`.	Enter `http.Access-Control-Allow-Origin`.
Source	Enter `"*"`.	Enter a specific origin (e.g., `https://example.com`).
Ignore if set	Leave the default value.	Leave the default value.
Priority	Leave the default value.	Leave the default value.

Any time you specify a single origin in the **Source** field to ensure CORS rules are applied in `vcl_deliver`, the system will generate VCL under `vcl_deliver` as `set resp.http.Access-Control-Allow-Origin = "<specified origin>";`.

7. Click **Create**. The new header appears on the Content page.

8. Click **Activate** to deploy your configuration changes.

IMPORTANT

Objects already cached won't have this header applied until you purge them.

Test it out

Running the command:

```
$ curl -I example.tld/path/to/resource
```

should include similar information to the following in your header:

```
Access-Control-Allow-Origin: http://example.tld
Access-Control-Allow-Methods: GET
Access-Control-Expose-Headers: Content-Length, Connection, Date...
```

TIP

`Access-Control-Allow-Methods` and `Access-Control-Expose-Headers` are examples of additional headers you can add with CORS. For more information about these headers, check out MDN Web Docs.

---

* * *

Removing headers from backend response

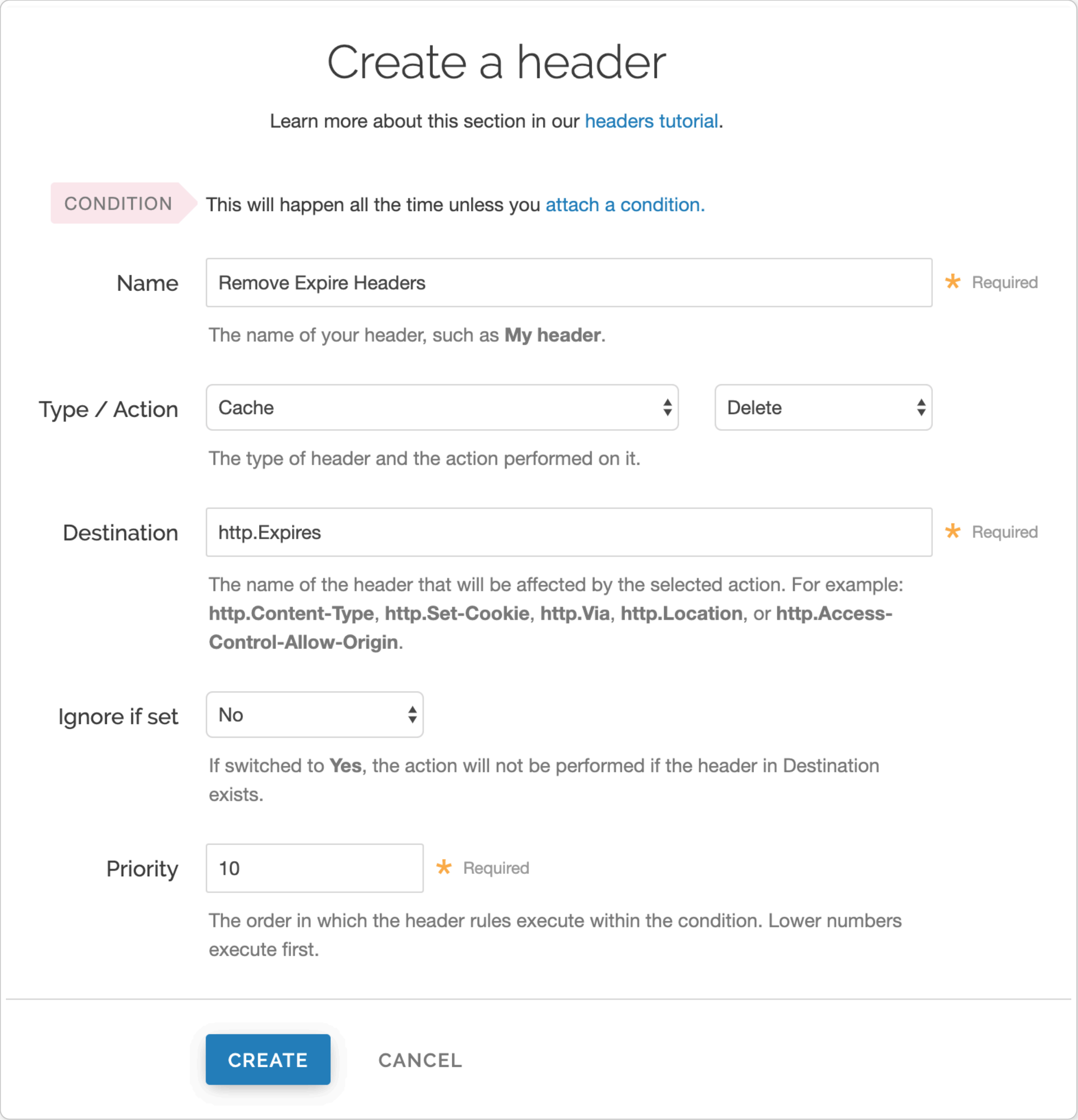
/en/guides/removing-headers-from-backend-response

You can remove headers from any backend response. This may be necessary if your application automatically sets headers. For example, Drupal can set the following Expires and Cache-Control headers to prevent caching:

```
Expires: Sun, 19 Nov 1978 05:00:00 GMT
Last-Modified: Wed, 18 Jul 2012 18:52:16 +0000
Cache-Control: no-cache, must-revalidate, post-check=0, pre-check=0
```

To remove a header from the backend response, add a new header as follows:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Remove Expire Headers

The name of your header, such as **My header**.

Type / Action: Cache / Delete

The type of header and the action performed on it.

Destination: http.Expires

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter a descriptive name for the header rule (e.g., `Remove Expire Headers`).
   - From the **Type** menu, select **Cache**, and from the **Action** menu, select **Delete**
   - In the **Destination** field, enter the name of the header (e.g., `http.Expires`).
   - From the **Ignore if set** menu, select **No**.
   - In the **Priority** field, enter `10`.
7. Click **Create**.
8. Click **Activate** to deploy your configuration changes.

TIP

You may also be interested in our information on setting content type based on file extension.

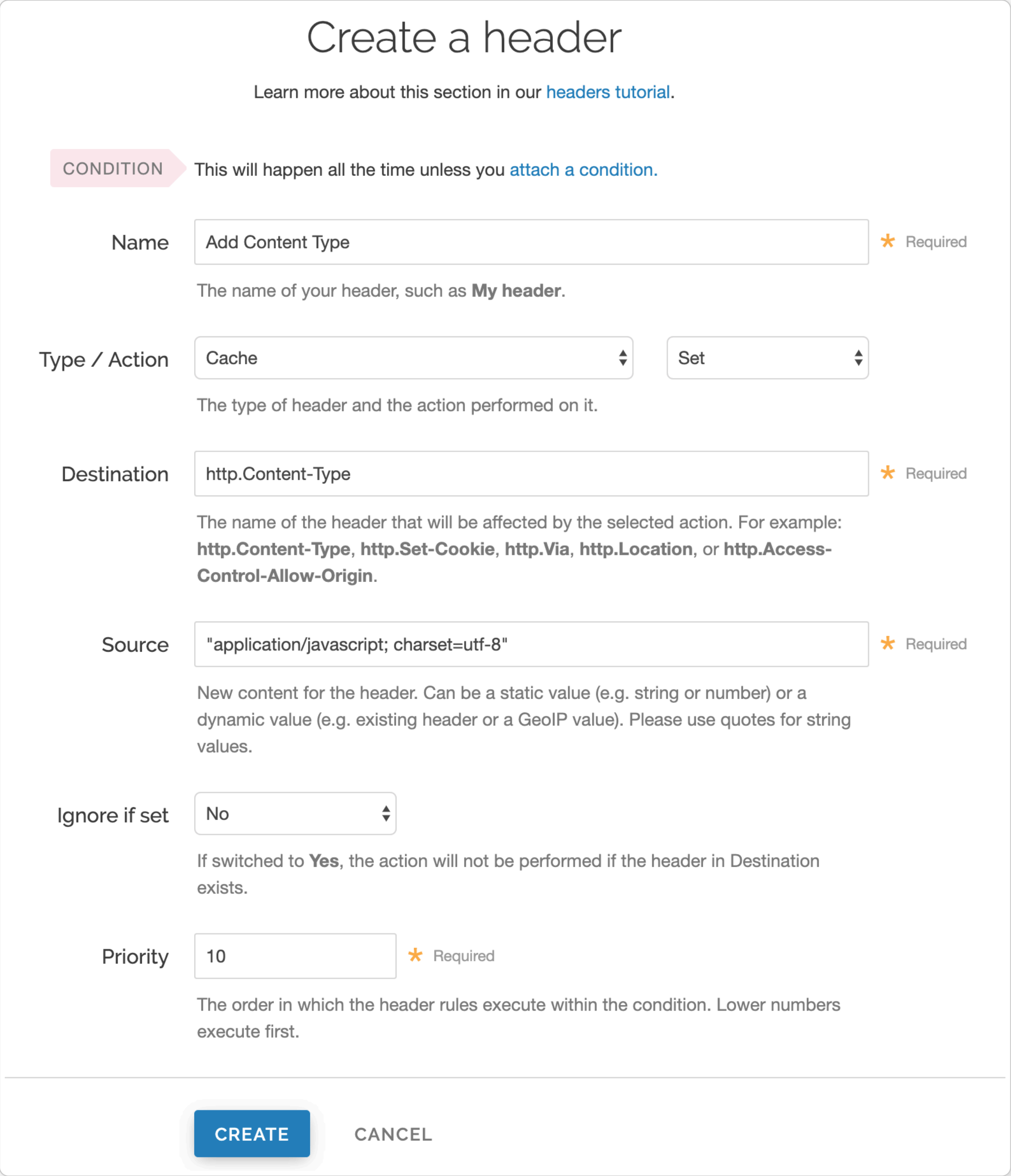
* * *

Setting Content Type based on file extension

/en/guides/setting-content-type-based-on-file-extension

In some situations you may want to override the content type that a backend returns. To do that you will need to create a new header object and an associated condition.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Add Content Type

The name of your header, such as **My header**.

Type / Action: Cache / Set

The type of header and the action performed on it.

Destination: http.Content-Type

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "application/javascript; charset=utf-8"

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

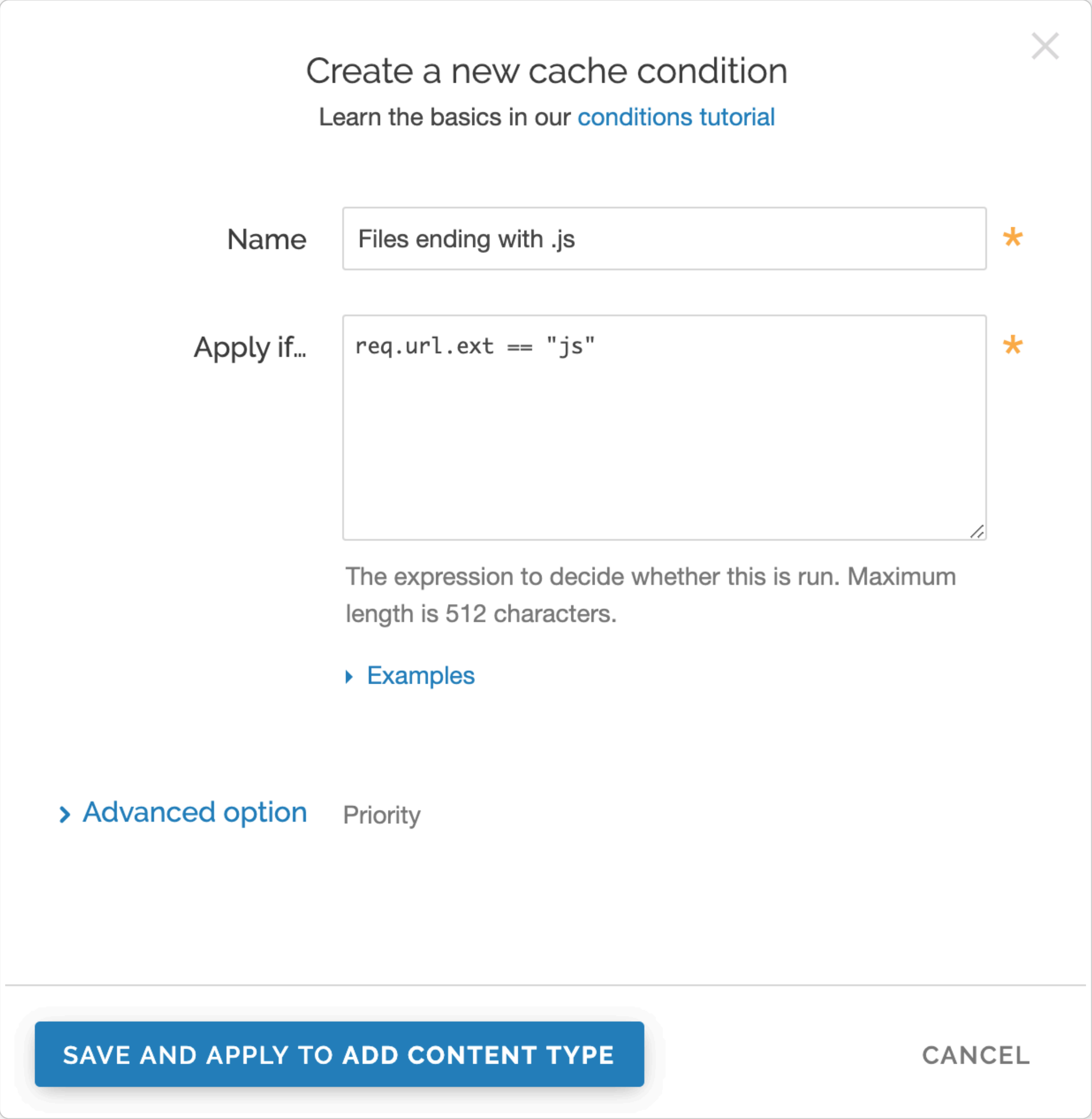
6. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter an appropriate name (e.g., `Add Content Type`).
    - From the **Type** menu, select **Cache**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `http.Content-Type`.

  - In the **Source** field, enter the content type you want to match, such as `"application/javascript; charset=utf-8"`.
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**.

Once you have created the header object, apply a condition. Otherwise, that particular object is applied to all requests.

1. Click **Attach a condition** to the right of the new header name.



Create a new cache condition

Learn the basics in our conditions tutorial

Name: Files ending with .js *

Apply if...:

```
req.url.ext == "js"
```

The expression to decide whether this is run. Maximum length is 512 characters.

2. Fill out the **Create a new cache condition** fields as follows:

   - In the **Name** field, enter a descriptive name, such as `Files ending with .js`.
   - In the **Apply if** field, enter the condition that matches your request, such as `req.url.ext == "js"` (to match the request for files ending in .js).

3. Click **Save and apply to**. The new condition is created.
4. Click **Activate** to deploy your configuration changes.

TIP

You may also be interested in our guide to Removing headers from backend response.

* * *

Category: Image optimization

These articles provide basic instructions for and examples of setting up and beginning to use the Fastly Image Optimizer.

About Fastly Image Optimizer

/en/guides/about-fastly-image-optimizer

Fastly's Image Optimizer (Fastly IO) is an image optimization service that manipulates and transforms your images in real time and caches optimized versions of them. When an image is requested from your origin server, Fastly IO can perform one or more transformation tasks before serving and caching the optimized version. For example, you can resize, adjust quality, crop and trim, serve responsive images, and more.

WARNING

Only send image content through Fastly IO. Non-image content can't be optimized using it, but will still be counted and charged as an image optimization request, which may cost you more.

Before you begin

Before you begin using Fastly IO, keep the following in mind:

- Fastly's Image Optimizer is disabled by default. It can be purchased for an account by contacting sales@fastly.com. Once purchased, it can be enabled for a service in the web interface by anyone assigned the role of superuser, which will result in changes to your monthly bill.
- Services that use image optimization require shielding. When setting up shielding, be sure to choose a shield location as geographically close to your image's origin as possible. Our guide to enabling shielding provides more information on how to enable shielding. Take special note of the step immediately following your shielding location selection in that guide. If the Host header for the service has been changed from the default, you must ensure the new hostname is added to the list of domains.

- Using premium Fastly IO features (e.g., the AVIF encoding format) requires the purchase of Image Optimizer Professional. Contact sales@fastly.com to purchase Image Optimizer Professional for your account.

Starting a trial

Anyone assigned the role of superuser can start a Fastly IO trial on the Trials page.

Enabling and disabling the Fastly Image Optimizer

After Fastly IO has been purchased for your account, it can be enabled and disabled in the web interface by anyone assigned the role of superuser.

> NOTE
>
> To enable or disable Fastly IO via the API, check out our developer documentation.

Enabling the Fastly Image Optimizer

To enable Fastly IO for a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Image Optimizer**.
5. Click the **Image Optimizer** switch to enable Fastly IO for the service.

> NOTE
>
> If you've been assigned the role of superuser and you don't see the switch to enable Fastly IO, contact sales.

Disabling the Fastly Image Optimizer

To disable Fastly IO for a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Image Optimizer**.
5. Click the **Image Optimizer** switch to disable Fastly IO for the service.

Setting up image optimization

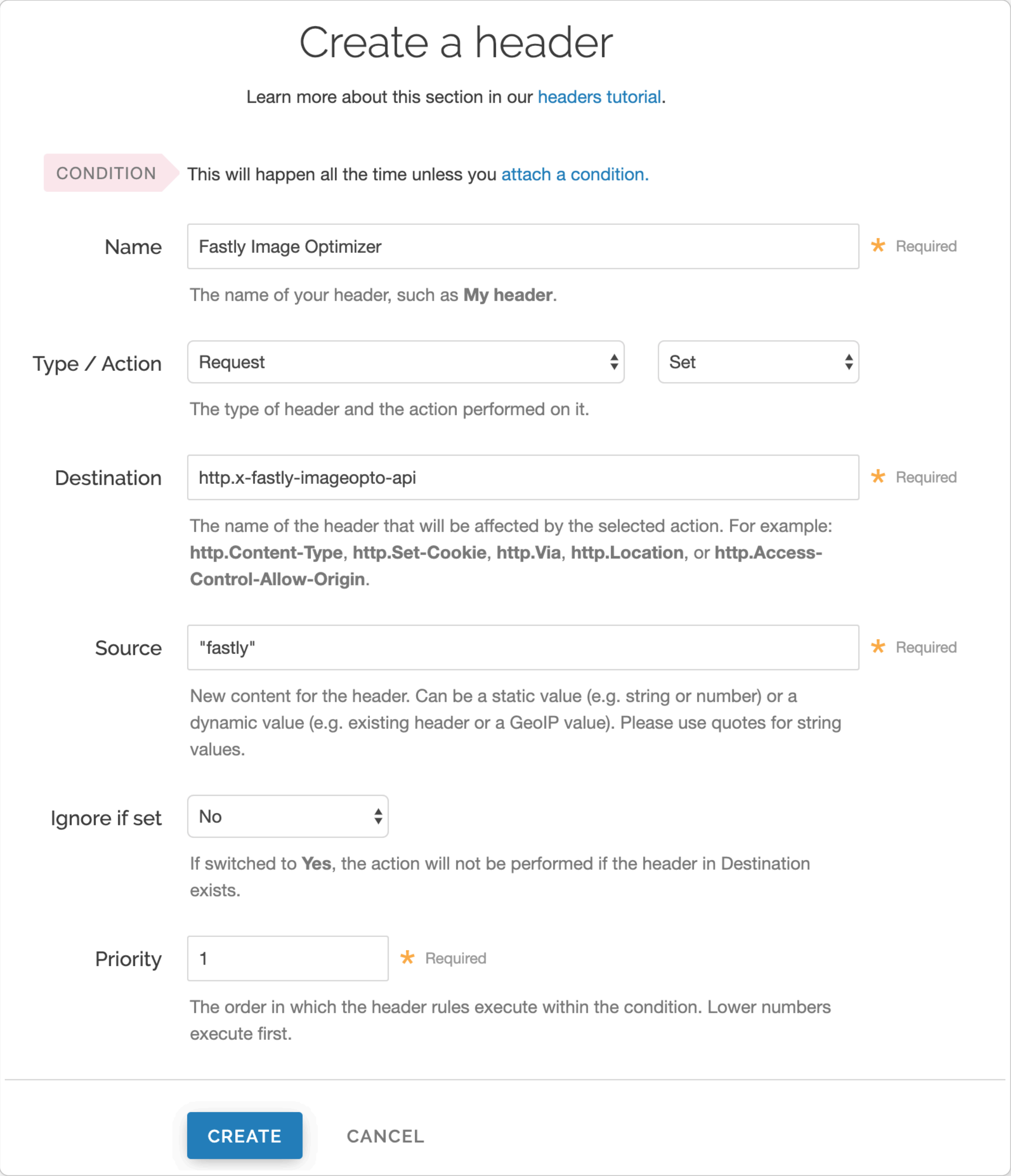
Once you've enabled Fastly IO for your service, you can set up image optimization by following this process:

1. Add a header.
2. Create a request condition.
3. Test an image.

Add the Fastly Image Optimizer header

Once image optimization has been enabled for your service, configure your service by adding the Fastly Image Optimizer header.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Fastly Image Optimizer

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.x-fastly-imageopto-api

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "fastly"

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 1

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** window as follows:
   - In the **Name** field, enter `Fastly Image Optimizer`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.x-fastly-imageopto-api`.
   - In the **Source** field, enter `"fastly"`. By default, the Fastly Image Optimizer removes any additional query string parameters that are not part of our image API. If your source image requires delivery of additional

query string parameters from origin then enter `"fastly; qp=*"` instead.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `1`.

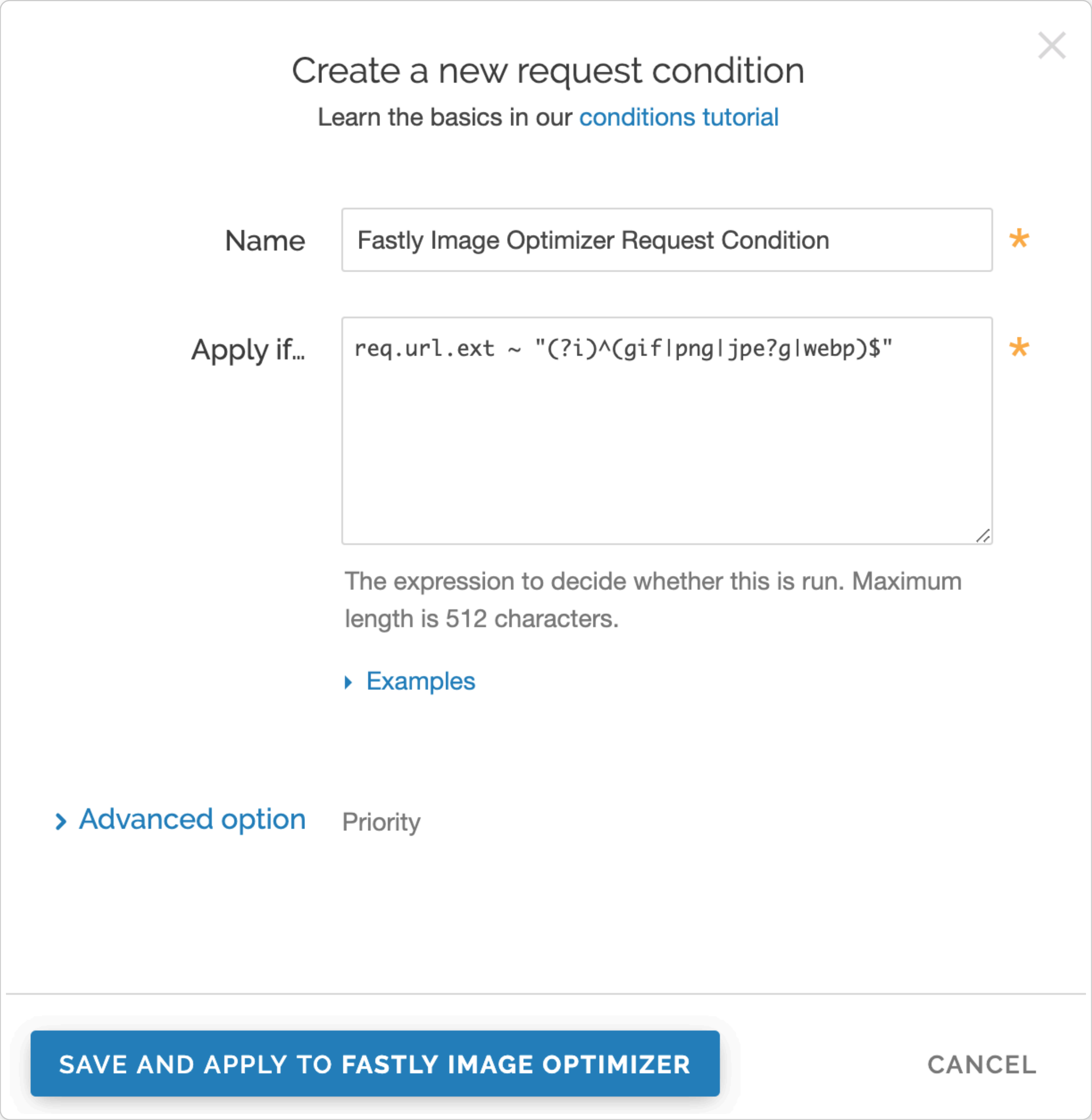
7. Click **Create** to create the new header.

> ✓ TIP
>
> For more help with adding or modifying headers, see our guide.

Create a request condition

To ensure only your image assets are routed via the Fastly Image Optimizer, create a request condition.

1. Click **Attach a condition** next to the `Fastly Image Optimizer` header.
2. Click **Create a new request condition**.



Create a new request condition

Learn the basics in our conditions tutorial

Name: Fastly Image Optimizer Request Condition

Apply if...:

```
req.url.ext ~ "(?i)^(gif|png|jpe?g|webp)$"
```

The expression to decide whether this is run. Maximum length is 512 characters.

3. Fill out the **Create a new request condition** window as follows:
    - In the **Name** field, enter a descriptive name for the new condition (for example, `Fastly Image Optimizer Request Condition`).
    - In the **Apply if** field, enter the appropriate request condition. For example, `req.url.ext ~ "(?i)^(gif|png|jpe?g|webp)$"` will send all files with gif, png, jpg, jpeg, and webp extensions via the Fastly Image Optimizer. Likewise, `req.url ~ "^/images/"` will send all files in the `images` directory via the Fastly Image Optimizer.
4. Click **Save and apply** to create the new condition for the header.
5. Click **Activate** to deploy your configuration changes.

TIP

For more help using conditions, see our guide.

Confirm everything is working

Once you've activated your changes, check to see if the Fastly Image Optimizer is processing your image request by typing the following command on the command line:

```
$ echo -n "Image Width: " ; curl -sI "https://www.fastly.io/image.jpg?width=200" | grep "
```

Replace `https://www.fastly.io/image.jpg?width=200` with the full image URL and width of the image you're testing.

The command line output will display the image's width, which should match the width query string you added to your image's URL. For example, the output might be:

```
Image Width: 200
```

Configuring default image settings

The Fastly Image Optimizer supports a variety of image formats and applies specific settings to all images by default. Use the Fastly web interface to review and adjust the default settings as appropriate. Changes to other image settings, including most image transformations, require using query string parameters.

To review and edit the default image settings via the web interface, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Image Optimizer**.

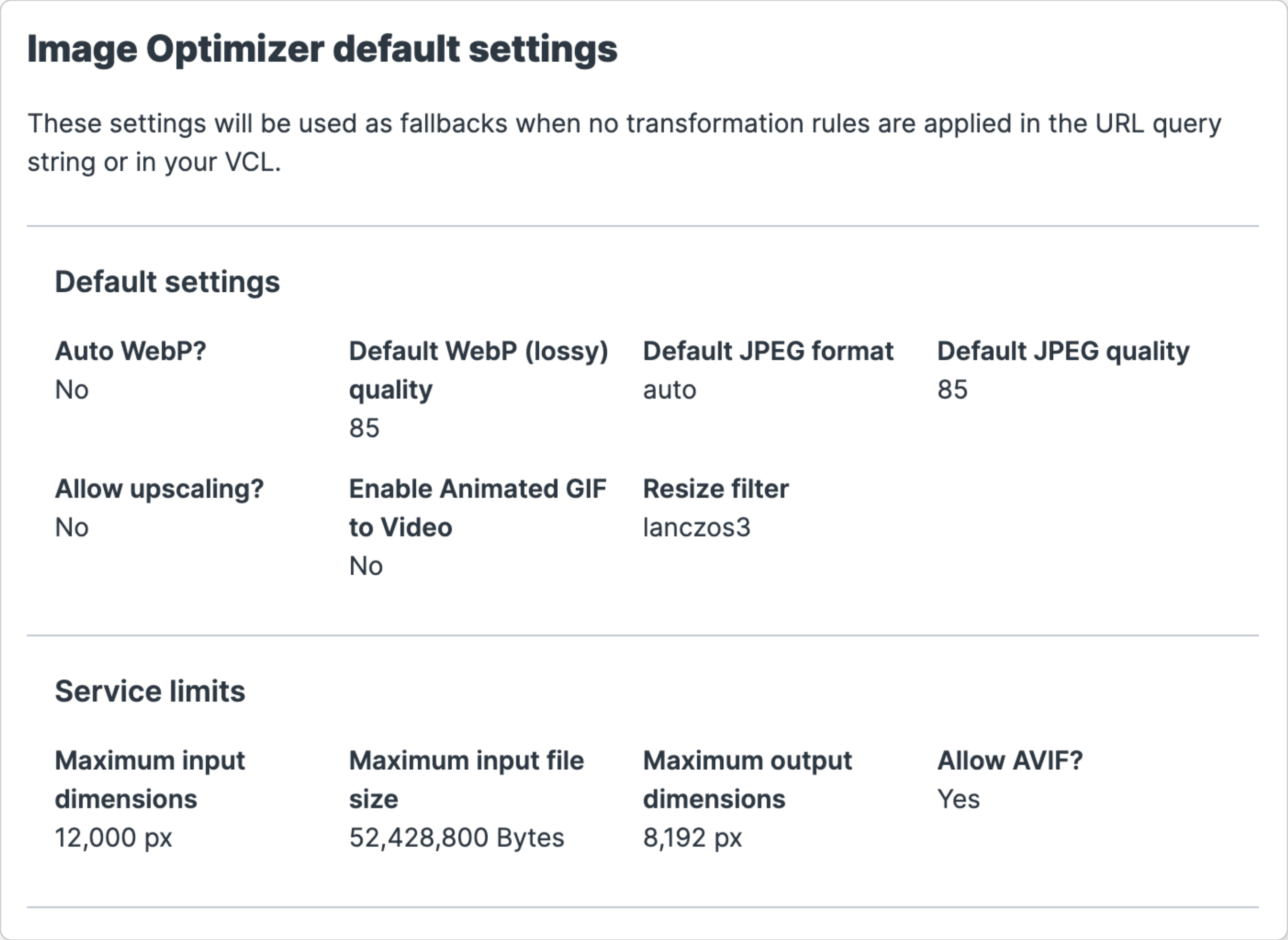


Image Optimizer default settings

These settings will be used as fallbacks when no transformation rules are applied in the URL query string or in your VCL.

Default settings

Auto WebP? No

Default WebP (lossy) quality 85

Default JPEG format auto

Default JPEG quality 85

Allow upscaling? No

Enable Animated GIF to Video No

Resize filter lanczos3

Service limits

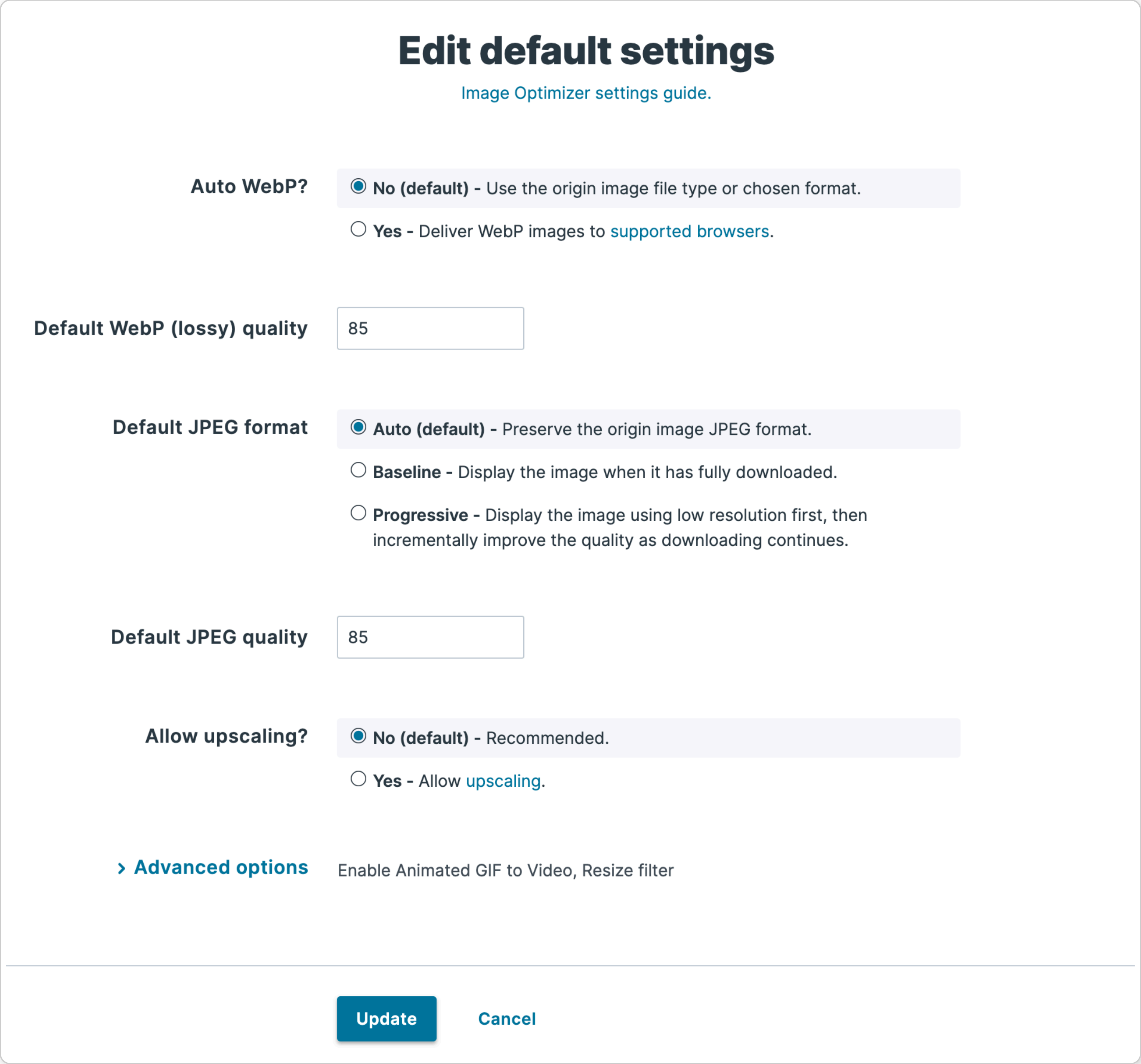
Maximum input dimensions 12,000 px

Maximum input file size 52,428,800 Bytes

Maximum output dimensions 8,192 px

Allow AVIF? Yes

5. Click the pencil next to the **Default settings**.



Edit default settings

Image Optimizer settings guide.

Auto WebP?
● **No (default) -** Use the origin image file type or chosen format.
○ **Yes -** Deliver WebP images to supported browsers.

Default WebP (lossy) quality: 85

Default JPEG format
● **Auto (default) -** Preserve the origin image JPEG format.
○ **Baseline -** Display the image when it has fully downloaded.
○ **Progressive -** Display the image using low resolution first, then incrementally improve the quality as downloading continues.

Default JPEG quality: 85

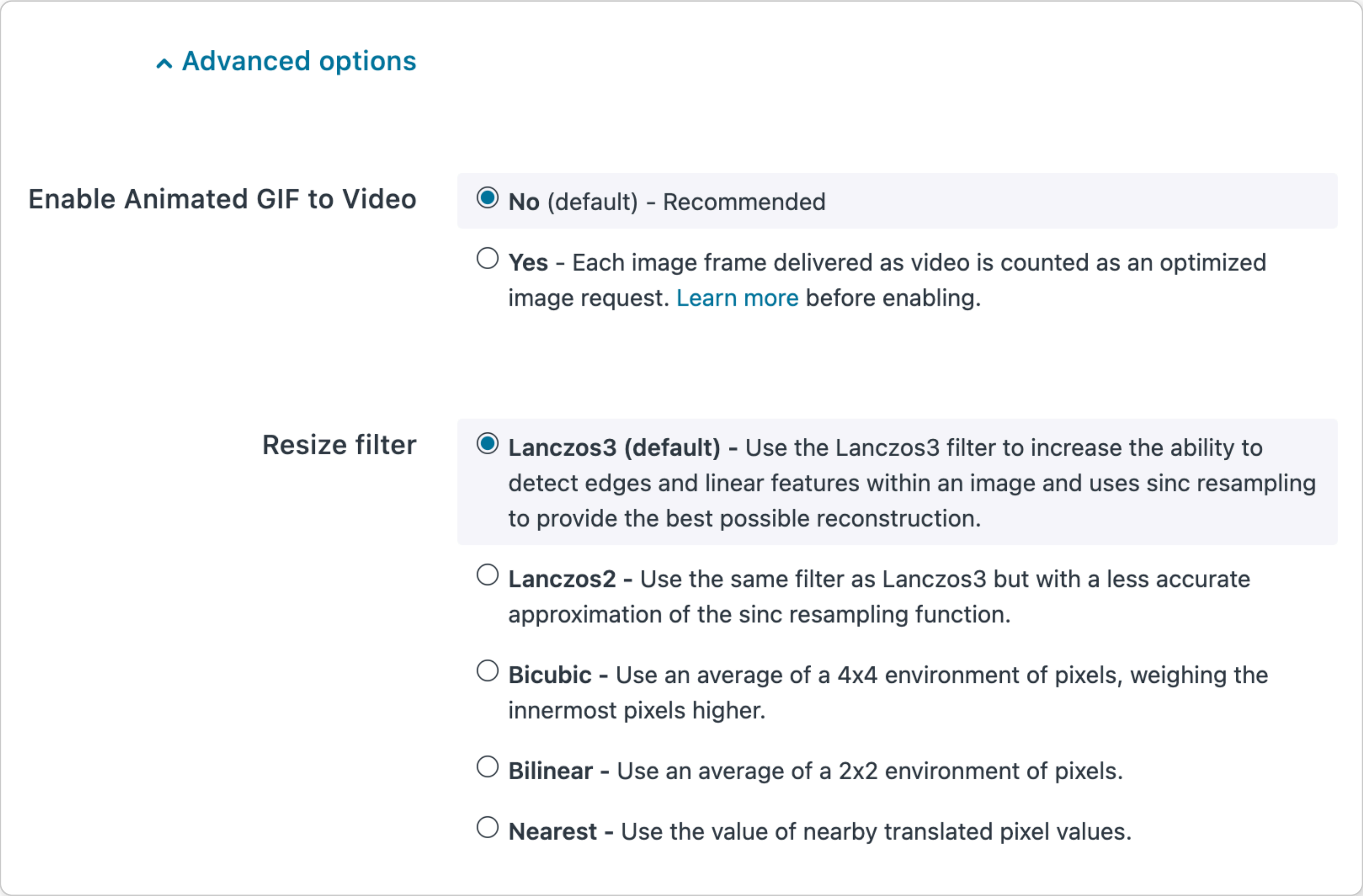
Allow upscaling?
● **No (default) -** Recommended.
○ **Yes -** Allow upscaling.

› Advanced options: Enable Animated GIF to Video, Resize filter

6. Adjust the **Edit default settings** as follows:
    - From the **Auto WebP** controls, leave the settings at their default or select **Yes** to convert images to the WebP format in browsers that support it. When you use the default setting, **No**, Fastly uses the image file type instead.
    - In the **Default WebP (lossy) quality** field, leave the settings at their default or enter the compression level for lossy file-formatted images. Fastly uses `85` for the default quality but you can specify any whole number between `1` and `100`.
    - From the **Default JPEG format** controls, leave the settings at their default or select the JPEG type to use when delivering the image. By default, Fastly sets the JPEG type to **Auto** to deliver images with the output type matching the input type. You can also select **Baseline** to display the image line by line starting from top left and going to the bottom right, or **Progressive** to display a blurry image that becomes clear as it loads.
    - In the **Default JPEG quality** field, leave the settings at their default or enter the compression level for quality of lossy file formats. Fastly uses `85` for the default quality but you can specify any whole number between `1` and `100`.

  - From the **Allow upscaling** controls, leave the settings at their default or select **Yes** to return images larger than the original source file so they can fit the requested dimensions.

7. Click **Advanced options**.



Advanced options

Enable Animated GIF to Video
● **No** (default) - Recommended
○ **Yes** - Each image frame delivered as video is counted as an optimized image request. Learn more before enabling.

Resize filter
● **Lanczos3 (default) -** Use the Lanczos3 filter to increase the ability to detect edges and linear features within an image and uses sinc resampling to provide the best possible reconstruction.
○ **Lanczos2 -** Use the same filter as Lanczos3 but with a less accurate approximation of the sinc resampling function.
○ **Bicubic -** Use an average of a 4x4 environment of pixels, weighing the innermost pixels higher.
○ **Bilinear -** Use an average of a 2x2 environment of pixels.
○ **Nearest -** Use the value of nearby translated pixel values.

8. Adjust the **Advanced options** settings as follows:

  - From the **Enable Animated GIF to Video** controls, leave the settings at their default or select **Yes** to enable Animated GIF to video functionality. Each video frame will be counted and charged as an image optimizer request.

  - From the **Resize filter** controls, select the image quality filter to use when resizing and generating new images to use a higher or lower number of pixels. By default, Fastly uses the **Lanczos3** filter. You can also choose **Lanczos2**, **Bicubic**, **Bilinear**, and **Nearest**.

Using advanced image settings

To go beyond the basic image optimization and transformation settings in the Fastly web interface, you must change your existing image URLs by adding query string parameters. For example, if your image source existed at `http://www.example.com/image.jpg`, you would need to add `?<PARAMETER=VALUE>` to the end of the URL to create the proper query string structure for Fastly to transform the image.

Our Fastly IO documentation describes each of the available image transformations in detail and includes the exact pattern you can add to URLs, along with a description and example of how to use each parameter and its values. These examples perform transformations and optimizations on our `www.fastly.io/image.jpg` URL so you can see exactly how they work before you change your image URLs. Additionally, it provides details you should know before you start adding Fastly IO query strings to your existing image URLs.

Debugging

Running into problems? See our details on image optimization debugging for more information.

What's next

Start experimenting with Fastly IO by following along with Introduction to Fastly Image Optimizer, a step-by-step tutorial that shows you how to set up Fastly IO for a real website. It builds on the concepts introduced in Introduction to Fastly's CDN, and it guides you through the steps of optimizing the images for Taco Labs, the static website we used as an example in Introduction to Fastly's CDN.

This article describes a product that may use third-party cloud infrastructure to process or store content or requests for content. For more information, check out our cloud infrastructure security and compliance program.

* * *

Category: Observability

These articles provide information about monitoring your services via the web interface.

About alerts

🔗 /en/guides/about-alerts

When an observed metric either exceeds or falls below a threshold, the Alerts feature displays a notification in the Fastly control panel and sends messages to your custom integrations (e.g., Slack channels and webhooks). From the Alerts pages, you can view triggered and resolved alert instances and manage your alert definitions and integrations.

Before you begin

The Alerts feature is disabled by default and can be purchased by contacting sales@fastly.com. Once enabled, any user with the appropriate roles and permissions will be able to manage alerts for Fastly services and historic stats metrics.

Viewing firing and resolved alert instances

To view details for an alert that is firing or resolved, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Alerts** > **Overview**.
3. Click on an alert instance that is firing or resolved. The details page for that alert appears.

* * *

How alerting works

/en/guides/how-alerting-works

The Alerts feature displays a notification in the Fastly control panel and sends messages to your custom integrations (e.g., Slack channels and webhooks) when a metric:

- exceeds a static threshold
- is below a static threshold
- exceeds a rate-based threshold
- is below a rate-based threshold

Alerting when a metric exceeds a static threshold

The Alerts feature supports alerting above a threshold. When creating an alert definition, you can define an above threshold based condition by setting the **Trigger an alert when the metric is** menu to `Above`. This alert will fire when the observed metric goes above the specified threshold during an evaluation period at any one minute polling interval. The alert will resolve when there are zero one minute intervals in the evaluation period that exceed the threshold.

Above a static threshold alert example

Let's say you want to be notified in your #website-alerts Slack channel when the number of 5XX responses exceed 1,000 on a single service or all your services. When configuring the alert definition, you choose whether to report on a single service or an aggregate of all services. Then, you set:

- the **Data source** menu to `Historic Stats`
- the **Metric** menu to `status_5xx`
- the **Trigger an alert when the metric is:** menu to `Above`
- the **Threshold** field to `1000`
- the **Evaluation period** menu to `5 minutes`
- the **Integration(s)** menu to `website-alerts`

After saving the alert definition, the system starts evaluating the observed metric in 5 minute rolling time windows (the Evaluation period). During a polling minute in one of those evaluation periods, the observed metric goes above 1,000 (the Threshold):

Evaluation period minute	Number of 5XX responses	Threshold exceeded
1	988	No
2	765	No
3	1021	Yes
4	843	No

Evaluation period minute	Number of 5XX responses	Threshold exceeded
5	1000	No

At the end of the fifth minute of that evaluation period, an alert is triggered because the observed metric went above the threshold during the third minute of the evaluation period. Notifications are also sent. Specifically, the Fastly application indicates that an alert has been triggered via the Alert bell icon and a message is sent to the #website-alerts Slack channel.

That alert then resolves at the end of this evaluation period because there are zero one minute intervals in the evaluation period that exceed the threshold:

Evaluation period minute	Number of 5XX responses	Threshold exceeded
1	576	No
2	674	No
3	516	No
4	577	No
5	732	No

Alerting when a metric is below a static threshold

The Alerts feature supports alerting below a threshold. When creating an alert definition, you can define a below threshold based condition by setting the **Trigger an alert when the metric is** menu to `Below`. This alert will fire when the observed metric remains below the specified threshold for two consecutive minutes in an evaluation period. The alert will resolve when there aren't two consecutive minutes in an evaluation period that are below the threshold. We look for two consecutive minutes in an evaluation period because network latency occasionally delays metric aggregation across POPs, and we want to ensure accuracy and limit false positives.

Below a static threshold alert example

Let's say you want to be notified in your #website-alerts Slack channel when the number of 2XX responses is below 500 on a single service or all your services. When configuring the alert definition, you choose whether to report on a single service or an aggregate of all services. Then, you set:

- the **Data source** menu to `Historic Stats`
- the **Metric** menu to `status_2xx`
- the **Trigger an alert when the metric is:** menu to `Below`
- the **Threshold** field to `500`
- the **Evaluation period** menu to `5 minutes`
- the **Integration(s)** menu to `website-alerts`

After saving the alert definition, the system starts evaluating the observed metric in 5 minute rolling time windows (the Evaluation period). During two consecutive polling minutes in one of those evaluation periods, the observed metric is below 500 (the Threshold):

Evaluation period minute	Number of 2XX responses	Below the threshold
1	1098	No
2	1254	No
3	372	Yes
4	415	Yes
5	500	No

At the end of the fifth minute of that evaluation period, an alert is triggered because the observed metric was below the threshold during the third and forth minutes of the evaluation period. Notifications are also sent. Specifically, the Fastly application indicates that an alert has been triggered via the Alert bell icon and a message is sent to the #website-alerts Slack channel.

That alert then resolves at the end of this evaluation period because there aren't two consecutive minutes in the evaluation period that are below the threshold:

Evaluation period minute	Number of 2XX responses	Below the threshold
1	1108	No
2	285	Yes
3	1284	No
4	1065	No
5	396	Yes

Alerting when a metric exceeds a rate-based threshold

The Alerts feature supports alerting above a rate of all requests. When creating an alert definition, you can define an above rate-based condition by selecting an error rate or success rate metric to observe and setting the **Trigger an alert when the metric is** menu to `Above`. This alert will fire when the observed metric goes above the specified percentage of overall requests during an evaluation period at any one minute polling interval. The alert will resolve when there are zero one minute intervals in the evaluation period that exceed the threshold.

Above a rate-based threshold alert example

Let's say you want to be notified in your #website-alerts Slack channel when the number of 4XX responses exceeds 5% of all responses sent by a single service or all your services. When configuring the alert definition, you choose whether to report on a single service or an aggregate of all services. Then, you set:

- the **Data source** menu to `Historic Stats`
- the **Metric** menu to `4XX Error Rate`
- the **Trigger an alert when the metric is:** menu to `Above`
- the **Threshold** menu to `5%`
- the **Evaluation period** menu to `5 minutes`

- the **Integration(s)** menu to `website-alerts`

After saving the alert definition, the system starts evaluating the observed metric in 5 minute rolling time windows (the Evaluation period). During a polling minute in one of those evaluation periods, the observed metric goes above 5% (the Threshold):

Evaluation period minute	Number of 4XX responses	Total number of requests	Percentage of overall requests	Threshold exceeded
1	0	988	0%	No
2	13	765	1.67%	No
3	64	1021	6.27%	Yes
4	14	843	1.66%	No
5	3	1000	0.3%	No

At the end of the fifth minute of that evaluation period, an alert is triggered because the observed metric went above the threshold during the third minute of the evaluation period. Notifications are also sent. Specifically, the Fastly application indicates that an alert has been triggered via the Alert bell icon and a message is sent to the #website-alerts Slack channel.

That alert then resolves at the end of this evaluation period because there are zero one minute intervals in the evaluation period that exceed the threshold:

Evaluation period minute	Number of 4XX responses	Total number of requests	Percentage of overall requests	Threshold exceeded
1	19	576	3.30%	No
2	30	674	4.45%	No
3	14	516	2.71%	No
4	0	577	0%	No
5	0	732	0%	No

Alerting when a metric is below a rate-based threshold

The Alerts feature supports alerting below a rate of all requests. When creating an alert definition, you can define a below rate-based condition by selecting an error rate or success rate metric to observe and setting the **Trigger an alert when the metric is** menu to `Below`. This alert will fire when the observed metric goes below the specified percentage of overall requests during an evaluation period at any one minute polling interval. The alert will resolve when there are zero one minute intervals in the evaluation period that exceed the threshold.

Below a rate-based threshold alert example

Let's say you want to be notified in your #website-alerts Slack channel when the Success Rate, that is, the percentage of responses with non-5XX codes, falls below 80% on a single service or all your services. When configuring the alert definition, you choose whether to report on a single service or an aggregate of all services. Then, you set:

- the **Data source** menu to `Historic Stats`

- the **Metric** menu to `Success Rate`
- the **Trigger an alert when the metric is:** menu to `Below`
- the **Threshold** menu to `80%`
- the **Evaluation period** menu to `5 minutes`
- the **Integration(s)** menu to `website-alerts`

After saving the alert definition, the system starts evaluating the observed metric in 5 minute rolling time windows (the Evaluation period). During a polling minute in one of those evaluation periods, the observed metric goes below 80% (the Threshold):

Evaluation period minute	Success Rate	Below the threshold
1	99%	No
2	79%	Yes
3	75%	Yes
4	95%	No
5	90%	No

At the end of the fifth minute of that evaluation period, an alert is triggered because the observed metric was below the threshold during two consecutive minutes in the evaluation period. Notifications are also sent. Specifically, the Fastly application indicates that an alert has been triggered via the Alert bell icon and a message is sent to the #website-alerts Slack channel.

That alert then resolves at the end of this evaluation period because there aren't two consecutive minutes in the evaluation period that are below the threshold:

Evaluation period minute	Success Rate	Below the threshold
1	90%	No
2	95%	No
3	99%	No
4	99%	No
5	89%	No

* * *

Managing alert definitions

/en/guides/managing-alert-definitions

An alert definition is a configuration rule that defines when an alert should be triggered and how notifications should be handled.

Limitations and considerations

The maximum number of alert definitions that can exist at any one time per customer is 50.

About alert definitions

An alert definition is a configuration rule that defines when an alert should be triggered and how notifications should be handled. Specifically, it outlines:

- the observed service, data pipeline, and metric.
- the set of conditions that must be met for an alert to trigger.
- the custom destinations for alert messages (e.g., Slack channels and webhooks).

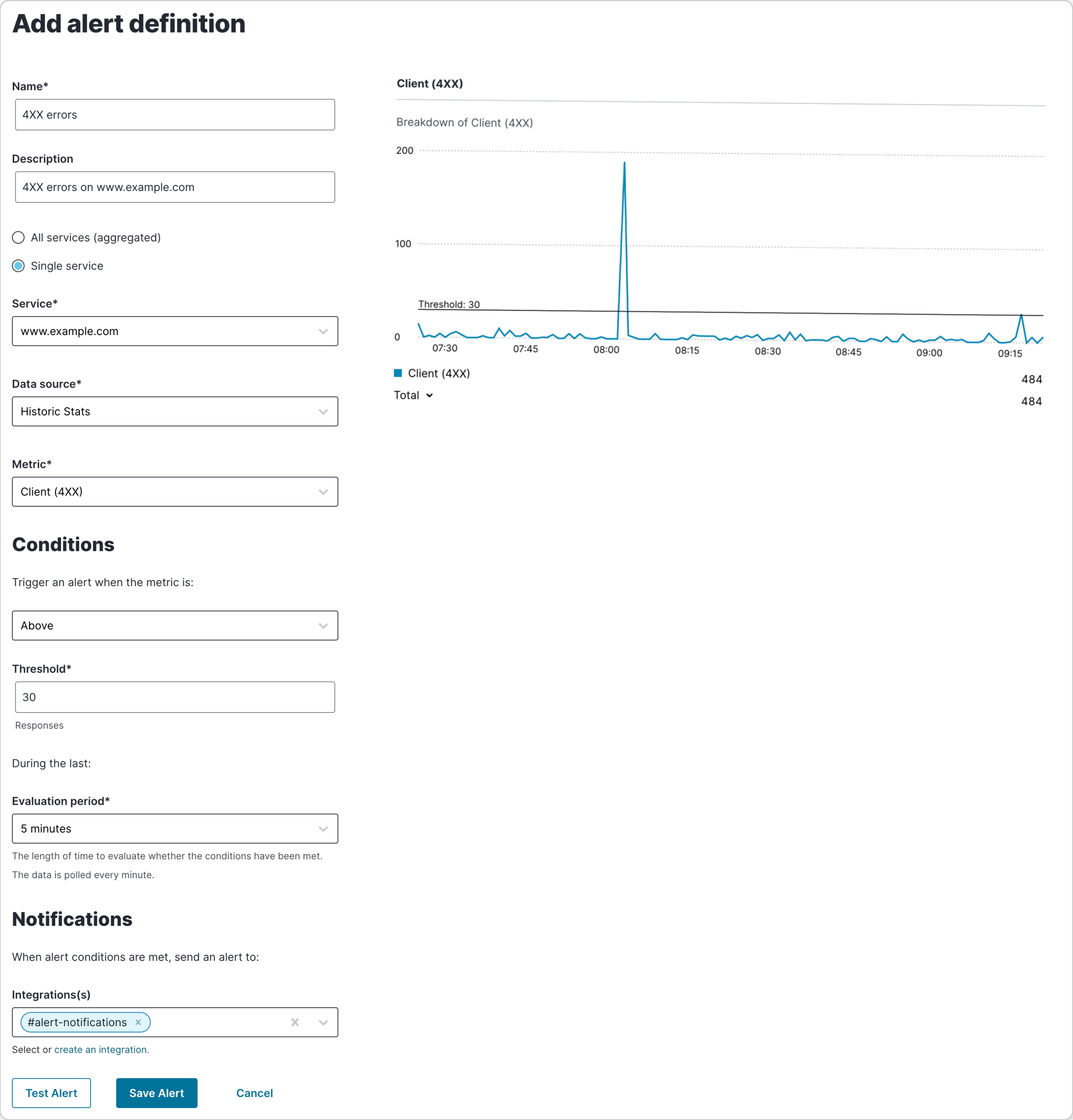
The condition set defines the threshold, the evaluation period, and whether the alert should trigger when the metric exceeds or remains below the threshold or rate. Specifically:

- A threshold is a static number or a percentage of all requests that the observed metric must either be above or below for the alert to trigger.
- The evaluation period is the number of minutes during which the observed metric is evaluated every minute to determine if the metric has gone above or remained below the specified threshold.

Creating alert definitions

To create an alert definition, start by navigating to the Add alert definition page using one of these methods:

- On the **Overview** page in the **Firing** section, click **Add alert definition**.
- On the **Alert definitions** page, click **Add alert definition**.



Add alert definition

Name*
4XX errors

Description
4XX errors on www.example.com

○ All services (aggregated)
● Single service

Service*
www.example.com

Data source*
Historic Stats

Metric*
Client (4XX)

Conditions

Trigger an alert when the metric is:
Above

Threshold*
30
Responses

During the last:

Evaluation period*
5 minutes
The length of time to evaluate whether the conditions have been met.
The data is polled every minute.

Notifications

When alert conditions are met, send an alert to:

Integrations(s)
#alert-notifications ×
Select or create an integration.

Client (4XX)
Breakdown of Client (4XX)
200
100
Threshold: 30
0
07:30
07:45
08:00
08:15
08:30
08:45
09:00
09:15
🟦 Client (4XX)
484
Total
484
Test Alert
Save Alert
Cancel

Next, fill out the general alert fields as follows:

1. In the **Name** field, enter the name of the alert.
2. In the **Description** field, enter a description of the alert definition.
3. Leave the **All services (aggregated)** value selected if you want to monitor all your services or select **Single service** if you want to monitor a specific service.
4. If you selected **Single service**, fill out the following fields:
    - From the **Service** menu, select a service.

  - From the **Data Source** menu, select a data pipeline. Data pipeline options include Historic Stats, Origin Inspector, and Domain Inspector.

Next, define the metric that you want to monitor:

1. If you're monitoring all services or selected `Historic Stats` from the **Data source** menu then, from the **Metric** menu, select the metric that you want to monitor.
2. If you selected `Domain Inspector` from the **Data source** menu, fill out the following fields:
    - From the **Domains** menu, select the domains that you want to monitor.
    - From the **Metric** menu, select the metric that you want to monitor.
3. If you selected `Origin Inspector` from the **Data source** menu, fill out the following fields:
    - From the **Origins** menu, select the origins that you want to monitor.
    - From the **Metric** menu, select the metric that you want to monitor.

Once you've done that, define the conditions that will trigger the alert:

If you selected an error rate or success rate alert type:

1. From the **Trigger an alert when the metric is:** menu, select whether the alert should be triggered when the metric is above or below a percentage of overall requests.
2. From the **Threshold** menu, select a percentage of overall requests that the metric must be above or below for the alert to trigger.
3. From the **Evaluation period** menu, select the number of minutes during which the observed metric is evaluated every minute to determine if the metric is above or below the specified threshold. Options include 5 minutes, 15 minutes, and 30 minutes.

For all other alert types:

1. From the **Trigger an alert when the metric is:** menu, select whether the alert should be triggered when the metric is above or below the threshold.
2. In the **Threshold** field, enter the number that the metric must be above or below for the alert to trigger. To set this field, you can also click and drag the threshold line on the graph.
3. From the **Evaluation period** menu, select the number of minutes during which the observed metric is evaluated every minute to determine if the metric is above or below the specified threshold. Options include 5 minutes, 15 minutes, and 30 minutes.

Next, optionally specify where alert notifications should be sent:

1. *(Optional)* From the **Integrations(s)** menu, select the notification channels for the alert.
2. *(Optional)* Click **Test Alert**. A test notification is sent to the notification channel you selected.

Finally, click **Save Alert** to save the alert. The Alert definitions page appears.

Editing alert definitions

To edit an alert definition, start by navigating to the Edit alert definition page using one of these methods:

- On the **Overview** page in the **Firing** section, click an instance of an alert definition that you want to edit and then click **Edit alert definition**.

- On the **Alert definitions** page, click the pencil to the right of the alert definition that you want to edit.

Next, update and save the alert definition:

1. Fill out the **Edit alert definition** fields as follows:
    - In the **Name** field, enter the name of the alert.
    - In the **Description** field, enter a description of the alert definition.
    - If the value of the **Data source** menu is `Domain Inspector` then, from the **Domains** menu, select the domains that you want to monitor.
    - If the value of the **Data source** menu is `Origin Inspector` then, from the **Origins** menu, select the origins that you want to monitor.
    - From the **Metric** menu, select the metric that you want to monitor.
    - From the **Trigger an alert when the metric is:** menu, select whether the alert should be triggered when the metric is above or below the threshold.
    - In the **Threshold** field, enter the number that must be above or below for the alert to trigger. To set this field, you can also click and drag the threshold line on the graph.
    - From the **Evaluation period** menu, select the number of minutes during which the observed metric is evaluated every minute to determine if the metric is above or below the specified threshold. Options include 5 minutes, 15 minutes, and 30 minutes.
    - From the **Integration(s)** menu, optionally select or remove notification channels for the alert.
2. *(Optional)* Click **Test Alert**. A test notification is sent to the notification channel you selected.
3. Click **Save Alert** to save the alert.

Deleting alert definitions

From the **Alert definitions** page, you can delete alert definitions by clicking the trash to the right of the alert definition that you want to delete. The alert definition is removed from the Alert definitions page.

* * *

Managing alert integrations

/en/guides/managing-alert-integrations

Alert integrations are external channels that receive notifications when alerts are triggered. When creating an alert definition, you specify the alert integrations that should be used. Multiple alert definitions can use the same alert integration.

Accessing alert definitions

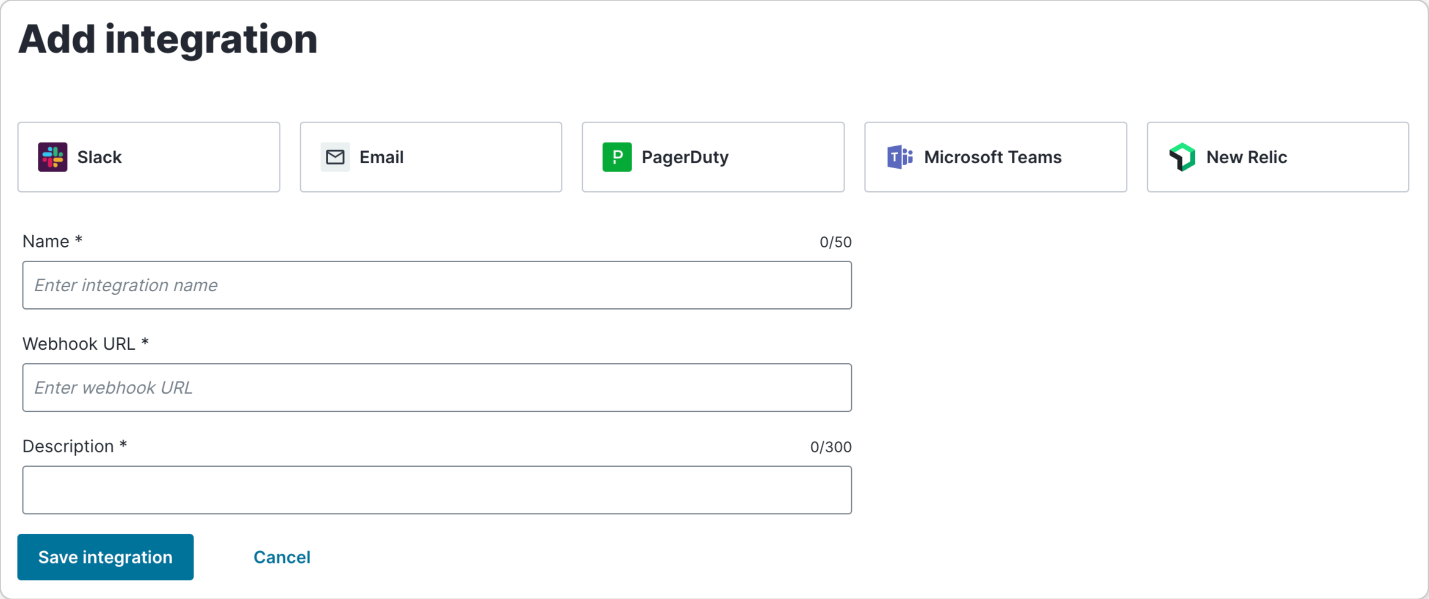
To view a list of your alert definitions, follow these steps:

1. Log in to the Fastly web interface.

2. Go to **Observability** > **Alerts** > **Definitions**. The Alert definitions page appears.

Creating alert integrations

To create an alert integration, follow these steps:



Add integration

Slack
Email
PagerDuty
Microsoft Teams
New Relic
Name *
0/50
Enter integration name
Webhook URL *
Enter webhook URL
Description *
0/300
Save integration
Cancel

1. In Slack, add an incoming webhook and copy the URL for the webhook.

2. Log in to the Fastly web interface.

3. Go to **Observability** > **Alerts** > **Integrations**.

4. Click **Add alert integration**.

5. Fill out the **Add integration** controls as follows:
   - Click **Slack**.
   - In the **Name** field, enter the name of the integration.
   - In the **Webhook URL** field, enter the URL of the webhook you created in the first step.
   - In the **Description** field, enter a description of the notification channel.

6. Click **Save integration**. The integration is added to the Alert integrations page.

1. Log in to the Fastly web interface.

2. Go to **Observability** > **Alerts** > **Integrations**.

3. Click **Add alert integration**.

4. Fill out the **Add integration** controls as follows:
   - Click **Email**.
   - In the **Name** field, enter the name of the integration.
   - In the **Email** field, enter the email address that should receive the alert notifications.

  - In the **Description** field, enter a description of the integration.

5. Click **Save integration**. An email asking you to confirm the address you entered in the **Email** field is sent to the email address you entered.

6. Navigate to the relevant email account and open the confirmation email from Fastly.

7. Click **Confirm email address**. Your email address is verified and the integration is added to the Alert integrations page.

1. In PagerDuty, create a new service and copy the integration key. Be sure to set the escalation policy and integration method to **Use Our API Directly**.

2. Log in to the Fastly web interface.

3. Go to **Observability** > **Alerts** > **Integrations**.

4. Click **Add alert integration**.

5. Fill out the **Add integration** controls as follows:
    - Click **PagerDuty**.
    - In the **Name** field, enter the name of the integration.
    - In the **Service API Key** field, enter the integration key you created in the first step.
    - In the **Description** field, enter a description of the notification channel.

6. Click **Save integration**. The integration is added to the Alert integrations page.

1. In Microsoft Teams, add an incoming webhook and copy the URL for the webhook.

2. Log in to the Fastly web interface.

3. Go to **Observability** > **Alerts** > **Integrations**.

4. Click **Add alert integration**.

5. Fill out the **Add integration** controls as follows:
    - Click **Microsoft Teams**.
    - In the **Name** field, enter the name of the integration.
    - In the **Webhook URL** field, enter the URL of the webhook you created in the first step.
    - In the **Description** field, enter a description of the notification channel.

6. Click **Save integration**. The integration is added to the Alert integrations page.

1. In New Relic, create a new License key on the account where you want to ingest data. Make note of both the License key and Account ID.

2. Log in to the Fastly web interface.

3. Go to **Observability** > **Alerts** > **Integrations**.

4. Click **Add alert integration**.

5. Fill out the **Add integration** controls as follows:
    - Click **New Relic**.

- In the **Name** field, enter the name of the integration.
- In the **License key** field, enter the license key you created in the first step.
- In the **Account ID** field, enter the account ID you obtained in the first step.
- In the **Description** field, enter a description of the notification channel.

6. Click **Save integration**. The integration is added to the Alert integrations page.

Deleting alert integrations

To delete an alert integration, follow these steps:

1. From the **Alert integrations** page, click the trash to the right of the alert integration that you want to delete.
2. Click **Confirm and delete**. The alert integration is removed from the Alert integrations page.

* * *

Working with Observability dashboards

/en/guides/working-with-observability-dashboards

From the Observability pages, you can access dashboards that contain collections of metrics represented as charts. Depending on the dashboard, the metrics may relate to all of your services or to the Fastly Deliver or Compute service that you select via the Service menu.

There are two types of dashboards:

- **Platform:** dashboards that provide an overview of all your services or that focus on a single service. You can not modify platform dashboards. Platform dashboards are located on the Service overview, All services, Origins, and Domains pages.
- **Custom:** dashboards that you build to surface relevant metrics for a service. You can view and manage your custom dashboards from the Custom dashboards page. Custom dashboards are disabled by default.

Before you begin

Be sure you know how to access the web interface controls and understand basic information about the Observability pages before learning about custom dashboards.

Limitations and considerations

The platform dashboards included on the Observability pages depend on your access permissions and the products and features you may have purchased. Your access permissions also determine your ability to build custom dashboards.

Custom dashboards are disabled by default and are only available to customers who have purchased an Observability package. Once enabled, you are limited to 100 custom dashboards per account and only you can view the custom dashboards that you've built. Custom dashboards are saved in local storage. If you clear your browser's cache, the custom dashboards you created will be deleted, and you will no longer be able to access them via the Custom dashboards page.

Accessing dashboards

To view a dashboard, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Observability** in the navigation sidebar and select an option.
3. *(Optional)* Use the Observability page controls to change the display of the metrics on the selected dashboard.

Sharing a dashboard

To share a specific dashboard, follow these steps:

1. Navigate to the dashboard that you want to share.
2. Click **Options** and then select **Copy link**. The link is saved to your computer's clipboard.
3. Manually share the link outside of the Fastly app.

To access a shared dashboard, you must have the appropriate permissions.

Working with custom dashboards

From the Custom dashboards page, you can create and delete custom dashboards.

Creating a custom dashboard

To create a custom dashboard, follow these steps:

1. On the **Custom dashboards** page, click **Create dashboard**.
2. In the **Dashboard Name** field, enter the name of your custom dashboard.
3. Click **Save**. The Custom dashboards page appears and displays the custom dashboard that you created.

After creating a dashboard, you can add charts to it.

Deleting a custom dashboard

To delete a custom dashboard, follow these steps:

1. Navigate to the custom dashboard that you want to delete.
2. Click **Options** and then select **Edit dashboard**.
3. Click the pencil next to the dashboard name.
4. Click **Delete** and then **Delete**. The custom dashboard is deleted and is no longer listed in the custom dashboard menu.

Working with dashboard charts

You can surface relevant metrics on custom dashboards by adding charts that highlight meaningful data and deleting existing charts that aren't needed anymore.

Expanding a chart on a platform dashboard

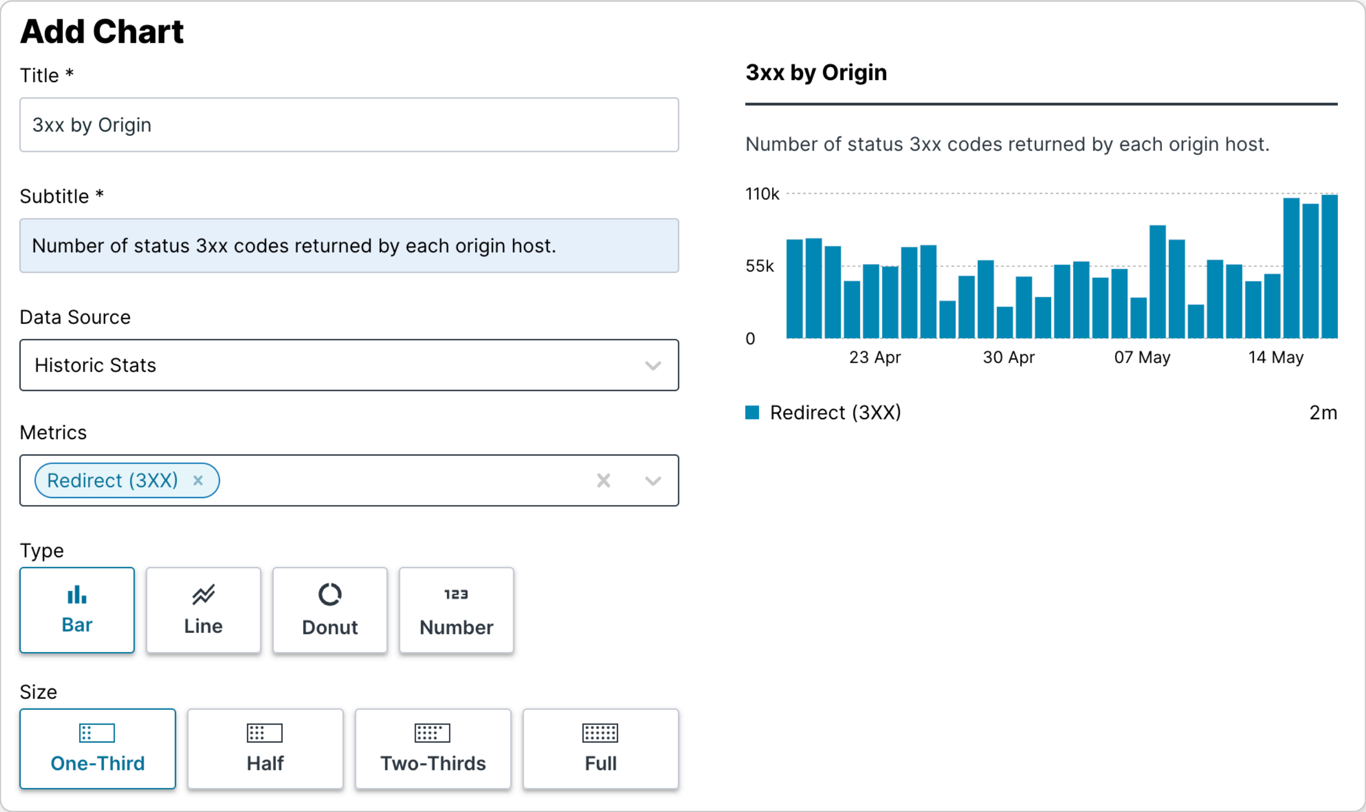
To expand a chart on a platform dashboard and display detailed information about the metric being tracked:

1. Navigate to a platform dashboard. Platform dashboards are located on the Service overview, All services, Origins, and Domains pages.
2. Click the double-sided arrow in the upper right corner of the chart that you want to expand. The Metric Details page for the chart appears.

Adding a chart to a custom dashboard

To add a chart to a custom dashboard, follow these steps:

1. Navigate to the custom dashboard that you want to add a chart to.
2. Click **Options** and then select **Edit dashboard**.
3. Click **Add Chart**.



Add Chart

Title *: 3xx by Origin

Subtitle *: Number of status 3xx codes returned by each origin host.

Data Source: Historic Stats

Metrics: Redirect (3XX)

Type: Bar

Size: One-Third

Type
Bar
Line
Donut
123
Number
Size
One-Third
Half
Two-Thirds
Full
3xx by Origin
Number of status 3xx codes returned by each origin host.
110k
55k
0
23 Apr
30 Apr
07 May
14 May
Redirect (3XX)
2m

4. Fill out the Add Chart fields as follows:
    - In the **Title** field, enter the name of the chart.
    - In the **Subtitle** field, enter a brief description of the chart.
    - From the **Data Source** menu, select a data pipeline. Data pipeline options include Historic Stats, Original Inspector, and Domain Inspector.

  - From the **Metrics** menu, select the type of data the chart should report.
  - From the **Type** menu, select an option to control how the data is displayed.
  - From the **Calculation Method** menu, select the single metric the chart should report. This field only appears when the Type field is set to Donut or Number.
  - From the **Size** menu, select an option to control the width of the chart.

  A preview of the title and subtitle appears above the form, and a preview of the chart appears below the form.

5. Click **Save**. The chart is added to the dashboard.
6. Click **Exit Edit Mode** to return to view mode for the dashboard.

Deleting a chart from a custom dashboard

To delete a chart from a custom dashboard, follow the steps below:

1. Navigate to the custom dashboard that you want to add a chart to.
2. Click **Options** and then select **Edit dashboard**.
3. Click the trash to the right of the chart that you want to delete. The chart is removed from the dashboard.
4. Click **Exit Edit Mode** to return to view mode for the dashboard.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

Category: Performance

These articles describe how to adjust the performance of Fastly's services beyond standard configuration methods.

Debugging with mtr

🔗 /en/guides/debugging-with-mtr

We think the mtr tool offers a great way to test network speed, evaluate performance, and perform connection diagnostics. The mtr tool combines traceroute and ping programs in a single network diagnostic tool. The program's source and installation instructions live in GitHub.

While mtr provides a number of practical uses for network engineering needs, the following command works well:

```
$ mtr -c 20 -w -r www.example.com
```

Be sure to replace `www.example.com` with the hostname of the domain you're working with. The command will generate the network hops to the destination you specify, any packet loss experienced, and aggregate connection statistics.

For example, if we wanted to test the network connection from Fastly's San Francisco office to the CDN, we would use the above command for `www.fastly.com`. The following output would appear:

```
$ mtr -c 20 -w -r www.fastly.com
Start: Mon Feb  2 15:27:20 2015
HOST: test-local-machine.local                Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- 10.100.20.2                            0.0%    20    2.1   2.2   1.6   4.3   0.5
  2.|-- ge-4-3-4.mpr4.sfo7.us.zip.zayo.com     0.0%    20    2.3   2.4   1.8   5.2   0.6
  3.|-- ae5.cr2.sjc2.us.zip.zayo.com           0.0%    20    4.6   6.5   2.9  35.3   7.7
  4.|-- ae10.mpr4.sjc7.us.zip.zayo.com         0.0%    20    4.7   4.8   3.6  14.5   2.3
  5.|-- be6461.ccr21.sjc03.atlas.cogentco.com  5.0%    20    5.1   5.9   4.2  15.3   2.6
  6.|-- fastly-inc.edge2.sanjose3.level3.net   0.0%    20    5.0   4.7   4.2   8.2   0.8
  7.|-- ???                                   100.0    20    0.0   0.0   0.0   0.0   0.0
  8.|-- 23.235.47.184                          0.0%    20    4.7  14.3   3.8  74.6  20.3
```

* * *

Enabling automatic compression

/en/guides/enabling-automatic-compression

Compression can help you reduce the size of your assets so traffic can flow faster. You can use Fastly to compress data automatically on our edge servers.

Much of the data delivered by Fastly can be compressed using static compression, especially text-based formats like HTML, JavaScript, and CSS. Static compression fetches content from origin, compresses it according to the user's requested format (either Brotli or gzip), and then caches it.

Dynamic compression, on the other hand, compresses content just before responses are delivered to the client and is used on content that can’t be cached. For example, dynamic compression can compress content unique to an end user.

Limitations and caveats

Keep in mind the following:

- **The API endpoint name is currently /gzip.** API documentation for automatic compression currently appears in the documentation for the gzip endpoint. The documented gzip API fully supports Brotli compression.
- **Custom VCL can override your web interface and API compression settings.** If you enable compression using custom VCL, that configuration will be used instead of the setting in the web interface.

When considering the general behavior of Fastly's compression feature, keep in mind the following:

- **Compression is only supported for content served uncompressed from your origin.** To use automatic compression, ensure your content doesn't require decompression before compression is applied.
- **Compression is currently incompatible with Segmented Caching.** To use automatic compression ensure Segmented Caching is disabled.

Enabling static compression

This section demonstrates how to use the Fastly web interface to enable and disable static compression for cacheable content. To enable compression, select a compression format and then set up a compression policy.

Selecting a compression format

To enable compression, start by selecting a compression format using the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Scroll to the **Compression** area. From the **Select compression format** options, do one of the following:
   - Select **Use Brotli compression when available** to use Brotli compression by default for browsers that support it. Once you've enabled compression and selected Brotli as the format, we will use it by default for responses any time a client's Brotli-capable browser sends us the appropriate Accept-Encoding header. Brotli is the preferred compression format for multiple Accept-Encoding values where gzip is also an option.
   - Select **Use gzip compression only** to avoid using Brotli and only use gzip as the compression format for this service.

Setting up a compression policy

Once you've selected the compression format, you'll need to set up a compression policy. Decide whether to:

- enable the default compression policy to compress content in files with `css`, `js`, `html`, `eot`, `ico`, `otf`, `ttf`, `json`, or `svg` file extensions.
- set up an advanced compression policy to customize the content and conditions for compression and specify exactly which types of content are compressed and the conditions under which this compression occurs.

Enabling the default compression policy

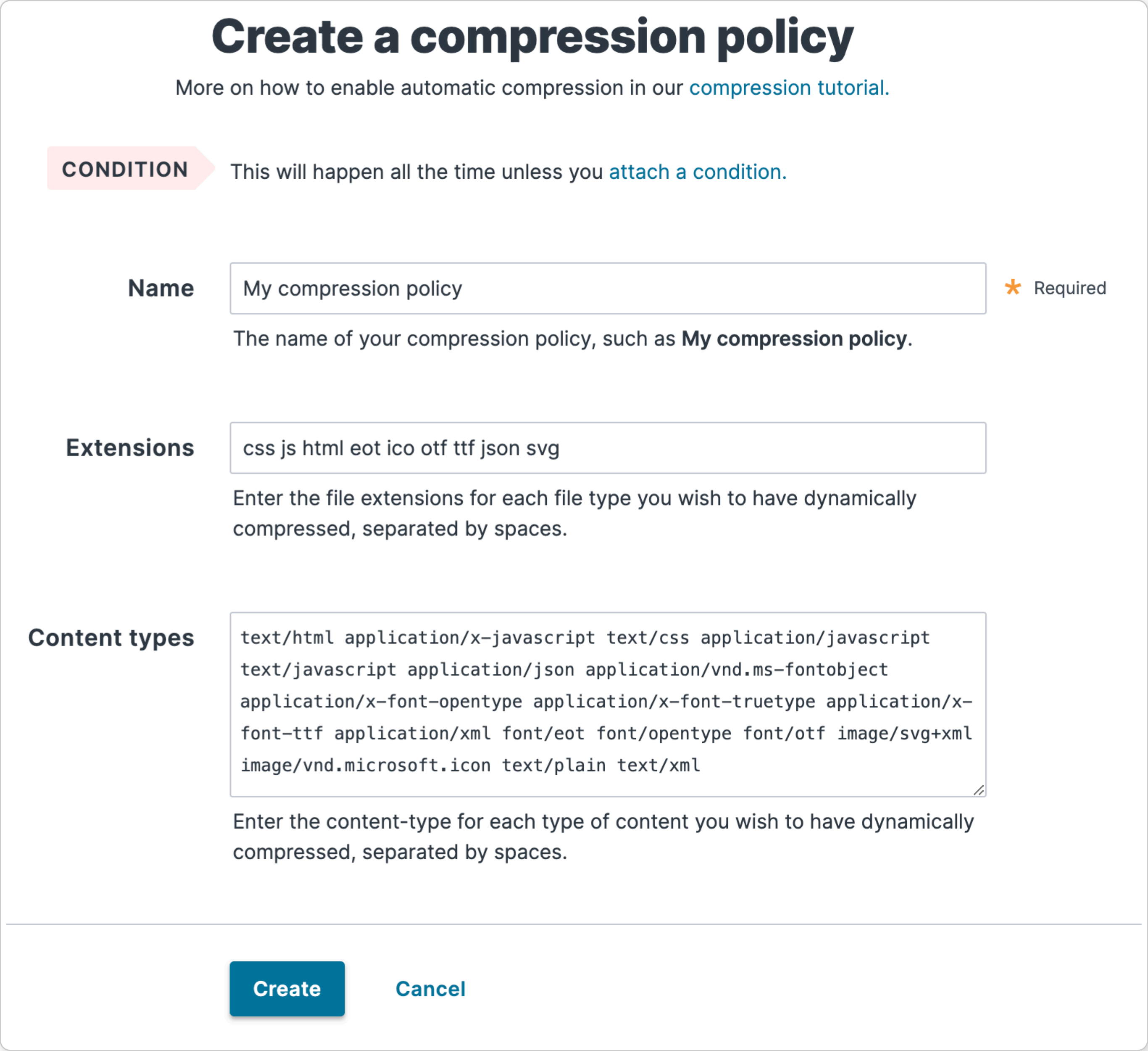
To enable the default compression policy, follow these steps:

1. Click the **Default compression policy** switch to enable compression using the format you selected to compress content using the default list of file extensions.
2. Click **Activate** to deploy your configuration changes.

Setting up an advanced compression policy

To set up an advanced compression policy, follow these steps:

1. Click **Set up advanced compression**.
2. Click **Override these defaults**.



Create a compression policy

More on how to enable automatic compression in our compression tutorial.

This will happen all the time unless you attach a condition.

Name: My compression policy

The name of your compression policy, such as **My compression policy**.

Extensions: css js html eot ico otf ttf json svg

Enter the file extensions for each file type you wish to have dynamically compressed, separated by spaces.

Content types:

```
text/html application/x-javascript text/css application/javascript
text/javascript application/json application/vnd.ms-fontobject
application/x-font-opentype application/x-font-truetype application/x-
font-ttf application/xml font/eot font/opentype font/otf image/svg+xml
image/vnd.microsoft.icon text/plain text/xml
```

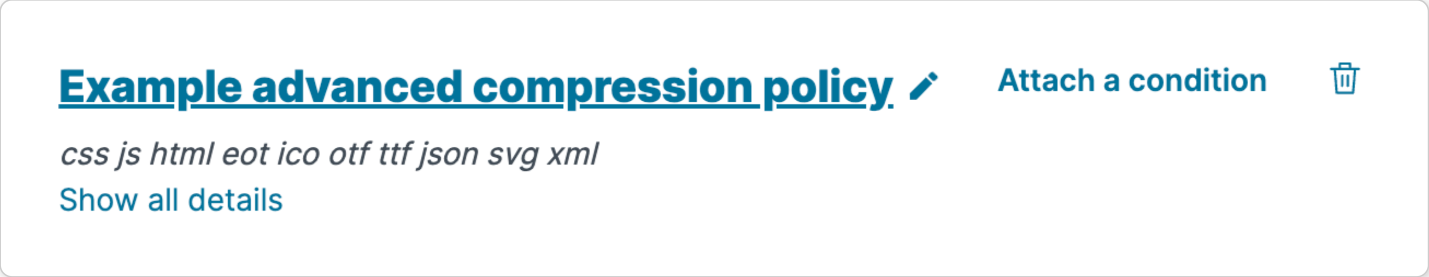
Enter the content-type for each type of content you wish to have dynamically compressed, separated by spaces.

3. Fill out the **Create a compression policy** fields as follows:
    - In the **Name** field, enter a human-readable name for your new compression policy.
    - In the **Extensions** field, enter the file extension for each file type to be dynamically compressed, separated by spaces. Only enter the three- or four-letter string representing the file extension.
    - In the **Content types** field, enter the content-type for each type of content you wish to have dynamically compressed, separated by spaces. Do not use regular expressions.
4. Click **Create**. The new compression policy appears.
5. Click **Activate** to deploy your configuration changes.

Disabling static compression

If you're currently using the default compression policy, you can disable compression by clicking the **Default compression policy** switch to display `OFF`.

If you're currently using an advanced compression policy, you can disable compression by deleting it using the trash icon to the right of the policy's title:



Example advanced compression policy

css js html eot ico otf ttf json svg xml

Automatic normalization

Because compression is one of the most common reasons to vary output based on a request header, Fastly will normalize the value of `Accept-Encoding` on incoming requests. The modified header will be set to a single encoding type, or none, and will reflect the best compression scheme supported by the browser.

Specifically, we run the following steps on inbound requests:

1. If the `User-Agent` matches a pattern for browsers that have problems with compressed responses, remove the `Accept-Encoding` header
2. Else if the `Accept-Encoding` header includes the string `br`, set the entire value to the string `br`
3. Else if the `Accept-Encoding` header includes the string `gzip`, set the entire value to the string `gzip`
4. Else if the `Accept-Encoding` header includes the string `deflate`, set the entire value to the string `deflate`
5. Else remove the `Accept-Encoding` header

Where this normalization process has changed the header value, the original value is made available in the custom header `Fastly-Orig-Accept-Encoding`.

Enabling dynamic compression

Fastly's Dynamic Content Compression feature allows you to compress dynamic content that wouldn't benefit from caching, like video streaming manifests and dynamic content from Compute applications. Once compressed, you are also able to apply dynamic web content assembly techniques like ESI. Our guide to delivering compressed content through Fastly discusses how to enable compression using custom VCL or the Fastly API via the X-Compress-Hint header and includes workflows for compressing dynamic content.

* * *

Enabling HTTP/3 for Fastly services

/en/guides/enabling-http3-for-fastly-services

This guide describes how to enable HTTP/3 for your Fastly services.

About HTTP/3

HTTP/3 uses a web transport protocol standard called QUIC that you can offer to end user clients as an optional upgrade from HTTP/1.1 and HTTP/2 connections. End user clients will initially connect to your Fastly service using an earlier version of HTTP and will only attempt to use HTTP/3 if the Fastly service offers it.

Unlike the TCP transport protocol used by earlier versions of HTTP, QUIC requires that all connections be secured and uses TLS 1.3 to secure them. Like TLS 1.3, QUIC also supports the optional 0-RTT feature to help reduce the latency of resumed connections.

Prerequisites

To use HTTP/3 on Fastly's edge cloud services, you will need:

- a Fastly user account assigned the role of superuser
- relevant domains you will be using for HTTP/3 added to a properly configured Fastly service
- access privileges to modify DNS records for the domains you will be using

Limitations and key behaviors

- Fastly currently only supports HTTP/3 for end user connections, as that is where we expect the primary benefits of these protocols will be seen. We do not support HTTP/3 between Fastly and your origin servers.
- The QUIC transport protocol used by HTTP/3 requires all connections to be secured using TLS 1.3. This means that when you configure domains to offer HTTP/3, you are also configuring them to offer TLS 1.3 to clients using HTTP/1.1 or HTTP/2.

Enabling HTTP/3 for Fastly services

To enable HTTP/3 for your Fastly services, start by configuring HTTP/3 on your domains and then configure your services to offer HTTP/3 on client connections.

Configuring HTTP/3 on your domains

To configure HTTP/3 on a new domain, refer to Setting up TLS 1.3 for a new domain.

To configure HTTP/3 on an existing domain, complete the following:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. Find the domain on which you plan to offer HTTP/3 and use the **TLS configuration and DNS details** column to verify whether HTTP/3 has been enabled.
    - If the column has a value of either `HTTP/3 & TLS v1.3` or `HTTP/3 & TLS v1.3 + 0RTT`, then the domain already supports HTTP/3. Continue to the next section to offer HTTP/3 support for traffic to that domain from your service.

  - If the column does not display an HTTP/3 value, follow the steps to enable TLS 1.3 for this domain before proceeding to the next section.

Offering HTTP/3 from your Fastly service

HTTP/3 is designed as an optional upgrade to end user client connections. This means that end user clients will initially connect to your Fastly service using an earlier version of HTTP and will only attempt to use HTTP/3 if the Fastly service offers it.

Use the **HTTP/3** switch to configure your service to offer HTTP/3:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. Click the **HTTP/3** switch to configure your service to offer HTTP/3.
6. Click **Activate** to deploy your configuration changes.

If you are configuring your service using VCL directly (either via regular VCL snippets or custom VCL methods), you can use the following VCL in the receive (`vcl_recv`) sub-routine to configure your service to offer HTTP/3:

```
h3.alt_svc();
```

Sending your HTTP/3 traffic to Fastly

Unless you use Fastly's dedicated IP addresses, then as a final step to enabling HTTP/3, you must ensure the DNS records of your domains are routing users to the correct HTTP/3 enabled Fastly addresses:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. Click **View details** for the domain you would like to route to Fastly. The domain's details page appears.

The CNAME records section contains the value for CNAME records. You can prefix the value with `dualstack` to enable IPv6 support (e.g., `dualstack.<letter>.sni.global.fastly.net`).

For apex domains (e.g., `example.com`), the A records section contains the global anycast IP addresses.

Testing client compatibility with your Fastly service

If you want test a browser or other client for HTTP/3 support, Fastly has created a publicly available HTTP/3 test page to verify client support. Using the client, navigate to https://http3.is and the resulting page will let you know whether or not HTTP/3 was successfully used to request the page. If the request did not use HTTP/3, the page will also let you know which IETF draft versions are currently being used by Fastly. You can use that information to update your client or client configurations to use one of those supported versions.

Keep in mind that even if correctly configured, the first request from a browser (or its first request after a time-out period set by the browser's developers) to any HTTP/3 enabled web site will always use a lower version of HTTP because it has not yet seen the HTTP/3 offer. If you don't see the HTTP/3 success page on your first request, be sure to perform a reload in the browser to give it an opportunity to use HTTP/3 for subsequent requests.

Serving HTTP/3 traffic

Once you've configured your services appropriately, requests made to the domains on those services should be capable of supporting HTTP/3 if they are being made from a client that supports these protocols.

If the QUIC connections and HTTP/3 requests are successful, the clients will continue using them for all subsequent requests and connections for those domains. Should problems occur with the QUIC connections or HTTP/3 requests, clients are expected to automatically fall back to a standard HTTP/1.1 or HTTP/2 connection over TCP. You should verify that your chosen client implements this fallback if this is a priority for you.

Monitoring your HTTP/3 traffic

You can monitor HTTP/3 requests using Fastly's Real-Time Log Streaming feature and Observability features. We have also added a number of VCL variables specifically related to HTTP/3 and QUIC.

Disabling HTTP/3

You can disable HTTP/3 support on a Fastly service by either activating a previous service version where HTTP/3 is not enabled or by creating a new service configuration version and disabling the HTTP/3 switch before activation.

* * *

Failure modes with large files

/en/guides/failure-modes-with-large-objects

If you haven't enabled Segmented Caching, you may encounter the following failure modes when working with large objects.

TIP

Large objects can cause inefficiencies in content distribution networks. The Segmented Caching feature can help resolve these large object inefficiencies. Fastly recommends enabling Segmented Caching on services that will be serving large objects. Without Segmented Caching enabled, object size limits for your account depend on when you become a Fastly customer:

- If you created your account on or after June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum object size of 20 MB.
- If you created your account prior to June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum cacheable object size of 2 GB for requests without Streaming Miss or 5 GB for requests with Streaming Miss.

Maximum object size limits

If the response from the origin has a `Content-Length` header field that exceeds the maximum object size, Fastly will generate a `503 Response object too large` response to the client. If no `Content-Length` header field is returned, Fastly will start to fetch the response body. If, while fetching the response body, we determine that the object exceeds the maximum object size, we will generate a status `503 Response object too large` response to the client.

If no `Content-Length` header field is present and the Streaming Miss feature is enabled, Fastly will stream the content back to the client. However, if while streaming the response body Fastly determines that the object exceeds the maximum cacheable object size, it will terminate the client connection abruptly. The client will detect a protocol violation, because it will see its connection close without a properly terminating 0-length chunk.

Origin read failures

If reading the response body from the origin fails or times out, the problem will be reported differently depending on whether or not you've enabled Streaming Miss to act when the object is fetched. Without Streaming Miss, a 503 response will be generated. With Streaming Miss, however, it is already too late to send an error response since the header will already have been sent. In this case, Fastly will abruptly terminate the client connection and the client will detect a protocol violation. If the response was chunked, the client will see its connection close without a properly terminating 0-length chunk. If Content-Length was known, the client will see the connection close before the number of bytes given.

* * *

Fastly's service status

/en/guides/fastlys-service-status

Fastly continuously monitors the performance and status of our global network. We constantly monitor the health of all its related services and post regular notices about them to customers. Any time we schedule maintenance, upgrade hardware, re-route traffic, or experience service interruptions, we post updates on our service status page at https://www.fastlystatus.com/. A complete, filterable history of past events is also available at https://www.fastlystatus.com/incidents/.

IMPORTANT

If you are experiencing problems and do not see a posted status page notification or if you have subscribed to status notifications and have not received an email or an SMS message, contact Customer Support for assistance.

Managing Fastly service status notifications

Fastly provides customers and the general public with multiple options for subscribing to status notifications.

TIP

Fastly offers syndicated status updates at https://www.fastlystatus.com/rss/ for use with any RSS-compatible feed readers.

Subscribing to notifications

You can subscribe to notifications and directly control which systems trigger those notifications by following these steps:

1. Using any browser, navigate to https://www.fastlystatus.com/.
2. Do one of the following:
    - To subscribe to service status notifications, click **Subscribe** at the top of the browser window.
    - To subscribe to notifications about a specific incident, click the title of any current or past incident and then click **Subscribe** at the bottom of the detailed incident report that appears.
3. When the subscription window appears, decide how you would like to be notified of updates. Specifically:
    - In the **Email Address** field, enter the email address to which email notifications, if any, should be sent.
    - From the **Phone Number** menu, select your country code and enter your telephone number to which SMS notifications, if any, should be sent.
4. Click the **Subscribed to** area to select which Fastly components should trigger notifications.

    ✓ TIP

    Specific details about each of the components appear when you click the arrow menu at the top right of the **Subscribed to** area. Use this menu to control the level of detail about each component for which notifications could be sent. For example, you could elect to receive only notifications for Fastly's Customer Services components and specifically choose only to receive notifications related to the availability of the docs.fastly.com website.

5. *(Optional)* In the **Webhook** field, enter the URL of a custom webhook to subscribe to the Fastly status page and automate the delivery of notifications to a system you control.
6. Click **Subscribe**. Activation messages will be sent to each subscription method you select with details allowing you to confirm your subscription.

Adjusting notification preferences

You can adjust your notification preferences at any time by following these steps:

1. Click **Manage Subscription** at the bottom of any status email Fastly has sent (or your original subscription email if you still have it).
2. *(Optional)* In the **Subscribed to** area, select or deselect which Fastly components should trigger notifications, if any.

    ✓ TIP

    Specific details about each of the components appear when you click the arrow menu at the top right of the **Subscribed to** area. Use this menu to control the level of detail about each component for which notifications could be sent. For example, you could elect to receive only notifications for Fastly's

Customer Services components and specifically choose only to receive notifications related to the availability of the docs.fastly.com website.

3. *(Optional)* In the **Notification Preferences** area, select or deselect the specific incident type for each method of contact that should trigger notifications. For example, you could elect to receive no notifications via email or webhook, only via SMS messages to a mobile phone.

4. Click **Update** to update your notification preferences.

Unsubscribing from notifications

You can unsubscribe from all notifications at once or from specific types of notifications individually.

Unsubscribing from all notification types at once

To unsubscribe from all notification types at once:

1. Click **Manage Subscription** at the bottom of any status email Fastly has sent (or your original subscription email if you still have it).

2. Click **Unsubscribe** at the bottom of the window.

3. Click **OK** to confirm your request to unsubscribe.

Unsubscribing immediately from specific types of notifications

To unsubscribe immediately from:

- email notifications, click **Unsubscribe** at the bottom of any status email.
- SMS notifications, reply **STOP** to any status message you receive.
- webhook notifications, use the process for adjusting notification preferences and specifically deselect the **Webhooks** option before clicking **Update**.

Viewing current and past status information

The main Fastly service status page includes details about the current status of Fastly's network and incidents or events, as well as summarized views of the past status and event history.

IMPORTANT

Fastly does not post current event statuses that could signal to bad actors that their attempts to negatively impact customers or Fastly directly have been successful. During suspected or confirmed events, our support team will contact you directly with details.

What the status indicators tell you

Indicator icons appear on the incident history grid and on all status page notifications that confirm the current operating status of the network and service components or various events that could be occurring at a specific time.

Status indicator icon	Status description
✅	**Normal.** Everything is operating normally. No events or incidents related to network or service components have been identified or announced.
ℹ	**Informational.** We've posted information for general awareness. No network and service components are impacted.
🕒	**Maintenance.** We've scheduled or are actively performing maintenance on the Fastly global network or one of its service components.
❗	**Degraded.** We've identified something that has caused a degradation of a portion of the Fastly network or service component. Customer services are being delivered but at reduced capacity.
❌	**Unavailable.** A portion of our network or a specific service component is not currently operational or available.
❓	**Investigating.** We've observed or received a report of something that's impacting the performance of our network or a service component. We are actively investigating this event.
✓	**Identified.** An investigation identified the cause of an event or problem and a fix for it is being implemented.
📈	**Monitoring.** A fix has been deployed for an event or problem that is now resolved. The impacted network or service components are now stable and being monitored for continued health.

Viewing summaries of current and past events

Summary posts related to the current status of any active incidents appear at the top of the Fastly service status page. Fastly also keeps track of all past statuses, displaying the most recent of them on the main status page, which is organized into the following sections:

- **Current status.** When incidents or events are in progress, you'll see them summarized in the Current Status area. This section is hidden when all conditions are normal.
- **History Grid.** The history grid displays the individual, daily status of each of Fastly's network and service components and their sub-components for today and the previous six days using indicator icons for each date. Click the plus icon next to any component to expand the view to display indicators for all sub-components. Click the arrow icons to the right and left of the dates to view previous weeks at a glance.
- **Scheduled Maintenance.** The Scheduled Maintenance area displays summaries of active and future maintenance work. Click on a maintenance post title to view its full history and details. To review past maintenance work, click **Incident History** at the top of the page.

Older, past history for incidents and scheduled maintenance as well as full details about each specific event are available at https://www.fastlystatus.com/incidents/.

Viewing detailed information about specific events

You can display the full report details of these events by clicking on their title in any area of the service status page. Detailed event and incident reports include the current status of the event or incident, the components impacted, and the full text of all updates that have been published. Timestamps appear for every update posted.

You can subscribe to notifications about any specific incident or event, past or present, by clicking the Subscribe button at the bottom of any detailed report.

Filtering displayed history information

On the main service status page, the filter controls appear immediately below the history grid. With them you can control which components, statuses, and types of history event information to display in both the history grid and the past history summaries for incident and maintenance events.

- **Component filters.** Filter by component by selecting the specific component from the **All Components** menu. Filter specific sub-components by clicking the plus icon next to the component name and then selecting a specific sub-component from the items that appear.
- **Status filters.** Filter by status type by selecting one of them from the **Any Status** menu.
- **Event type filters.** Filter by event type by selecting one of them from the **All Types** menu.

On the full incident history page you can also filter by event date and order in addition to component, status, and event type:

- **Date filters.** Filter by date by clicking the **Date range** selector, selecting the specific date or date range for events, and then clicking **Apply**.
- **Event filters.** Filter by order by clicking the **Sort by** menu and then selecting either **Newest** or **Oldest**.

> Interested in an authenticated version of the Fastly Service Status page for your team? If you're a delivery customer with Enterprise Support and either a Technical Account Manager or Security Technical Account Manager, contact your account manager or contact support to see if your company meets the qualification criteria for this option.

* * *

HTTP/2 server push

/en/guides/http2-server-push

HTTP/2 server push allows you to set up rules that enable Fastly to pre-emptively load and then send responses to an HTTP/2-compliant client before that client requests them. You can initiate an HTTP/2 server push via a response header or VCL function.

Server push with the `link` response header

Fastly recognizes `link` headers with the preload keyword sent by an origin server and pushes the designated resource to a client. For example, this `link` response header triggers an HTTP/2 push:

```
link: </assets/jquery.js>; rel=preload; as=script
```

We support multiple `link` headers and multiple assets in one `link` header:

```
link: </assets/jquery.js>; rel=preload; as=script, </assets/base.css>; rel=preload; as=s le
```

Additional attributes used in the `link` header can further control server push and how the header itself is handled. If no additional attributes are included, the `link` header will trigger server push and be forwarded to the client:

```
link: </assets/jquery.js>; rel=preload; as=script
```

If used with the `nopush` directive, the header will *not* trigger a push and will be passed as is to the client:

```
link: </assets/jquery.js>; rel=preload; as=script; nopush
```

If used with the `x-http2-push-only` directive, the header will trigger a server push but will be subsequently removed and not forwarded to the client:

```
link: </assets/jquery.js>; rel=preload; as=script; x-http2-push-only
```

The attributes can be mixed and matched if needed:

```
link: </assets/jquery.js>; rel=preload; as=script, </assets/base.css>; rel=preload; as=s le
```

Link headers and Amazon S3 buckets

If you're using an Amazon Simple Storage Service (S3) bucket as your origin server, you can still use `link` headers by applying a cache setting condition like this one:

`set beresp.http.Link = beresp.http.x-amz-meta-Link`

Server push with the `h2.push()` function

Server push can also be triggered with the `h2.push()` VCL function. The asset to be pushed is passed to the function as a parameter. For example:

```
sub vcl_recv {
#FASTLY recv

  if (fastly_info.is_h2 && req.url ~ "^/index.html")
  {
    h2.push("/assets/jquery.js");
  }
}
```

The `h2.push()` function triggers server push as soon as it's called, which removes the need for a `link` header to arrive with a server response. This means assets can be pushed to the client before the response for the request that triggered the push is received from the server, accelerating their delivery.

* * *

Making query strings agnostic

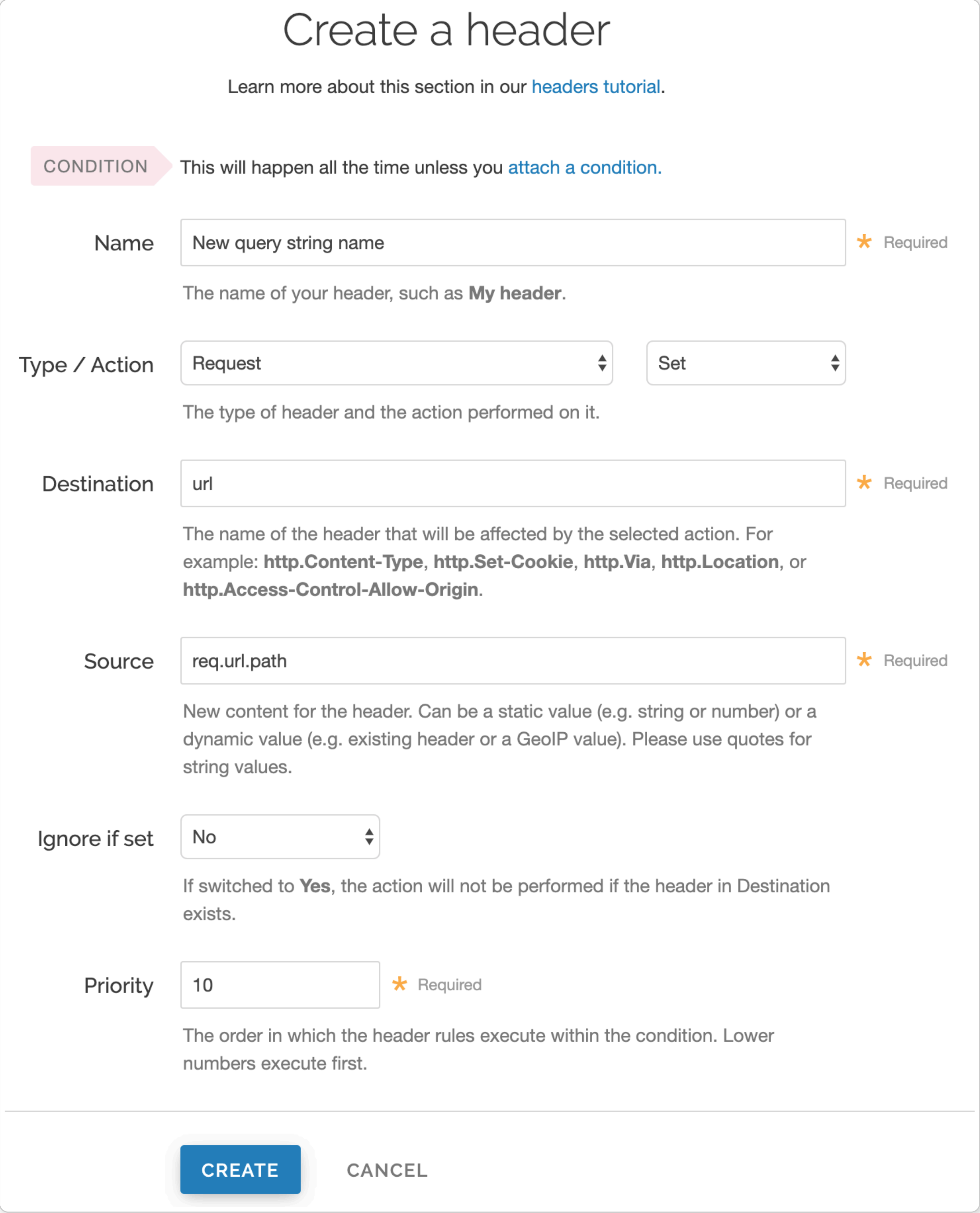
🔗 /en/guides/making-query-strings-agnostic

Under normal circumstances, Fastly would consider these URLs different objects that are cached separately:

- `http://example.com`
- `http://example.com?asdf=asdf`
- `http://example.com?asdf=zxcv`

It is possible, however, to have them all ignore the query string and return the same cached resource.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: New query string name

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: url

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: req.url.path

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter a description for the header (e.g., `New query string name`).
   - From the **Type** menu, select **Request**, and from **Action** menu, select **Set**.

  - In the **Destination** field, enter `url`.
  - In the **Source** field, enter `req.url.path`.
  - From the **Ignore if set** menu, select **No**.
  - Set the **Priority** field to whatever priority you want.
7. Click **Create** to create the new header.
8. Click **Activate** to deploy your configuration changes.

The request will be sent to the origin as a URL without the query string.

For more information about controlling caching, see our documentation on cache freshness.

* * *

Precision Path

/en/guides/precision-path

Fastly can automatically detect and, in real time, route around transient connection problems that occur when fetching content from your origin servers or when delivering content to end users from Fastly's Edge Cloud. Fastly uses different, automated techniques depending on where we observe a connection problem. This page describes two capabilities that Precision Path uses to automatically improve fetching and delivery of content to your users: origin connection rerouting and edge connection rerouting.

Origin connection rerouting

It's critical for Fastly to maintain high performance and reliable access to your origin services to fetch new or updated content when needed. The inability to do so, even temporarily, can result in 5xx HTTP server errors being returned to end users, regardless of how resilient and performant connections may be. Based on historical observations of our worldwide network, we estimate a significant portion of 5xx errors served to end users are due to the impact of transient internet performance issues (commonly called *internet weather*) on the connections between Fastly's point of presence (POP) locations and our customers' origin services. Origin rerouting is designed to address this.

How origin connection rerouting works

Our origin connection rerouting capability monitors Fastly's connections to your origin services for signs of internet weather and, if detected, automatically attempts to route around them so they don't impact your services. The vast majority of the time you won't notice changes to those connections in your Fastly services because there will be no issues to route around. However, when the default route from one of our POPs across the internet to your origins experiences a severe enough problem, it will automatically test other available routes to your origin and, if any successful alternatives are found, will select the most viable alternative route with which to re-establish the connection. All of this will happen before TCP timeouts occur on the end user connection, thereby avoiding 5xx errors being returned to end users.

Requirements

Precision Path's origin connection rerouting only works for origin connections using TLS. If your Fastly service has not been configured to use a TLS connection between Fastly and your origin server, you can enable TLS by following the instructions in our guide on working with hosts.

No special requirements are necessary to take advantage of Fastly's origin connection rerouting. This capability is enabled by default for all Fastly services using TLS for origin and shielding connections.

Many Fastly customers protect their origin services from security threats by implementing either a firewall service or IP address access control lists (ACLs). While this is a good security practice, you could be inadvertently blocking a subset of Fastly IP addresses from accessing your origins if these configurations aren't updated regularly.

To successfully implement origin connection rerouting on your Fastly services, ensure you aren't blocking any Fastly IP addresses. Fastly provides an API for you to obtain a complete list of the IPv4 and IPv6 address ranges owned by Fastly. Any firewalls or ACLs protecting your origin services should be updated to ensure all these IP address ranges are allowed to connect to your origins.

Monitoring

Fastly's real-time log streaming feature can be used to monitor for origin connection rerouting events that may occur on your Fastly services. The following VCL variables can be used to monitor origin connection rerouting activity:

- `beresp.used_alternate_path_to_origin` - This boolean value indicates whether or not the request to origin was made over an alternate route selected by the origin connection rerouting mechanism. Counting the number of true values for this variable over any given time period in your logs will indicate how many times this origin rerouting mechanism has been triggered.
- `beresp.backend.src_ip` - This variable indicates which Fastly source IP was used to make the request to origin. For most connections, this will be one of the most frequently used Fastly server IP addresses used for default routes across the Internet. If the origin rerouting mechanism was triggered by a problem with the original connection attempt over the default route, then this value will show an alternate Fastly server IP address from a pool of IPs reserved for alternate routes between that POP and your origin.
- `beresp.backend.alternate_ips` - This variable lists all the possible source IP addresses available to the origin rerouting mechanism for the Fastly server handling this origin connection. This information can be used to identify the set of valid Fastly IP addresses from which you may see connection attempts to your origin from this Fastly server.

Edge connection rerouting

When delivering content from Fastly to your end users, Fastly tracks the health of every TCP connection. When we observe connection-impacting degradation (e.g., congestion), we automatically switch delivery to a new network path to route around the issue. This automatic mitigation is enabled by default on all of our POPs and applies to all Fastly traffic. No additional configuration is required.

No special requirements are necessary to take advantage of Fastly's edge connection rerouting. All traffic delivered from Fastly's POPs is assessed for connection issues and the fast path failover feature is applied only when necessary.

Support

For more information on Precision Path or its individual features, contact support.

* * *

Serving stale content

/en/guides/serving-stale-content

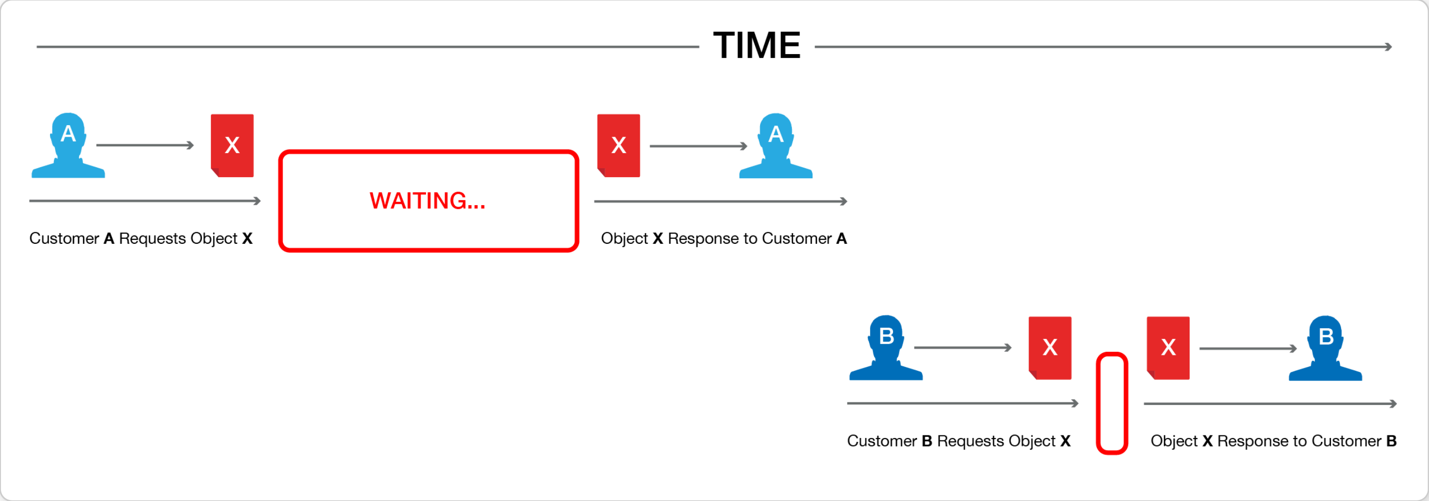
Fastly can optionally serve stale content when there is a problem with your origin server or if new content is taking a long time to fetch from your origin server. For example, if Fastly can't contact your origin server, our POPs will continue to serve cached content when users request it. These features are not enabled by default.

TIP

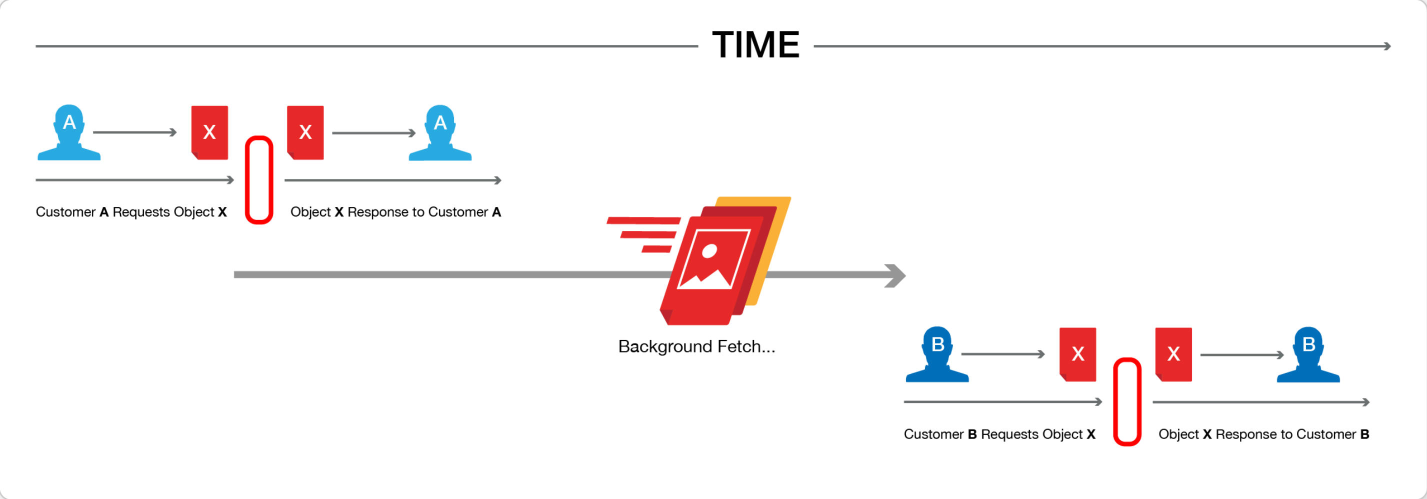
For more information on serving stale content, see our documentation on staleness and revalidation.

Serving old content while fetching new content

Certain pieces of content can take a long time to generate. Once the content is cached it will be served quickly, but the first user to try and access it will pay a penalty.


TIME
A
X
WAITING...
X
A
Customer A Requests Object X
Object X Response to Customer A
B
X
X
B
Customer B Requests Object X
Object X Response to Customer B

This is unavoidable if the cache is completely cold, but if this is happening when the object is in cache and its TTL is expired, then Fastly can be configured to show the stale content while the new content is fetched in the background.


TIME
A
X
Customer A Requests Object X
Object X Response to Customer A
Background Fetch...
B
Customer B Requests Object X
Object X Response to Customer B

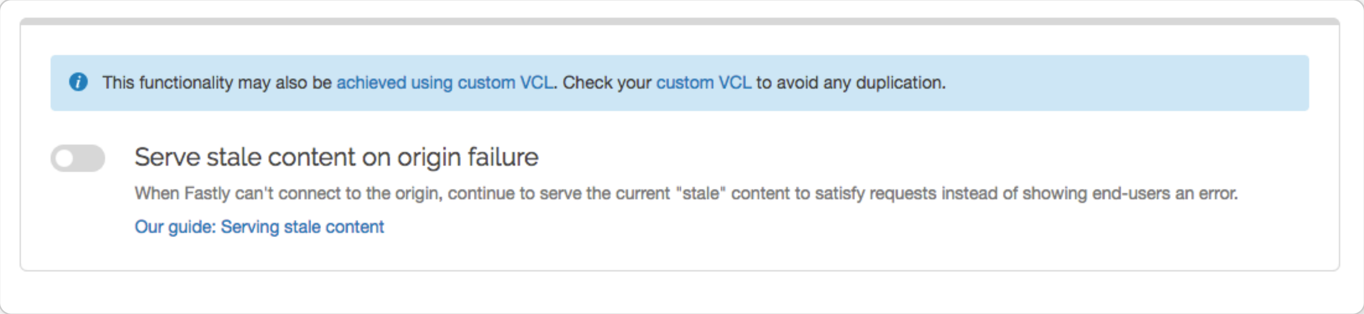
Fastly builds on the behavior proposed in RFC 5861 "HTTP Cache-Control Extensions for Stale Content" by Mark Nottingham, which is under consideration for inclusion in Google's Chrome browser.

Enabling serve stale

> NOTE
>
> If you've already enabled serving stale content on an error via custom VCL, adding this feature via the web interface will set a different stale TTL. To avoid this, check your custom VCL and remove the `stale-if-error` statement before enabling this feature via the web interface.

To enable serving stale content on an error via the web interface for the default TTL period (43200 seconds or 12 hours), follow the steps below.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.



This functionality may also be achieved using custom VCL. Check your custom VCL to avoid any duplication.

Serve stale content on origin failure

When Fastly can't connect to the origin, continue to serve the current "stale" content to satisfy requests instead of showing end-users an error.

Our guide: Serving stale content

5. Click the **Serve stale** switch to automatically enable serving stale content for the default TTL period of 43200 seconds (12 hours).
6. Click **Activate** to deploy your configuration changes.

Manually enabling serve stale

These instructions provide an advanced configuration that allows all three possible origin failure cases to be handled using VCL.

In the context of Varnish, there are three ways an origin can fail:

- The origin can be marked as unhealthy by failing health checks.
- If Varnish cannot contact the origin for any reason, a 503 error will be generated.
- The origin returns a valid HTTP response, but that response is not one we wish to serve to users (for instance, a 503).

The custom VCL shown below handles all three cases. If the origin is unhealthy, the default serve stale behavior is triggered by `stale-if-error`. In between the origin failing and being marked unhealthy, Varnish would normally return 503s. The custom VCL allows us to instead either serve stale if we have a stale copy, or to return a synthetic error page. The error page can be customized. The third case is handled by intercepting all 5XX errors in `vcl_fetch` and either serving stale or serving the synthetic error page.

> WARNING
>
> Do not purge all cached content if you are seeing 503 errors. Purge all overrides stale-if-error and increases the requests to your origin server, which could result in additional 503 errors.

Although not strictly necessary, health checks should be enabled in conjunction with this VCL. Without health checks enabled, all of the functionality will still work, but serving stale or synthetic responses will take much longer while waiting for an origin to timeout. With health checks enabled, this problem is averted by the origin being marked as unhealthy.

The custom VCL shown below includes the Fastly standard boilerplate. Before uploading this to your service, be sure to customize or remove the following values to suit your specific needs:

- `if (beresp.status >= 500 && beresp.status < 600)` should be changed to include any HTTP response codes you wish to serve stale/synthetic for.
- `set beresp.stale_if_error = 86400s;` controls how long content will be eligible to be served stale and should be set to a meaningful amount for your configuration. If you're sending `stale_if_error` in Surrogate-Control or Cache-Control from origin, remove this entire line.
- `set beresp.stale_while_revalidate = 60s;` controls how long the `stale_while_revalidate` feature will be enabled for an object and should be set to a meaningful amount for your configuration. This feature causes Varnish to serve stale on a cache miss and fetch the newest version of the object from origin in the background. This can result in large performance gains on objects with short TTLs, and in general on any cache miss. Note that `stale_while_revalidate` overrides `stale_if_error`. That is, as long as the object is eligible to be served stale while revalidating, `stale_if_error` will have no effect. If you're sending `stale_while_revalidate` in Surrogate-Control or Cache-Control from origin, remove this entire line.
- `synthetic {"<!DOCTYPE html>Your HTML!</html>"};` is the synthetic response Varnish will return if no stale version of an object is available and should be set appropriately for your configuration. You can embed your HTML, CSS, or JS here. Use caution when referencing external CSS and JS documents. If your origin is offline they may be unavailable as well.

```
sub vcl_recv {
```

```
#FASTLY recv

  # Normally, you should consider requests other than GET and HEAD to be uncacheable
  # (to this we add the special FASTLYPURGE method)
  if (req.method != "HEAD" && req.method != "GET" && req.method != "FASTLYPURGE") {
    return(pass);
  }

  # If you are using image optimization, insert the code to enable it here
  # See https://www.fastly.com/documentation/reference/io/ for more information.

  return(lookup);
}

sub vcl_hash {
  set req.hash += req.url;
  set req.hash += req.http.host;
  #FASTLY hash
  return(hash);
}

sub vcl_hit {
#FASTLY hit
  return(deliver);
}


sub vcl_miss {
#FASTLY miss
  return(fetch);
}

sub vcl_pass {
#FASTLY pass
  return(pass);
}

sub vcl_fetch {
  /* handle 5XX (or any other unwanted status code) */
  if (beresp.status >= 500 && beresp.status < 600) {

    /* deliver stale if the object is available */
    if (stale.exists) {
      return(deliver_stale);
    }

    if (req.restarts < 1 && (req.method == "GET" || req.method == "HEAD")) {
      restart;
    }
  }

  /* set stale_if_error and stale_while_revalidate (customize these values) */
  set beresp.stale_if_error = 86400s;
  set beresp.stale_while_revalidate = 60s;

#FASTLY fetch

  /* handle 5XX (or any other unwanted status code) */
  if (beresp.status >= 500 && beresp.status < 600) {

    /* deliver stale if the object is available */
    if (stale.exists) {
      return(deliver_stale);
    }
  }

  # Unset headers that reduce cacheability for images processed using the Fastly image
  if (req.http.X-Fastly-Imageopto-Api) {
    unset beresp.http.Set-Cookie;
    unset beresp.http.Vary;
  }

  # Log the number of restarts for debugging purposes
  if (req.restarts > 0) {
    set beresp.http.Fastly-Restarts = req.restarts;
  }

  # If the response is setting a cookie, make sure it is not cached
  if (beresp.http.Set-Cookie) {
    return(pass);
  }


  # By default we set a TTL based on the `Cache-Control` header but we don't parse addi
  # like `private` and `no-store`. Private in particular should be respected at the edg
  if (beresp.http.Cache-Control ~ "(?:private|no-store)") {
    return(pass);
  }

  # If no TTL has been provided in the response headers, set a default
  if (!beresp.http.Expires && !beresp.http.Surrogate-Control ~ "max-age" && !beresp.htt
    set beresp.ttl = 3600s;

    # Apply a longer default TTL for images processed using Image Optimizer
    if (req.http.X-Fastly-Imageopto-Api) {
      set beresp.ttl = 2592000s; # 30 days
      set beresp.http.Cache-Control = "max-age=2592000, public";
    }
  }

  return(deliver);
}
```

```


sub vcl_error {
#FASTLY error

  /* handle 503s */
  if (obj.status >= 500 && obj.status < 600) {

    /* deliver stale object if it is available */
    if (stale.exists) {
      return(deliver_stale);
    }

    /* otherwise, return a synthetic */

    /* include your HTML response here */
    synthetic {"<!DOCTYPE html><html>Replace this text with the error page you would li
    return(deliver);
  }

}


sub vcl_deliver {
#FASTLY deliver
  return(deliver);
}

sub vcl_log {
#FASTLY log
}
```

Adding headers

After you add the custom VCL, you can finish manually enabling serving stale content by adding a `stale-while-revalidate` or `stale-if-error` directive to either the `Cache-Control` or `Surrogate-Control` headers in the response from your origin server. For example:

```
Cache-Control: max-age=600, stale-while-revalidate=30
```

will cache some content for 10 minutes and, at the end of that 10 minutes, will serve stale content for up to 30 seconds while new content is being fetched.

Similarly, this statement:

```
Surrogate-Control: max-age=3600, stale-if-error=86400
```

instructs the cache to update the content every hour (3600 seconds) but if the origin is down then show stale content for a day (86400 seconds).

Alternatively, these behaviors can be controlled from within VCL by setting the following variables in `vcl_fetch`:

```
set beresp.stale_while_revalidate = 30s;
set beresp.stale_if_error = 86400s;
```

Interaction with grace

Stale-if-error works exactly the same as Varnish's grace variable such that these two statements are equivalent:

```
set beresp.grace = 86400s;
set beresp.stale_if_error = 86400s;
```

However, if a grace statement is present in VCL it will override any `stale-if-error` statements in any Cache-Control or Surrogate-Control response headers.

Why serving stale content may not work as expected

Here are some things to consider if Fastly isn't serving stale content:

- **Cache:** Stale objects are only available for cacheable content.
- **VCL:** Setting `req.hash_always_miss` or `req.hash_ignore_busy` variable to `true` invalidates the effect of `stale-while-revalidate`.
- **Shielding:** If you don't have shielding enabled, a POP can only serve stale on errors if a request for that cacheable object was made through that POP before. We recommend enabling shielding to increase the probability that stale content on error exists. Shielding is also a good way to quickly refill the cache after performing a purge all.
- **Requests:** As traffic to your site increases, you're more likely to see stale objects available (even if shielding is disabled). It's reasonable to assume that popular assets will be cached at multiple POPs.
- **Least Recently Used (LRU):** Fastly has an LRU list, so objects are not necessarily guaranteed to stay in cache for the entirety of their TTL (time to live). But eviction is dependent on many factors, including the object's request frequency, its TTL, the POP from which it's being served. For instance, objects with a TTL of 3700s or longer get written to disk, whereas objects with shorter TTLs end up in transient, in-memory-only storage. We recommend setting your TTL to more than 3700s when possible.
- **Purges:** Whenever possible, you should purge content using our soft purge feature. Soft purge allows you to easily mark content as outdated (stale) instead of permanently deleting it from Fastly's caches. If you can't use soft purge, we recommend purging by URL or using surrogate keys instead of performing a purge all.

---

* * *

Streaming Miss

/en/guides/streaming-miss

When fetching an object from the origin, the Streaming Miss feature ensures the response is streamed back to the client immediately and is written to cache only after the whole object has been fetched. This reduces the first-byte latency (the time that the client must wait before it starts receiving the response body).

TIP

Fastly recommends enabling Segmented Caching on services that will be serving large resources. Without Segmented Caching enabled, the resource size limits for your account depend on when you become a Fastly customer:

- If you created your account on or after June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum object size of 20 MB.
- If you created your account prior to June 17, 2020 and haven't enabled Segmented Caching, your Fastly services have a maximum cacheable object size of 2 GB for requests without Streaming Miss or 5 GB for requests with Streaming Miss.

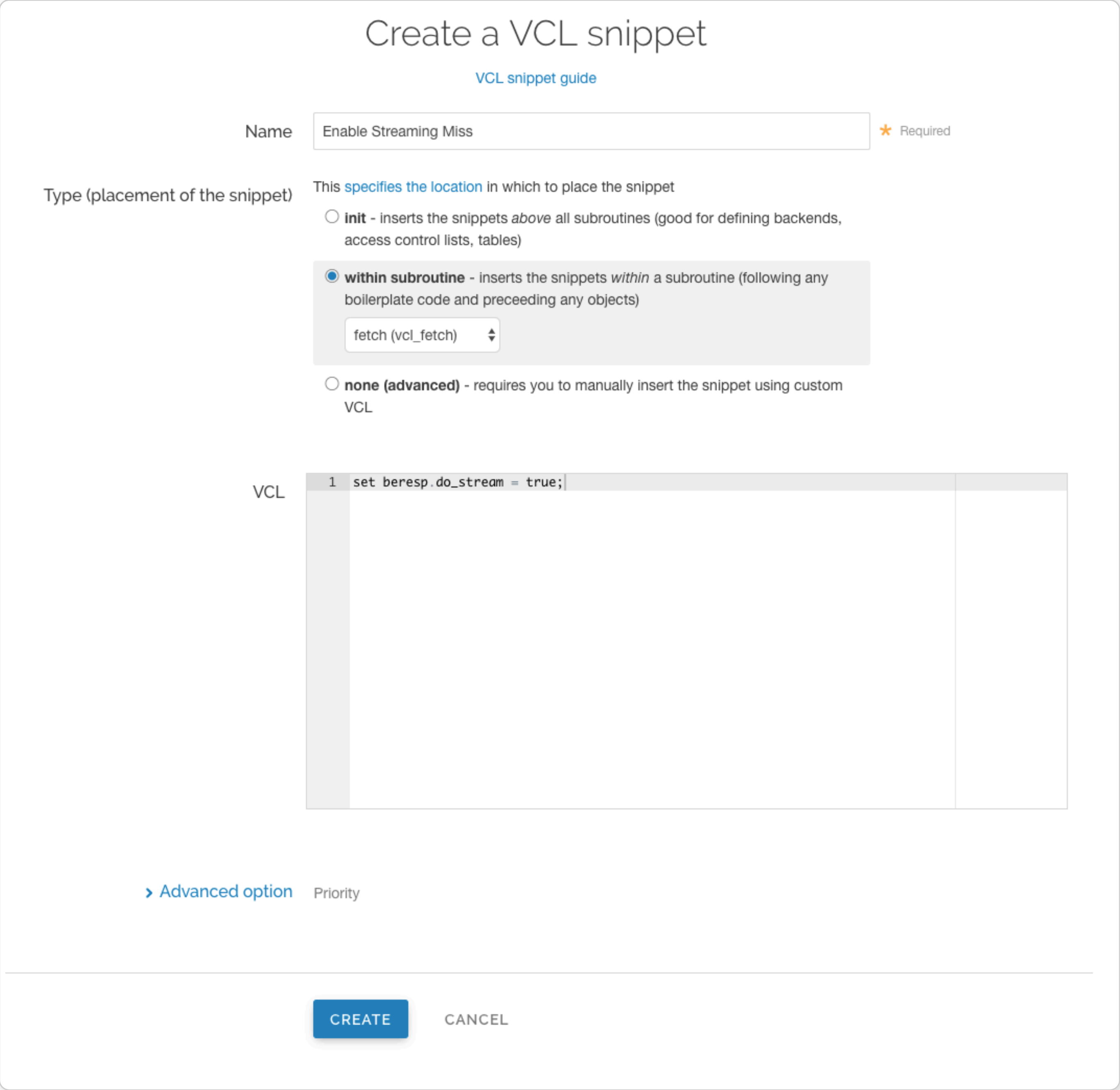
NOTE

If you enable Streaming Miss, be aware that if an error occurs while transferring the response body, Fastly cannot send an error because the headers are already sent to the client. All we can do is truncate the response.

Enable Streaming Miss by creating a VCL Snippet

Enable Streaming Miss by setting `beresp.do_stream` to `true` in `vcl_fetch` with a VCL Snippet using these steps.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create your first VCL snippet**.



Create a VCL snippet

VCL snippet guide

Name: Enable Streaming Miss

Type (placement of the snippet): This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

fetch (vcl_fetch)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

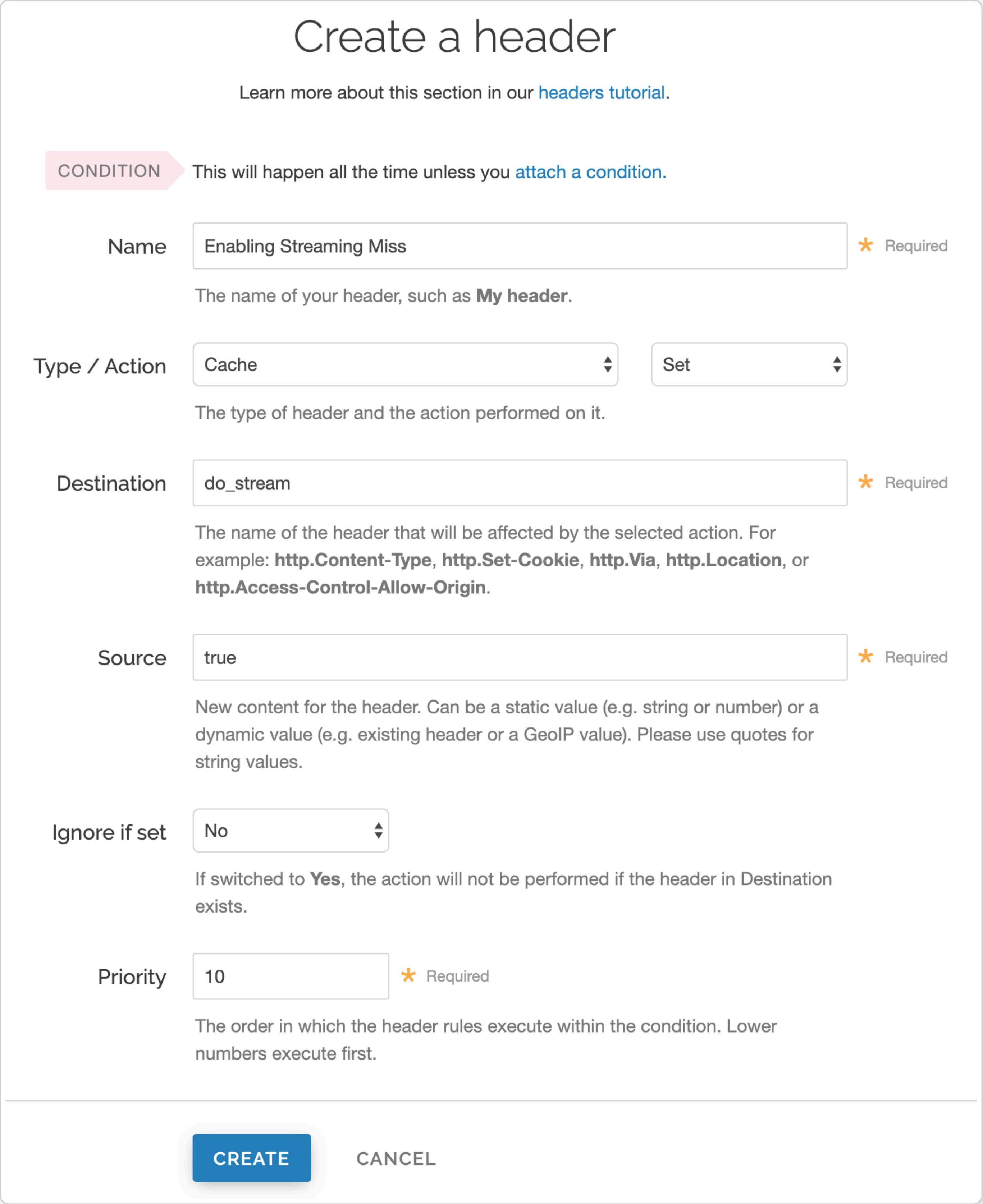
```
set beresp.do_stream = true;
```

6. Fill out the **Create a VCL snippet** fields as follows:
    - In the **Name** field enter an appropriate name (e.g., `Enabling Streaming Miss`).
    - From the **Type (placement of the snippets)** controls, select **within subroutine**.
    - From the **Select subroutine** menu, select **fetch (**`vcl_fetch`**)**.
    - In the **VCL** field, add `set beresp.do_stream = true;`.
7. Click **Create** to create the snippet.
8. Click **Activate** to deploy your configuration changes.

Enable Streaming Miss by creating a Fastly header

Enable Streaming Miss by setting `beresp.do_stream` to `true` in `vcl_fetch` with a Fastly header using these steps.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Enabling Streaming Miss (Required)

The name of your header, such as **My header**.

Type / Action: Cache / Set

The type of header and the action performed on it.

Destination: do_stream (Required)

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: true (Required)

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 (Required)

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the Create a header fields as follows:
    - In the **Name** field, enter the name of your header rule (for example, `Enabling Streaming Miss`).
    - From the **Type** menu, select **Cache**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `do_stream`.

  - In the **Source** field, enter `true`.
  - From the **Ignore if set** menu, select **No** if you want the header in the **Destination** field modified or select **Yes** if you don't want it modified.
  - In the **Priority** field, enter the order the header rules execute.

7. Click **Create**.
8. Click **Activate** to deploy your configuration changes.

Streaming Miss limitations

There are several limitations to using Streaming Miss.

Streaming Miss is not available to HTTP/1.0 clients

If an HTTP/1.0 request triggers a fetch and the response header from the origin does not contain a Content-Length field, then Streaming Miss will be disabled for the fetch and the fetched object will be subject to the non-streaming-miss object size limit of 2GB. Without the client receiving the Content-Length, the client cannot distinguish the proper end of the download from an abrupt connection breakage anywhere upstream from it.

If an HTTP/1.0 request is received while a Streaming Miss for an object is in progress, the HTTP/1.0 request will wait for the response body to be downloaded before it will receive the response header and the response body, as if the object was being fetched without Streaming Miss.

Cache hits are not affected. An HTTP/1.0 client can receive a large object served from cache, just like an HTTP/1.1 client.

Streaming Miss is not compatible with on-the-fly compression of fetched objects

Streaming Miss can handle large files whether or not they are compressed. On-the-fly compression of objects not already compressed is not compatible with Streaming Miss. Specifically, if VCL sets `beresp.gzip` or `beresp.brotli` to true, Streaming Miss will be disabled.

Streaming Miss is not compatible with ESI (Edge-Side Includes)

Responses processed through ESI, which dynamically inserts content into cached pages, cannot be streamed. Responses included from an ESI template also cannot be streamed. When ESI is enabled for a response or when a response is fetched using `<esi:include>`, then Streaming Miss will be disabled and the fetched object will be subject to the non-streaming-miss object size limit of 2GB.

* * *

Category: Purging

These articles describe how to purge cache.

Authenticating URL purge requests via API

🔗 /en/guides/authenticating-api-purge-requests

Fastly's URL purge feature allows you to purge individual URLs on your website. By default, authentication is not required to purge a URL with the Fastly API, but you can enable API token authentication in the Fastly web interface by adding a header or by using custom VCL.

WARNING

We recommend that all customers enable authentication for URL purge requests.

NOTE

All purge requests other than URL purges require authentication by default, as indicated in the API documentation.

Limitations and considerations

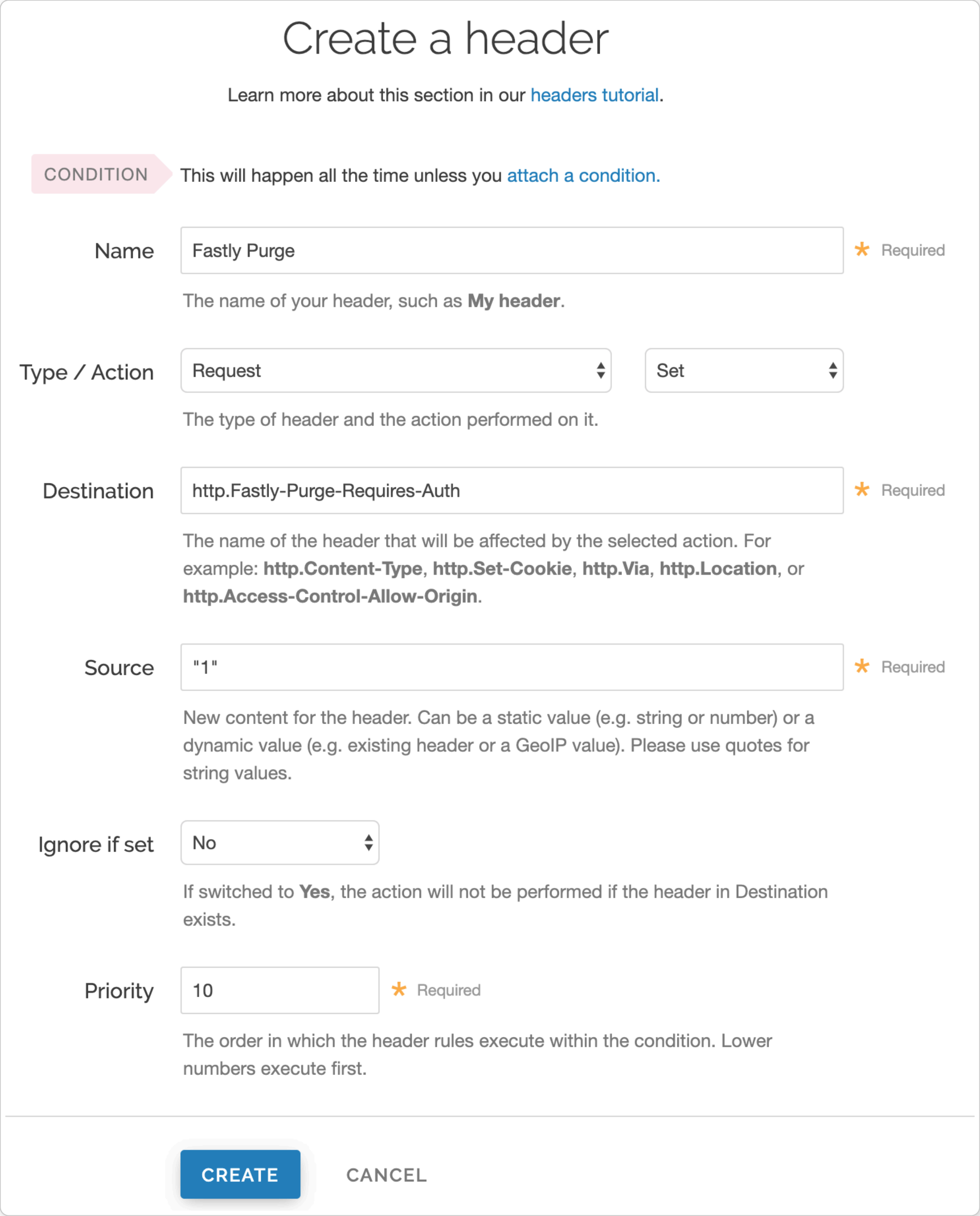
The solution outlined in this guide must be implemented on every service that requires authentication of URL purge requests. To enable purge authentication at the account level, contact support.

Enabling authentication in the Fastly web interface

You can enable API token authentication for URL purge requests by adding a header and optionally attaching a condition in the Fastly web interface.

Adding the header

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Fastly Purge * Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Fastly-Purge-Requires-Auth * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "1" * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter the name of your header rule (for example, `Fastly Purge`).
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.

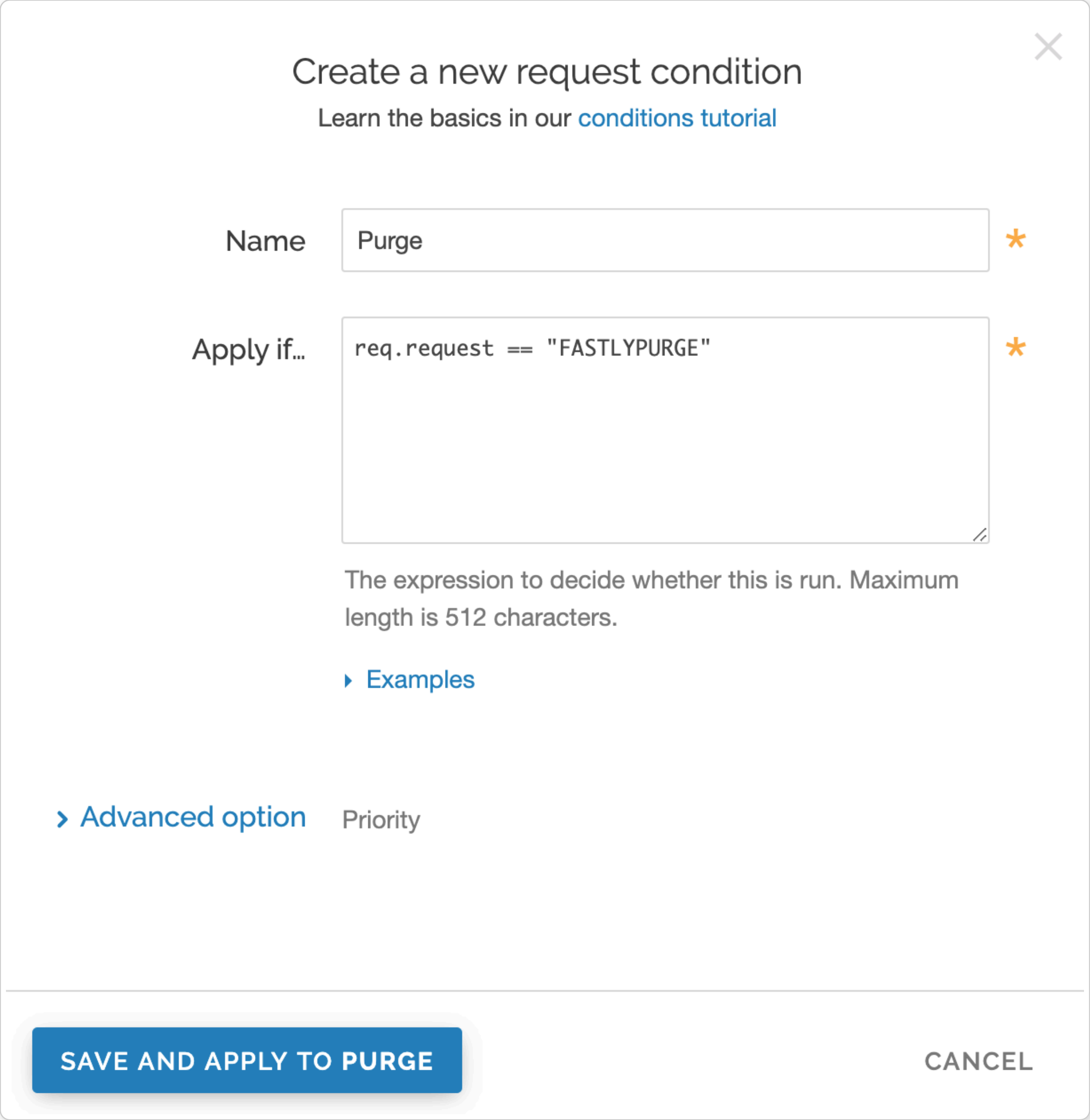
  - In the **Destination** field, enter `http.Fastly-Purge-Requires-Auth`.
  - In the **Source** field, enter `"1"`.
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**.

Attaching a condition

Attaching the following condition is optional. Without the condition, the header you just created will be added to all requests. With the condition, the header will be added to purge requests only.

1. On the Content page, click **Attach a condition** to the right of your new header.


Create a new request condition
Learn the basics in our conditions tutorial
Name
Purge
Apply if...
req.request == "FASTLYPURGE"
The expression to decide whether this is run. Maximum length is 512 characters.
Examples
Advanced option
Priority
SAVE AND APPLY TO PURGE
CANCEL

2. Fill out the **Create a new request condition** fields as follows:
    - In the **Name** field, enter a descriptive name for the new condition (for example, `Purge`).
    - In the **Apply if** field, enter `req.request == "FASTLYPURGE"`.
3. Click **Save and apply to**.
4. Click **Activate** to deploy your configuration changes.

Enabling authentication with VCL Snippets

You can also enable API token authentication for URL purge requests using VCL Snippets. Refer to the developer documentation for details on setting the `Fastly-Purge-Requires-Auth` header.

Purging URLs with an API token

After you've enabled API token authentication for URL purge requests, you'll need to provide your API token in the URL purge API request:

```
$ curl -X PURGE -H Fastly-Key:FASTLY_API_TOKEN https://www.example.com/
```

which would return this response:

```
{"status": "ok", "id": "1234567890"}
```

WARNING

If your website is not configured to use HTTPS, do not use the Fastly API to purge URLs. Doing so could expose your API token since the data in transit will not be encrypted.

* * *

Purging a URL via the web interface

/en/guides/purging-a-url

Fastly provides several levels of cache purging. You can use the *Purge URL* option to purge a single URL via the web interface.

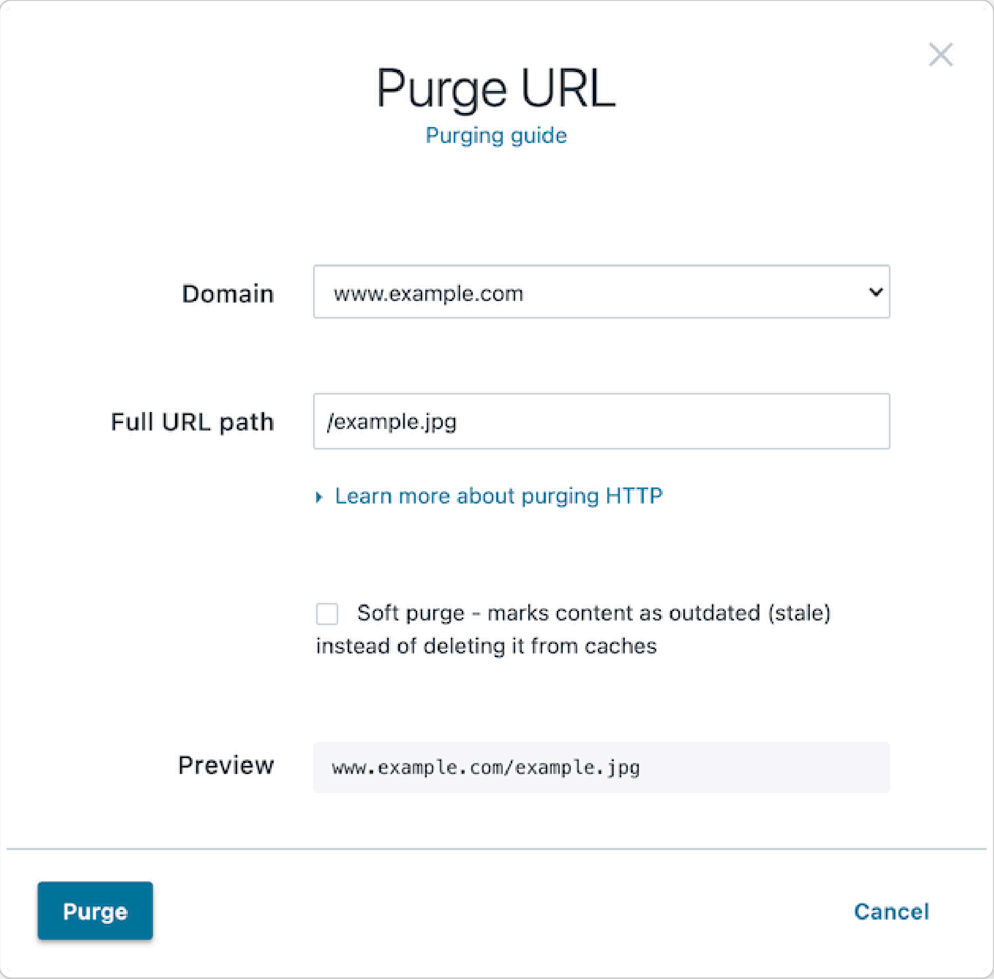
Before you begin

If you're new to the concept of purging, we recommend familiarizing yourself with our purging basics guide.

Purging a URL

To purge a single URL, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Purge** menu, select **Purge URL**.



Domain: www.example.com
Full URL path: /example.jpg
☐ Soft purge - marks content as outdated (stale) instead of deleting it from caches
Preview: www.example.com/example.jpg

4. From the **Domain** menu, select the domain on which your content resides. If the domain you select is a wildcard domain (e.g., `*.example.com`) the Subdomain field will appear.
5. If the **Subdomain** field appears, enter the subdomain to purge for the wildcard domain you've selected (e.g., `www`).
6. In the **Full URL path** field, enter the path to the content you'll be purging (e.g., `/example.jpg`). The Preview field displays the URL that will be purged.
7. *(Optional)* Select the Soft purge checkbox to mark your content as outdated instead of deleting it from cache.
8. Click **Purge**.

What's next

Explore the other purging methods available with Fastly such as purging with surrogate keys and purging all content.

* * *

Purging all content via the web interface

/en/guides/purging-all-content

Fastly provides several levels of cache purging. You can use the *Purge all* option to purge all content under a service.

> ⚠ WARNING
>
> Exercise caution when purging all content from cache, especially if you're seeing an increase in 503 errors. Purging all content overrides other caching-related settings that allow you to serve stale content and forces Fastly to retrieve that content again from your origin servers before it can be re-cached in Fastly POPs. This means that purging all content will temporarily cause your cache hit ratio to drop dramatically and cause an associated spike of traffic to your origin servers. Be certain your origins can handle that temporary spike in traffic until the cache repopulates.

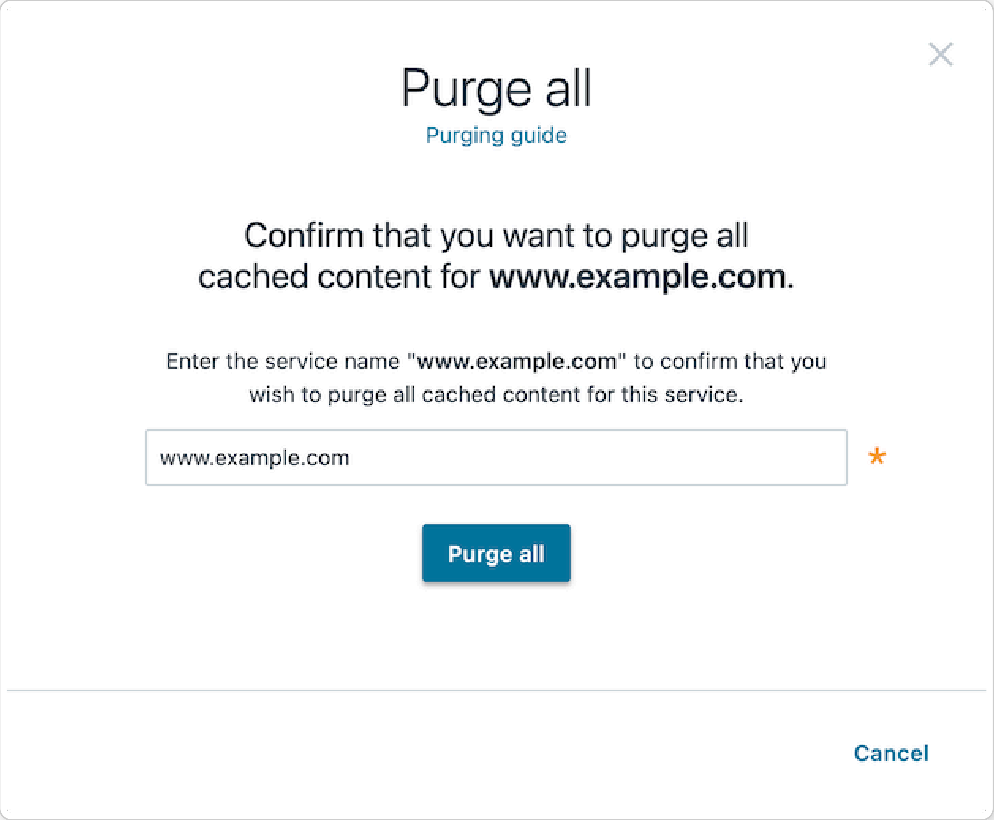
Before you begin

If you're new to the concept of purging, we recommend familiarizing yourself with our purging basics guide.

Purging all content

To instantly purge all content under your service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Purge** menu, select **Purge all**. The Purge all window appears and displays the service name you'll be purging.



Purge all

Confirm that you want to purge all cached content for **www.example.com**.

Enter the service name "**www.example.com**" to confirm that you wish to purge all cached content for this service.

www.example.com

4. In the field of the **Purge all** window, confirm you want to purge all cached content on a service by entering the exact name of the service that appears (e.g., `An Example Service`).

5. Click **Purge all**.

What's next

Explore the other purging methods available with Fastly such as purging with surrogate keys and purging a URL.

* * *

Purging with surrogate keys

/en/guides/purging-with-surrogate-keys

Fastly provides several levels of cache purging and choosing the right purging method is essential to keeping your website fast. While Fastly's purge all is a speedy way to invalidate your cache, it may temporarily increase your website's load time while the cache rebuilds. If you find yourself purging all cache on more than a weekly basis, consider using surrogate keys for more targeted purging.

Surrogate keys are unique identifiers that you assign to portions of content that share common traits, making them easier to process as groups in some way. Using the `Surrogate-Key` header, you can tag an object, such as an image or a blog post, with one or more keys. When you need to explicitly remove content from the Fastly edge cache, rather than allowing it to expire or to be evicted, you can use surrogate keys to purge groups of content selectively instead of purging all content.

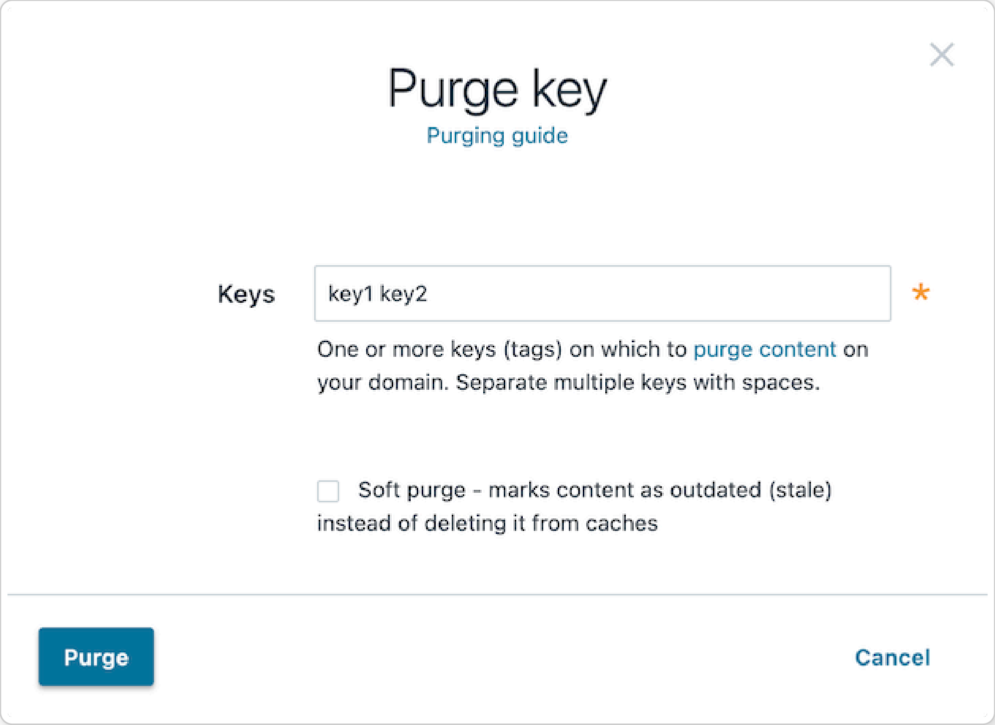
Before you begin

This guide assumes you're already familiar with how to work with the `Surrogate-Key` header to tag related pieces of cacheable content. If you're new to the concept of purging, however, we recommend familiarizing yourself with our purging basics guide.

Purging objects with surrogate keys

You can use the Fastly web interface to manually purge objects via key:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Purge** menu, select **Purge key**.



Purge key

Purging guide

Keys: key1 key2 *

One or more keys (tags) on which to purge content on your domain. Separate multiple keys with spaces.

☐ Soft purge - marks content as outdated (stale) instead of deleting it from caches

4. In the **Keys** field, enter one or more surrogate keys. Use spaces to separate multiple keys.
5. *(Optional)* Select the Soft purge checkbox to mark your content as outdated instead of deleting it from cache.
6. Click **Purge**.

You can also use our Purge API to purge objects via key.

What's next

Explore the other purging methods available with Fastly such as purging a URL and purging all content.

* * *

Soft purges

/en/guides/soft-purges

Fastly provides a Soft Purge feature that allows you to mark content as outdated (*stale*). Stale objects remain available to use in some circumstances while Fastly fetches a new version from origin, unlike objects invalidated by a *hard* purge which are made immediately unusable. You can purge by URL or by surrogate key using Soft Purge.

Before using Soft Purge, we recommend you implement one of the following methods to verify a stale object should temporarily remain in cache during revalidation:

- Set up `ETag` or `Last-Modified` headers for relevant content on your origin servers.
- Configure `stale_while_revalidate` to serve stale content and fetch the newest version of the object from origin in the background. If you choose this revalidation method, you must also configure `stale_if_error` at

the same time.

To enable Soft Purge, add a `Fastly-Soft-Purge` request header (such as `Fastly-Soft-Purge: 1`) to any single URL or key-based purge.

To purge the URL `www.example.com` with Soft Purge, you would issue the following command:

```
$ curl -X PURGE -H "Fastly-Soft-Purge:1" http://www.example.com
```

To purge a surrogate key with Soft Purge, you would issue the following command:

```
$ curl -X POST -H 'Fastly-Soft-Purge:1' -H 'Fastly-Key: YOUR_FASTLY_TOKEN' -H 'Accept: a  li
```

TIP

Purge all requests cannot be done as a soft purge and will always immediately invalidate all cached content associated with the service. To do a soft purge all, consider applying a constant surrogate key tag (e.g., `all`) to all objects.

* * *

Working with surrogate keys

/en/guides/working-with-surrogate-keys

Surrogate keys are unique identifiers that you assign to groups of content for processing. While there are many use cases for surrogate keys, one of the primary way for using them with Fastly is to make purging more efficient.

Surrogate keys allow you to selectively purge related content. Using the `Surrogate-Key` header, you can "tag" content with a key term, a string of any characters you want. When you want to purge content associated with that key, you then issue a key purge request and all of the objects associated with that key will be purged. This process makes it easier to cache and purge content that changes rapidly and unpredictably without having to purge your entire cache.

Before you begin

This guide assumes you're already familiar with the way content delivery networks (CDNs) work in general, and the way Fastly's CDN works in particular. We also recommend reviewing our guide on adding headers on HTTP requests and responses.

Understanding the `Surrogate-Key` header

To understand how surrogate keys work, especially in the context of content purging, you need to understand what a Surrogate-Key header is and how it behaves.

About the `Surrogate-Key` header

Any time your origin responds to a request for content, the response it sends will include bits of code called *headers* in front of the actual body of the response. Those headers are used to give additional detail and provide context for the body of the response itself. HTTP headers sent as part of a response typically look something like this:

```
HTTP/1.1 200 OK
Content-Type: text/html
Connection: keep-alive
...
```

You can control that additional information and therefore control how content is served by adding to or modifying the headers your server responds with. The `Surrogate-Key` header is one of them. It's what categorizes or "tags" the content as part of a specific group. The tag you choose is called a *key* and it describes the common trait each element of a content group shares.

For example, suppose you have a website where you publish taco recipes. You know that entire categories of recipes will need to be purged, so you thoughtfully include these recipe categories as keys in the `Surrogate-Key` header. The response your server sends might look more like this:

```
HTTP/1.1 200 OK
Surrogate-Key: veggie seasonal central-mexico
Content-Type: text/html
...
```

The addition of the Surrogate-Key header there tells you that the content is tagged with specific identifiers that categorize it for future use. This response contains three surrogate keys: `veggie`, `seasonal`, and `central-mexico`. When Fastly receives a response like this, we use the surrogate keys to create a mapping from each key to the cached content, then we strip out the `Surrogate-Key` header so it's not included in the response to your readers.

TIP

You can't use duplicate surrogate keys. For example, if you tried to use `foo`, `bar`, and `foo`, the two `foo`s would be collapsed into a single instance of the term.

Creating relationships between keys and objects

One of the major advantages of surrogate keys is that they allow for a many-to-many relationship between keys and objects. An origin server's response can associate multiple keys with the object and the same key can be provided in different responses. Take a look at these two requests and responses:

```
GET /blog/healthy-taco-recipes HTTP/1.1
Host: www.tacolabs.com

HTTP/1.1 200 OK Content-Type: text/html
Content-Length: 1234
Surrogate-Key: mainpage low-carb
```

```
GET /recipes/low-carb-cheese-taco-shell HTTP/1.1
Host: www.tacolabs.com
```

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 2345
Surrogate-Key: low-carb cheese
```

In this example, there are two objects (`/blog/healthy-taco-recipes` and `/recipes/low-carb-cheese-taco-shell`) with three keys (`mainpage`, `low-carb`, and `cheese`). Two of the keys (`mainpage` and `cheese`) are associated with a single object and a third key (`low-carb`) is associated with both objects.

By using the `Surrogate-Key` header to associate keys with one or more objects, you can precisely control which objects are removed from cache during a purge. Consider the example presented above. Purging the `mainpage` key would remove only the `/blog/healthy-taco-recipes` object from the cache. On the other hand, purging the `low-carb` key would remove both the `/blog/healthy-taco-recipes` and `/recipes/low-carb-cheese-taco-shell` objects from the cache.

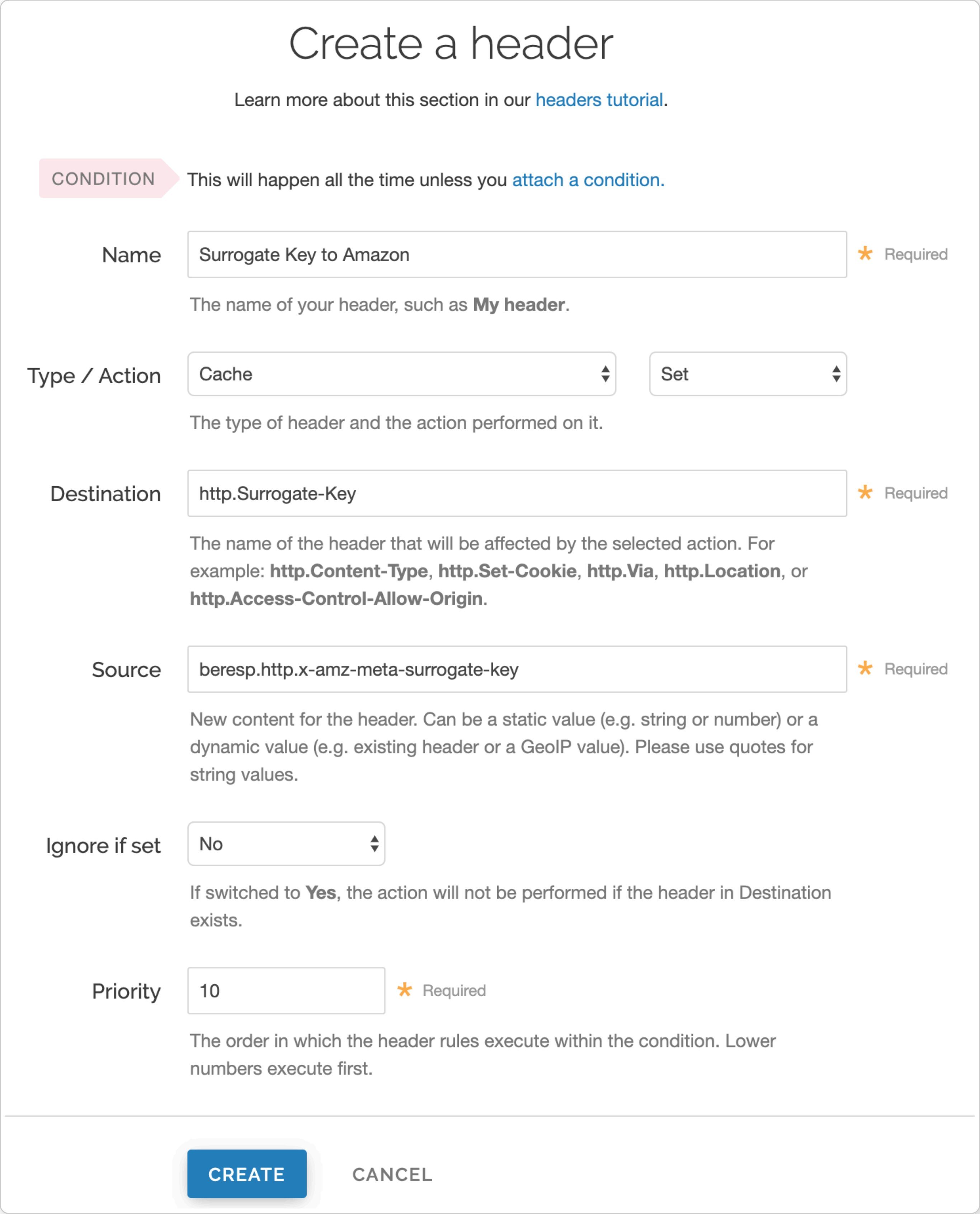
Wherever there's a relation between two different pieces of content, there might be a good reason to keep them categorized by using a surrogate key.

Generating and setting surrogate keys

There are two ways to set the `Surrogate-Key` header: by adding the header in the Fastly web interface, or by generating the keys with your own application. You can make your surrogate key associations on your own application server and include them in the HTTP response that you send to Fastly. This is the more useful method because you can assign exactly the keys that you want on every response.

However, if you want to set the `Surrogate-Key` header in the Fastly web interface you can do so following the steps below.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Surrogate Key to Amazon * Required

The name of your header, such as **My header**.

Type / Action: Cache | Set

The type of header and the action performed on it.

Destination: http.Surrogate-Key * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: beresp.http.x-amz-meta-surrogate-key * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter a human-readable name for the header. This name is displayed in the Fastly web interface.
    - From the **Type** menu, select **Cache**, and from the **Action** menu, select **Set**.

   - In the **Destination** field, enter `http.Surrogate-Key`.
   - In the **Source** field, enter where the content for the header comes from. For example, if you're using Amazon S3 as an origin server, you could set the field to `beresp.http.x-amz-meta-surrogate-key` to specify `x-amz-meta-surrogate-key` as the response header received from the backend.
   - From the **Ignore if set** menu, select **No**.
   - In the **Priority** field, enter `10`.

7. Click the **Create** button to create your header.
8. Click **Activate** to deploy your configuration changes.

Troubleshooting

You can check the surrogate keys for a URL by using the `Fastly-Debug: 1` header. See the instructions on using a Fastly-Debug header with curl for more information.

Limitations

The surrogate keys sent by your origin server can be as simple or complex as you need, subject to format limitations. Surrogate keys must be formatted as a single string without spaces. Spaces separate keys.

Surrogate keys are subject to size limitations. Individual surrogate keys may not exceed 1,024 bytes in length and `Surrogate-Key` header values (comprising one or more space-separated keys) may not exceed 16,384 bytes in length. If either of the key or key header value limits are reached while parsing a `Surrogate-Key` header, the key currently being parsed and all keys following it within the same header will be ignored.

What's next

Now that you understand how the `Surrogate-Key` header works and how to set it, learn how to purge objects with surrogate keys.

---

* * *

Category: Requests

These articles describe configuration settings and changes you can make to requests when working with Fastly services.

Checking multiple backends for a single request

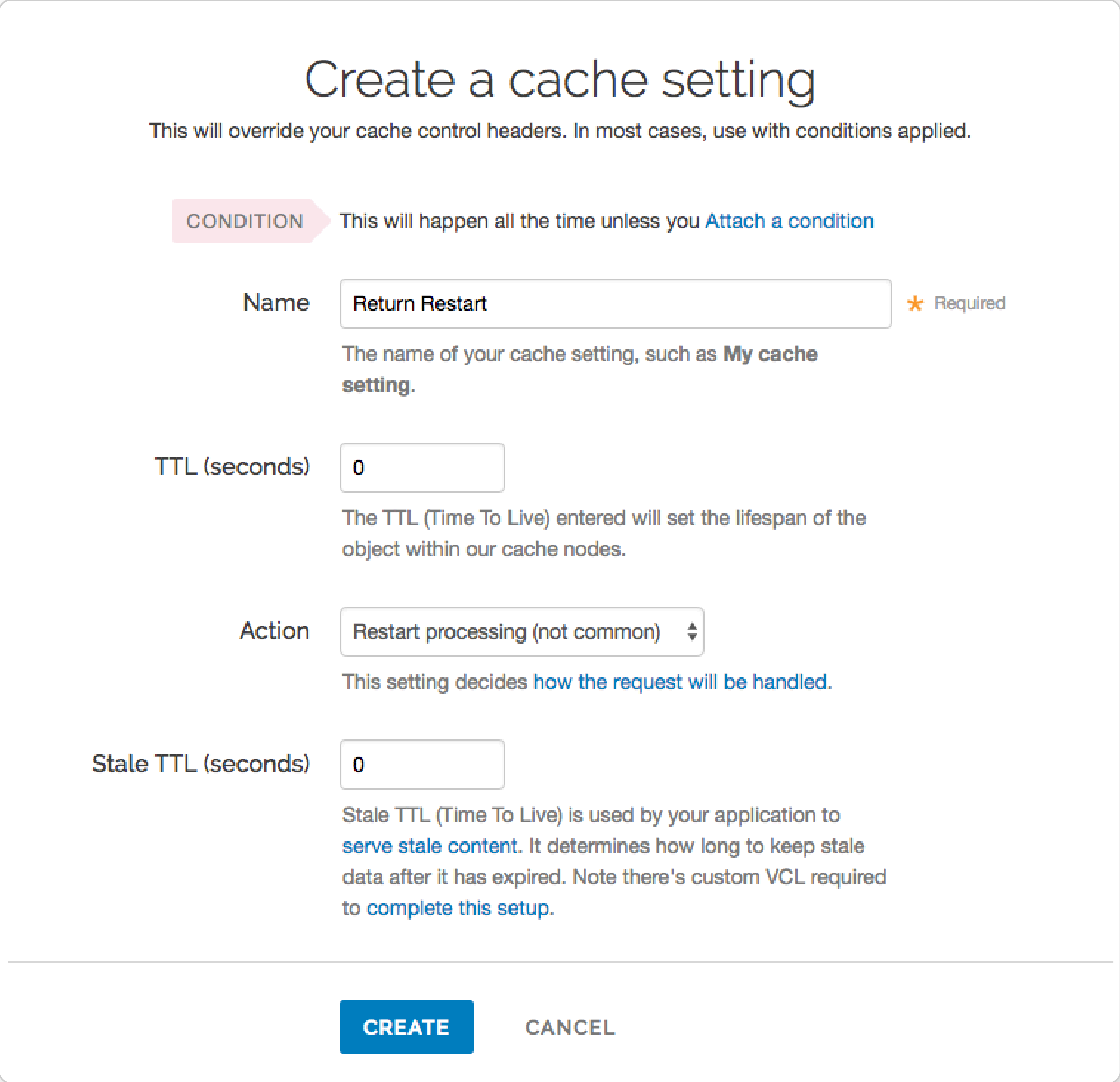
/en/guides/checking-multiple-backends-for-a-single-request

Using a restart is a good option to check multiple backends for a single request. This can be created using a cache setting rule and request headers.

Create a new cache setting rule

Follow these steps to create a cache restart within `vcl_fetch`.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. Click **Create cache setting**.



Create a cache setting

This will override your cache control headers. In most cases, use with conditions applied.

CONDITION This will happen all the time unless you Attach a condition

Name: Return Restart ★ Required

The name of your cache setting, such as **My cache setting**.

TTL (seconds): 0

The TTL (Time To Live) entered will set the lifespan of the object within our cache nodes.

Action: Restart processing (not common)

This setting decides how the request will be handled.

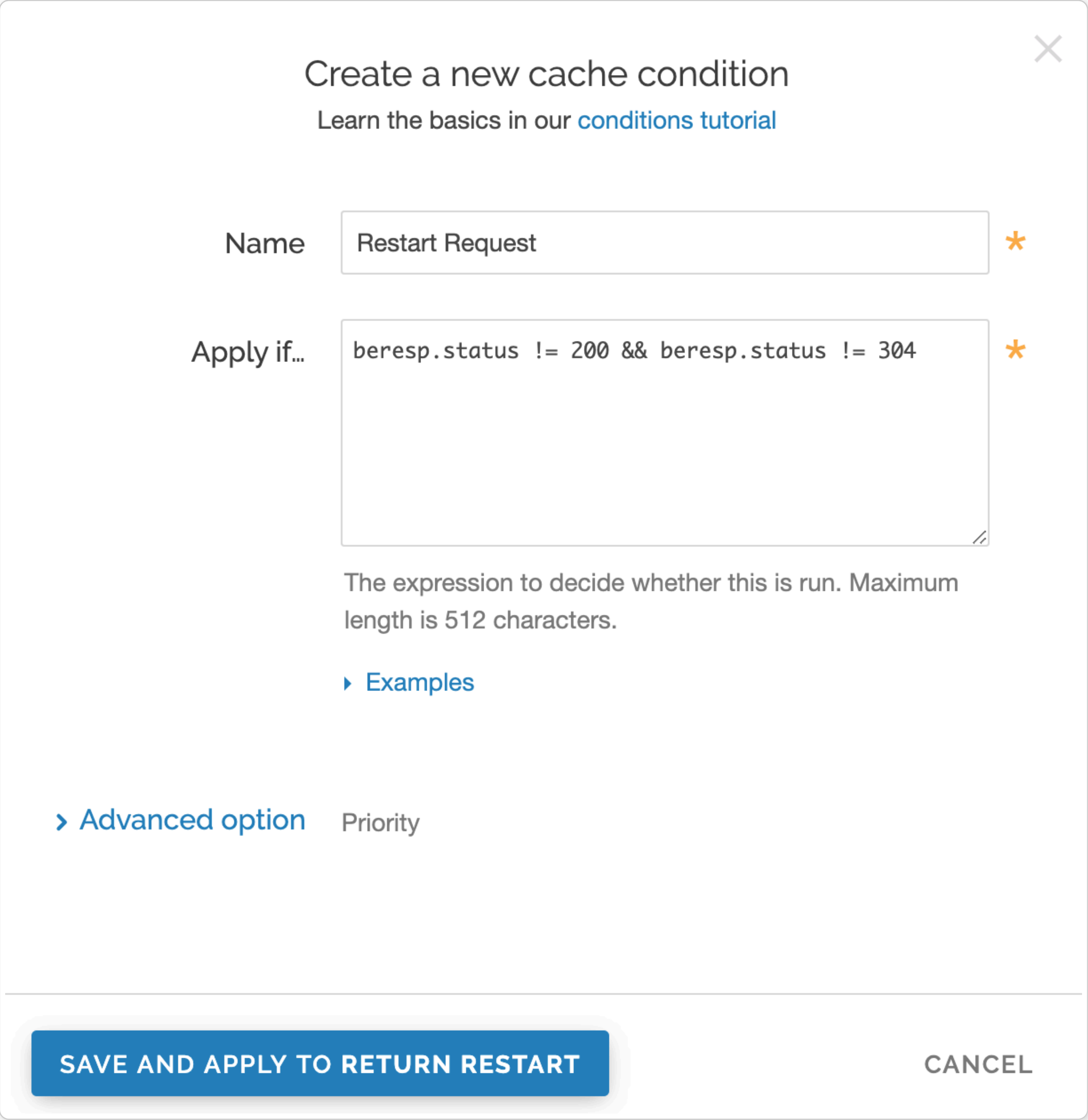
Stale TTL (seconds): 0

Stale TTL (Time To Live) is used by your application to serve stale content. It determines how long to keep stale data after it has expired. Note there's custom VCL required to complete this setup.

6. Fill out the **Create a cache setting** fields as follows:
   - In the **Name** field, enter `Return Restart` (or any meaningful, preferred name).
   - In the **TTL (seconds)** field, enter `0`.

  - From the **Action** menu, select **Restart processing**.
  - In the **Stale TTL (seconds)** field, enter `0`.

7. Click **Create**.
8. On the **Settings** page, click **Attach a condition** next to the cache setting you just created.



Create a new cache condition

Learn the basics in our conditions tutorial

Name: Restart Request *

Apply if...: `beresp.status != 200 && beresp.status != 304` *

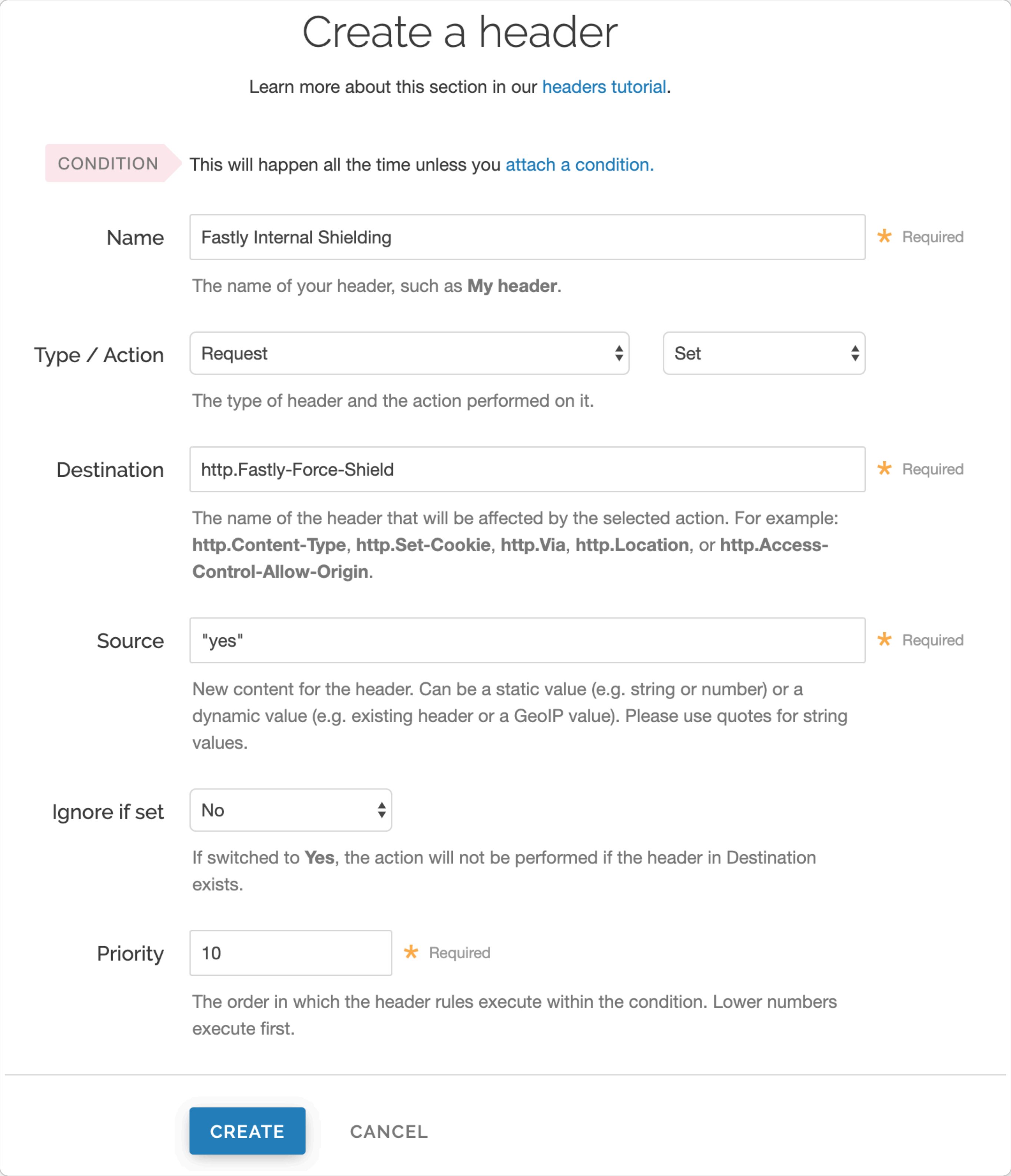
The expression to decide whether this is run. Maximum length is 512 characters.

9. Fill out the **Create a new cache condition** fields as follows:
   - In the **Name** field, enter `Restart Request` (or any meaningful, preferred name).
   - In the **Apply if** field, enter `beresp.status != 200 && beresp.status != 304`.
10. Click **Save and apply** to create the condition.

Create new request headers

Follow these steps to create a request header within `vcl_recv`.

1. Click **Content**.
2. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Fastly Internal Shielding ★ Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Fastly-Force-Shield ★ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "yes" ★ Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

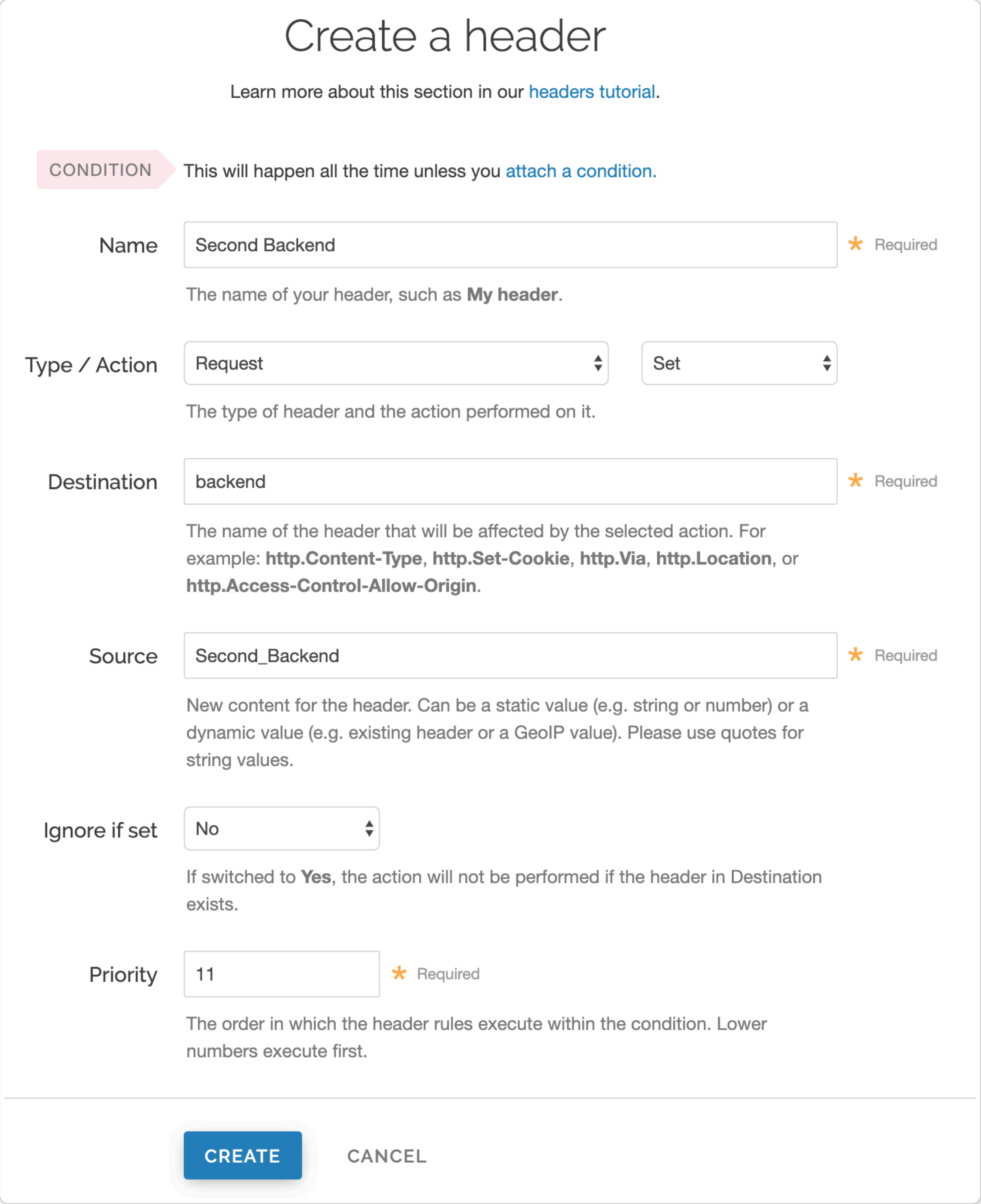
Priority: 10 ★ Required

The order in which the header rules execute within the condition. Lower numbers execute first.

3. Fill out the **Create a new header** fields as follows:

- In the **Name** field, enter `Fastly Internal Shielding` (or any meaningful, preferred name).
- From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
- In the **Destination** field, enter `http.Fastly-Force-Shield`.
- In the **Source** field, enter `"yes"`.
- From the **Ignore if set** menu, select **No**.
- In the **Priority** field, enter `10`.

4. Click **Create**.
5. Click **Create header** to create another header to switch to the next backend.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Second Backend

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: backend

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: Second_Backend

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 11

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Second Backend` (or any meaningful, preferred name).
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `backend`.

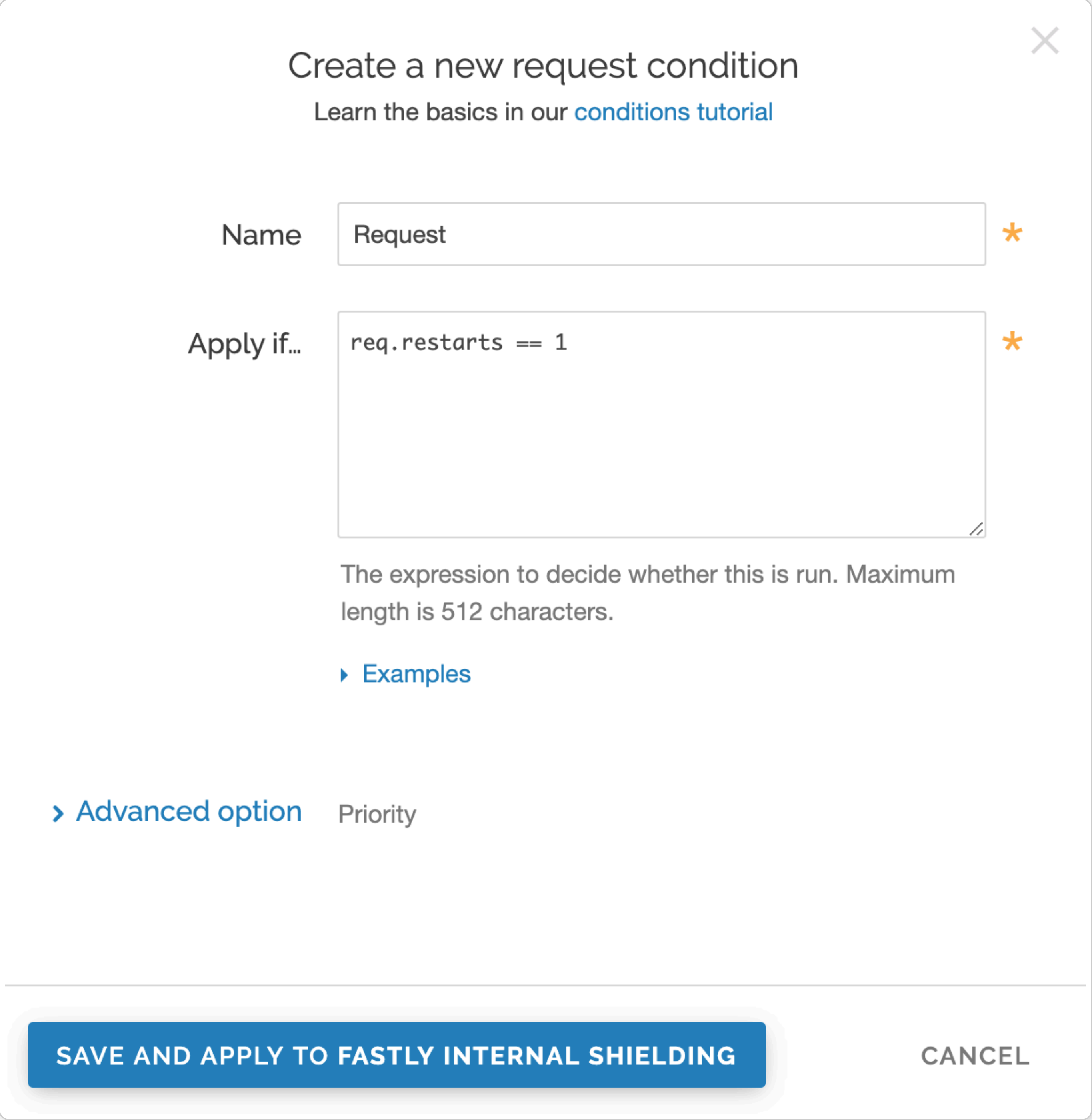
  - In the **Source** field, enter `Second_Backend` (this should match the name of your other backend).
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `11`.

7. Click **Create**. The new header appears on the Content page.

Create new header conditions

Follow these steps to create conditions for the headers.

1. On the **Content** page, click **Attach a condition** next to one of the headers you just created.


Create a new request condition
Learn the basics in our conditions tutorial

Name: Request

Apply if...:

```
req.restarts == 1
```

The expression to decide whether this is run. Maximum length is 512 characters.
Examples
Advanced option
Priority
SAVE AND APPLY TO FASTLY INTERNAL SHIELDING
CANCEL

2. Fill out the **Create a new request condition** fields as follows:

   - In the **Name** field, enter `Req.request` (or any meaningful, preferred name).
   - In the **Apply if** field, enter `req.restarts == 1`.
3. Click **Save and apply to**. The condition appears on the Content page.
4. Repeat steps 1-3 for the other header.
5. Click **Activate** to deploy your configuration changes.

* * *

Conditionally changing a URL

/en/guides/conditionally-changing-a-url

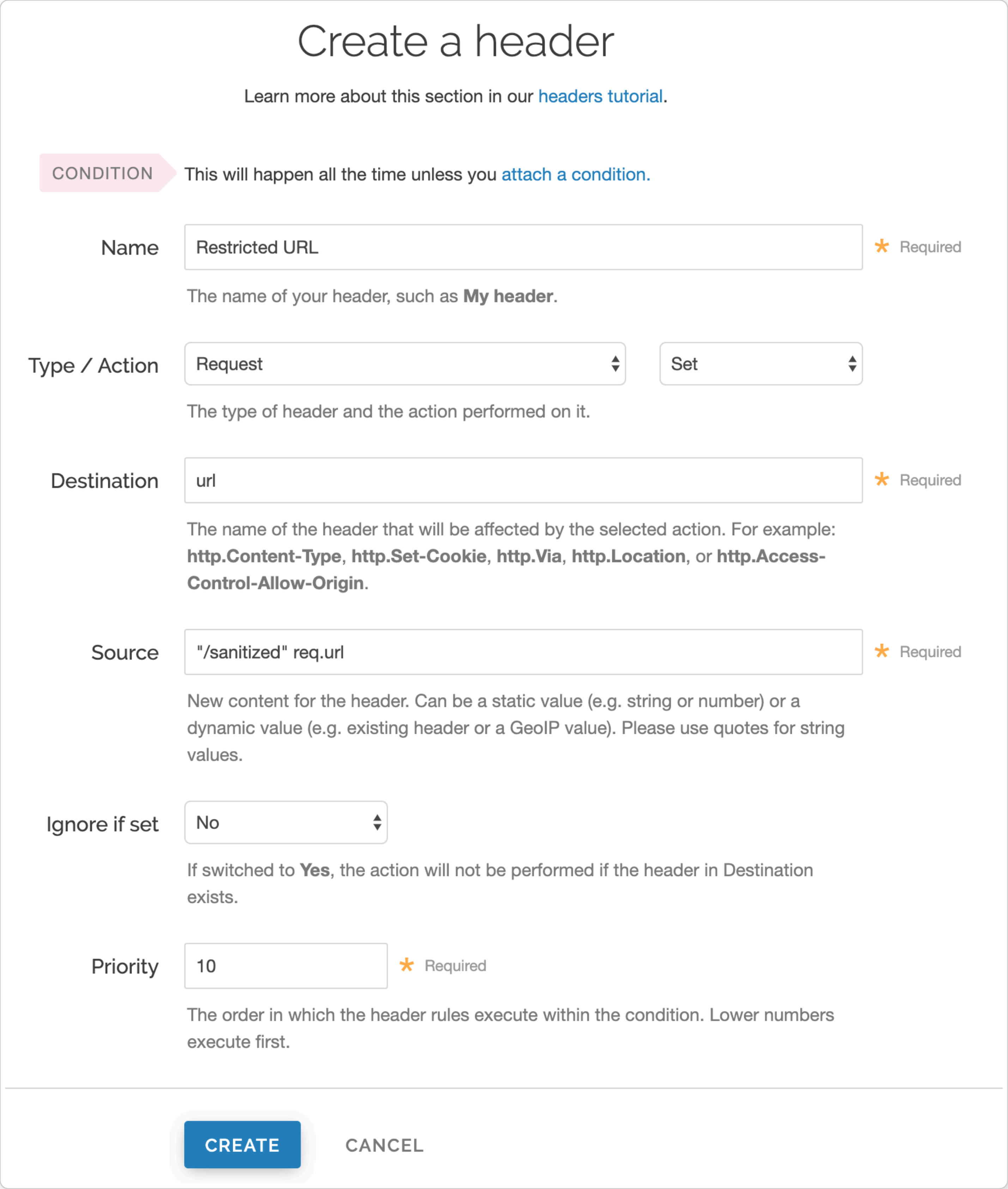
To conditionally change a URL based on the domain, include VCL that looks something like this:

```
if (req.http.host ~ "^restricted") {
  set req.url = "/sanitized" req.url;
}
```

If you have shielding enabled, however, add the following code instead to avoid rewriting the URL twice:

```
if (req.http.host ~ "^restricted" && req.url !~ "^/sanitized") {
  set req.url = "/sanitized" req.url;
}
```

In Fastly's web interface, this VCL would be the equivalent of creating a new Header:



Create a header
Learn more about this section in our headers tutorial.
CONDITION This will happen all the time unless you attach a condition.

Name: Restricted URL

* Required
The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: url

* Required
The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "/sanitized" req.url

* Required
New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

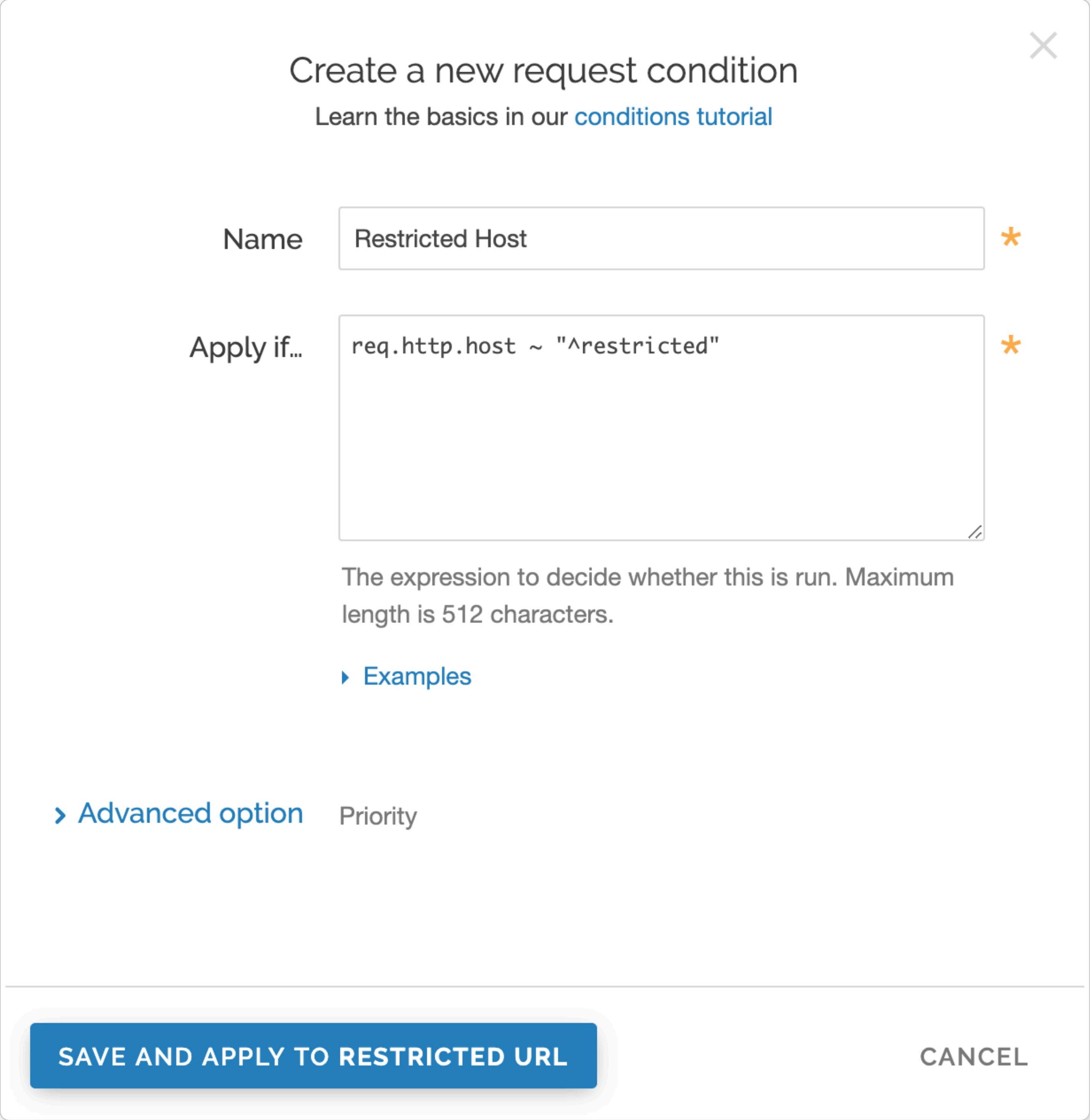
Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

* Required
The order in which the header rules execute within the condition. Lower numbers execute first.
CREATE
CANCEL

and then creating a request condition that restricts connections to that host:



Create a new request condition

Learn the basics in our conditions tutorial

Name: Restricted Host

Apply if...

```
req.http.host ~ "^restricted"
```

The expression to decide whether this is run. Maximum length is 512 characters.

Examples
Advanced option
Priority
SAVE AND APPLY TO RESTRICTED URL
CANCEL

* * *

How request settings are applied

/en/guides/how-request-settings-are-applied

Request settings are applied based on the Action you select in the Create a new request setting page. You can choose any one of the following settings:

- **Do nothing now** - Apply the request setting options, but don't force a lookup or a pass action. The request settings are applied as the system continues through the VCL logic.
- **Lookup (in cache)** - Immediately search the cache for content. If the content isn't found (a MISS), then send the request to the origin.
- **Pass (do not cache)** - Immediately send the request to the origin each time and ignore additional request configurations. See our info on understanding the different PASS action behaviors to learn more.

* * *

Category: Responses

These articles describe configuration settings and changes you can make to your response settings when setting up Fastly services.

Creating and customizing a robots.txt file

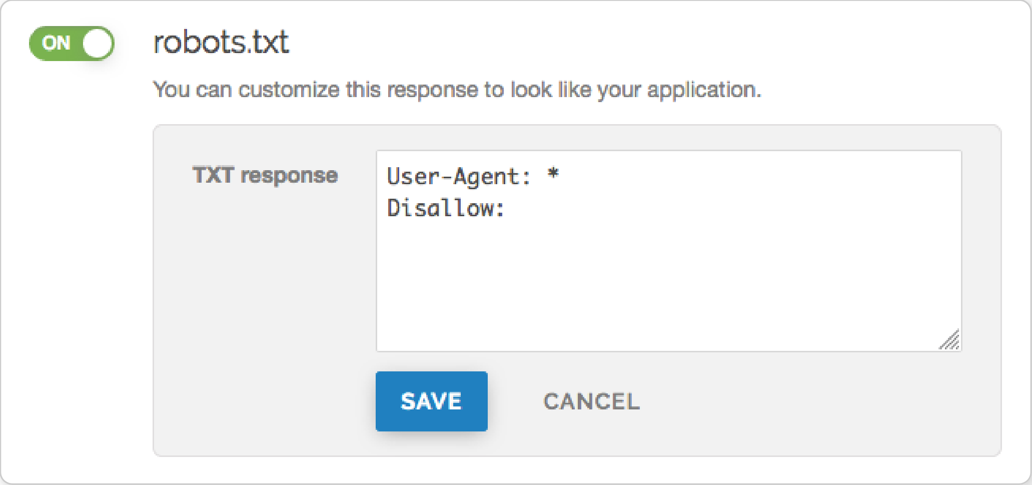
/en/guides/creating-and-customizing-a-robots-file

The robots.txt file tells web robots how to crawl webpages on your website. You can use Fastly's web interface to create and configure a robots.txt file. If you follow the instructions in this guide, Fastly will serve the robots.txt file from cache so the requests won't hit your origin.

Creating a robots.txt file

To create and configure your robots.txt file, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click the **robots.txt** switch to enable the robots.txt response.


ON
robots.txt
You can customize this response to look like your application.
TXT response
SAVE
CANCEL

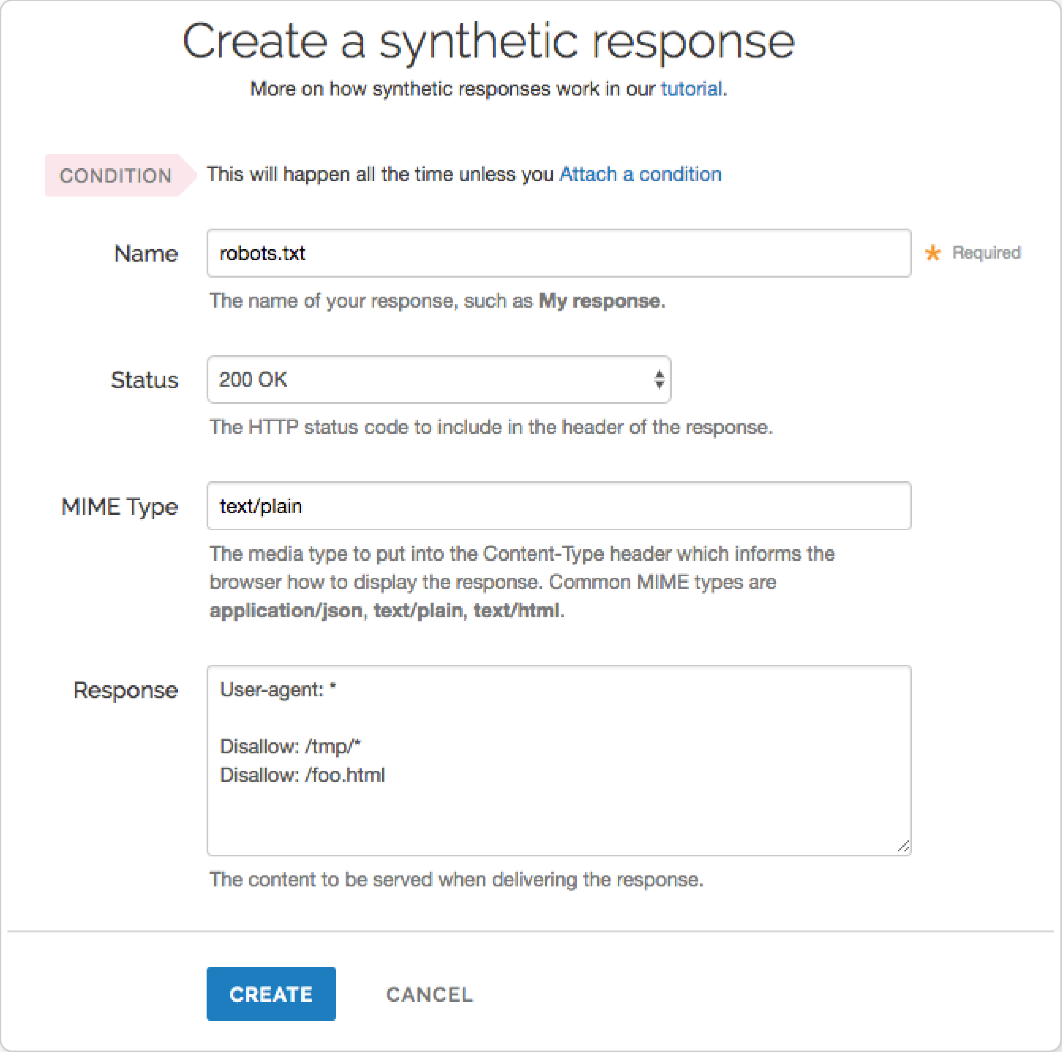
```
User-Agent: *
Disallow:
```

6. In the **TXT Response** field, customize the response for the robots.txt file.
7. Click **Save** to save the response.
8. Click **Activate** to deploy your configuration changes.

Manually creating and customizing a robots.txt file

If you need to customize the robots.txt response, you can follow the steps below to manually create the synthetic response and condition:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Set up advanced response**.



Create a synthetic response

More on how synthetic responses work in our tutorial.

CONDITION This will happen all the time unless you Attach a condition

Name: robots.txt ★ Required

The name of your response, such as **My response**.

Status: 200 OK

The HTTP status code to include in the header of the response.

MIME Type: text/plain

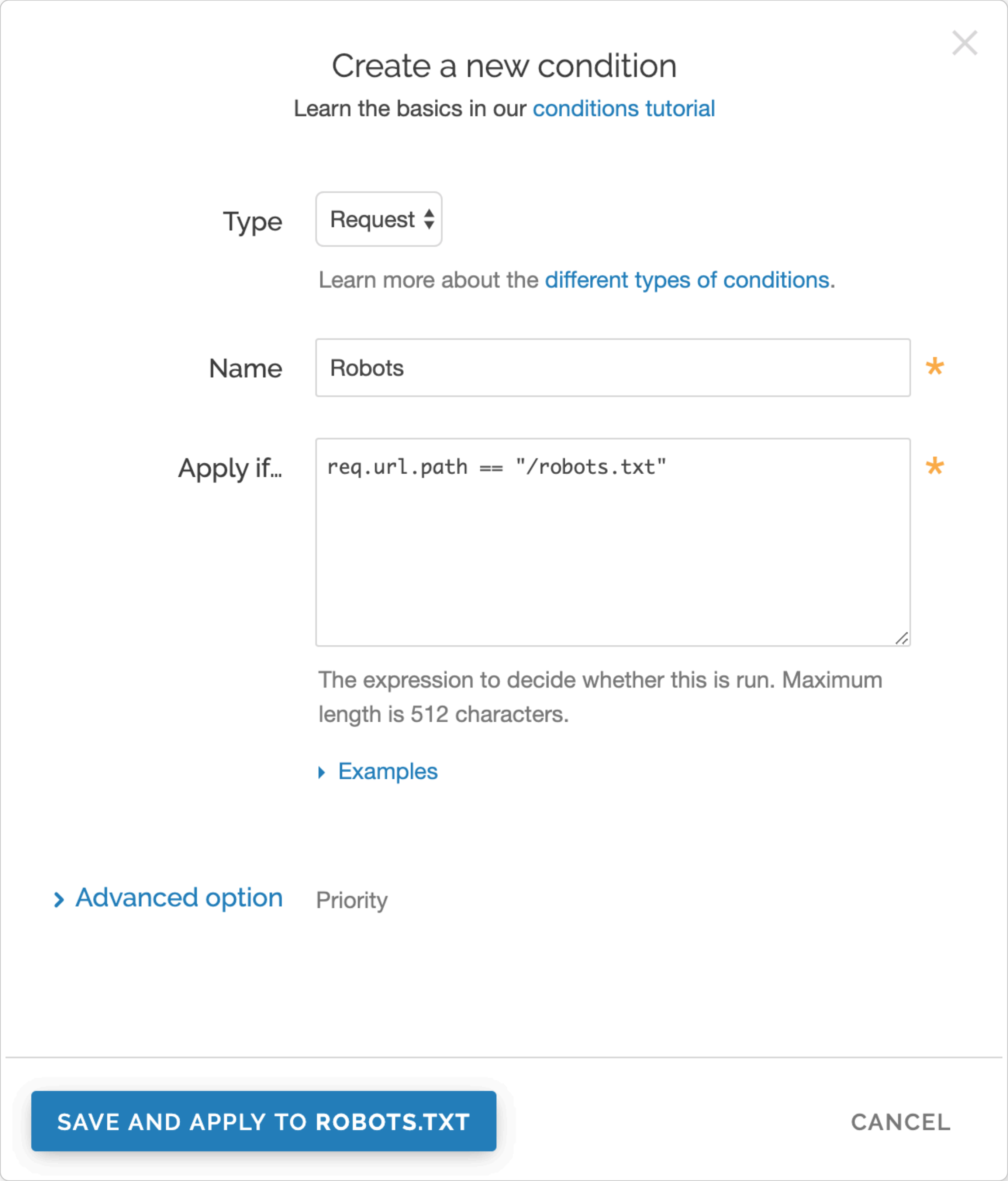
The media type to put into the Content-Type header which informs the browser how to display the response. Common MIME types are **application/json**, **text/plain**, **text/html**.

Response:

```
User-agent: *

Disallow: /tmp/*
Disallow: /foo.html
```

The content to be served when delivering the response.

6. Fill out the **Create a synthetic response** fields as follows:
   - In the **Name** field, enter an appropriate name. For example `robots.txt`.
   - Leave the **Status** menu set at its default `200 OK`.
   - In the **MIME Type** field, enter `text/plain`.
   - In the **Response** field, enter at least one User-agent string and one Disallow string. For instance, the above example tells all user agents (via the `User-agent: *` string) they are not allowed to crawl anything after `/tmp/` directory or the `/foo.html` file (via the `Disallow: /tmp/*` and `Disallow: /foo.html` strings respectively).
7. Click **Create**.
8. Click the **Attach a condition** link to the right of the newly created response.


Create a new condition
Learn the basics in our conditions tutorial
Learn more about the different types of conditions.
The expression to decide whether this is run. Maximum length is 512 characters.
Examples
Advanced option
Priority
SAVE AND APPLY TO ROBOTS.TXT
CANCEL

Type: Request

Name: Robots

Apply if...: `req.url.path == "/robots.txt"`

9. Fill out the **Create a condition** fields as follows:
   - From the **Type** menu, select the desired condition (for example, `Request`).
   - In the **Name** field, enter a meaningful name for your condition (e.g., `Robots`).
   - In the **Apply if** field, enter the logical expression to execute in VCL to determine if the condition resolves as true or false. In this case, the logical expression would be the location of your robots.txt file (e.g., `req.url.path == "/robots.txt"`).

10. Click **Save**.

11. Click **Activate** to deploy your configuration changes.

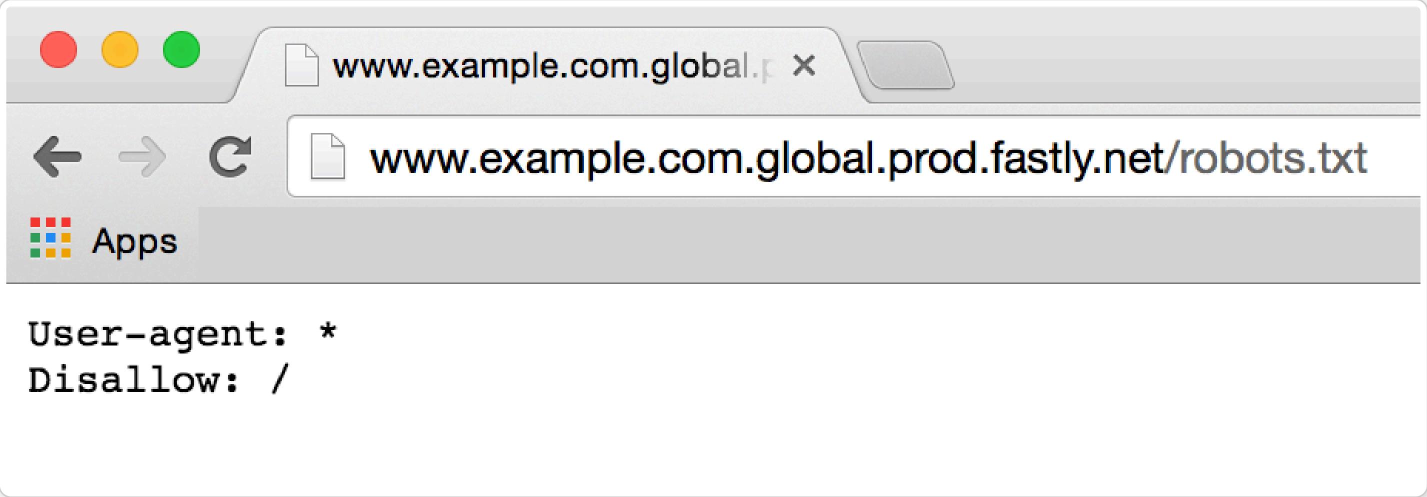
NOTE

For an in-depth explanation of creating custom responses, check out our Responses Tutorial.

Why can't I customize my robots.txt file with global.prod.fastly.net?

Adding the `.global.prod.fastly.net` extension to your domain (for example, `www.example.com.global.prod.fastly.net`) via the browser or in a curl command can be used to test how your production site will perform using Fastly's services.

To prevent Google from accidentally crawling this test URL, we provide an internal robots.txt file that instructs Google's webcrawlers to ignore all pages for all hostnames that end in `.prod.fastly.net`.



```
User-agent: *
Disallow: /
```

This internal robots.txt file cannot be customized via the Fastly web interface until after you have set the CNAME DNS record for your domain to point to `global.prod.fastly.net`.

* * *

Creating error pages with custom responses

/en/guides/creating-error-pages-with-custom-responses

The default error responses served by Fastly can be jarring for your users, especially when using Fastly for consumer applications. To mitigate this, consider configuring your service to present them with a custom page or a synthetic response when Fastly receives an error code from your origin.

Fastly offers two quick configuration options for creating 404 and 503 error pages directly in the web interface, but you can also use the interface to create error pages for other status codes. If you're working with large blocks of content when styling your error pages, consider creating custom responses using VCL snippets instead.

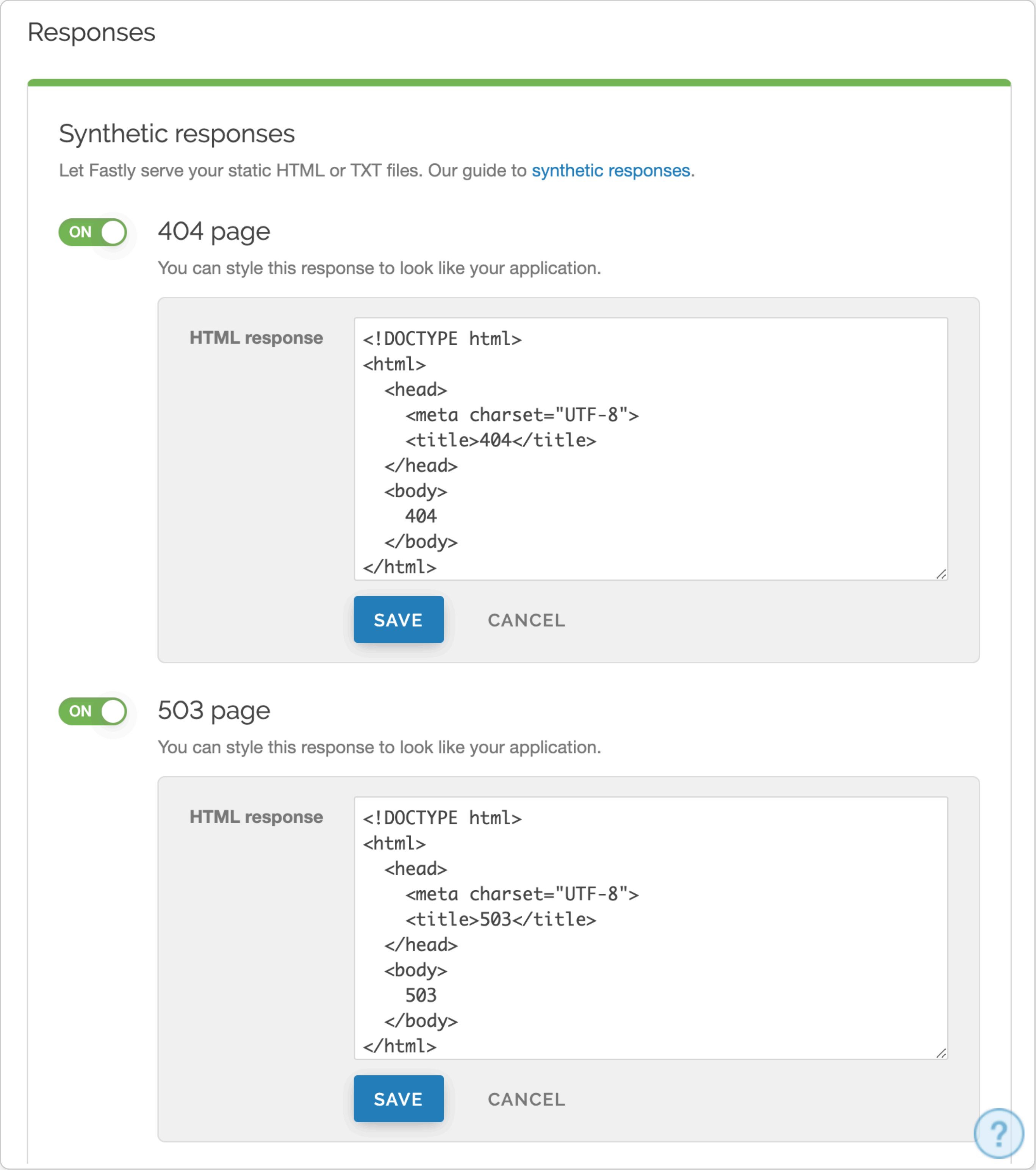
TIP

Instead of an error message, Fastly can optionally serve stale content when there is a problem with your origin server. For more information, check out our guide on serving stale content.

Creating error pages for 404 and 503 errors

To create error pages with custom responses for 404 and 503 errors, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. To create error pages with custom responses for 404 and 503 errors, click the **404 page** and **503 page** switches.


Responses
Synthetic responses
Let Fastly serve your static HTML or TXT files. Our guide to synthetic responses.
ON
404 page
You can style this response to look like your application.
HTML response

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>404</title>
  </head>
  <body>
    404
  </body>
</html>
```

SAVE
CANCEL
ON
503 page
You can style this response to look like your application.
HTML response

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>503</title>
  </head>
  <body>
    503
  </body>
</html>
```

SAVE
CANCEL

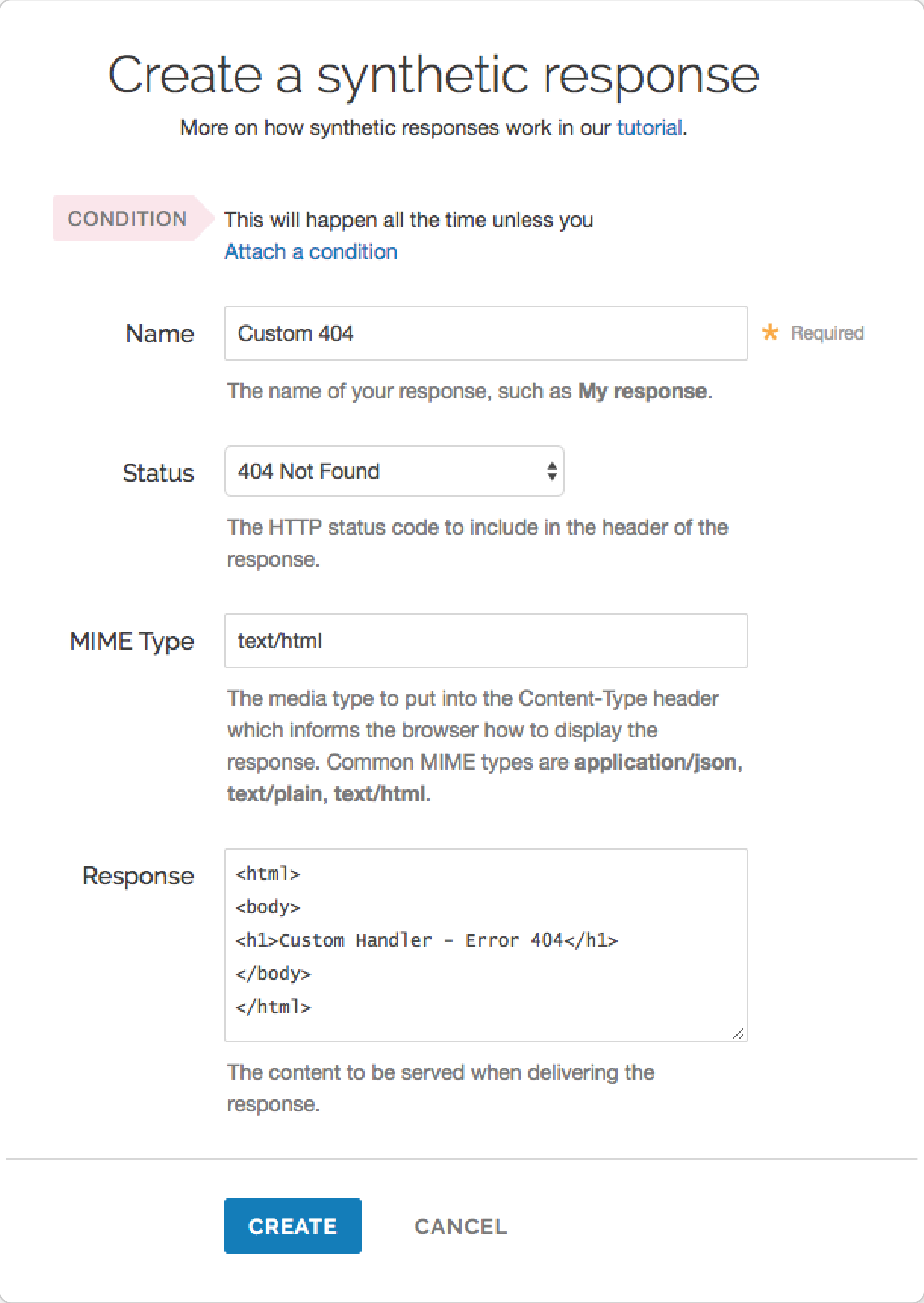
6. In the **HTML response** fields, customize the response for the 404 and 503 error pages.
7. Click **Save** to save the responses.
8. Click **Activate** to deploy your configuration changes.

Creating error pages for other status codes

You can also create error pages for other HTTP status codes. We provide example HTML, but you can use any HTML you see fit. The response object will require that you use a condition in order for a custom error page to be served, otherwise a generic error page will be served.

To create and configure an error page for an HTTP status code other than 404 or 503, follow the steps below to create the custom response and the condition under which it should be applied using the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Set up advanced response**.



Create a synthetic response

More on how synthetic responses work in our tutorial.

CONDITION This will happen all the time unless you Attach a condition

Name: Custom 404 (Required)

The name of your response, such as **My response**.

Status: 404 Not Found

The HTTP status code to include in the header of the response.

MIME Type: text/html

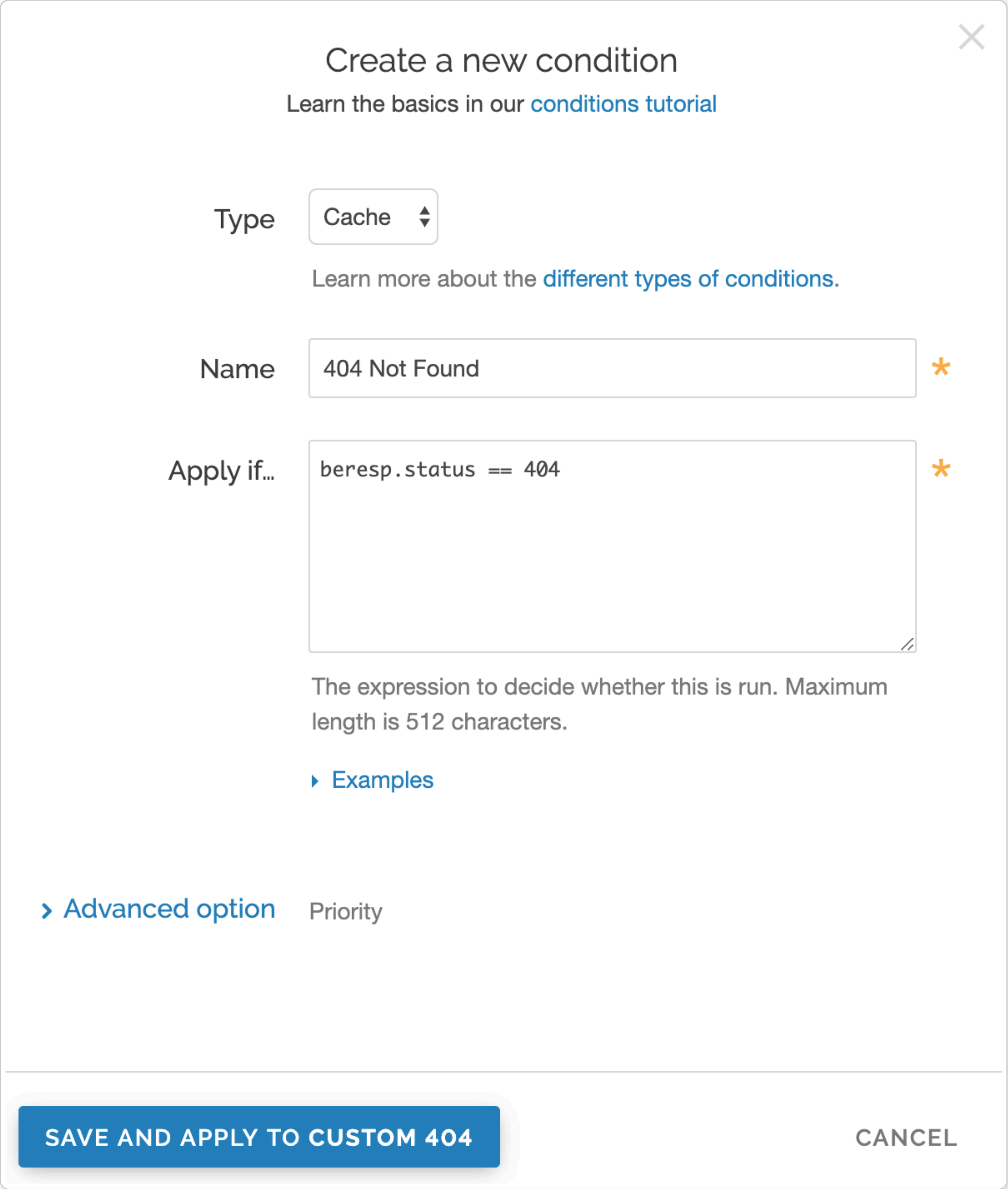
The media type to put into the Content-Type header which informs the browser how to display the response. Common MIME types are **application/json**, **text/plain**, **text/html**.

Response:

```
<html>
<body>
<h1>Custom Handler - Error 404</h1>
</body>
</html>
```

The content to be served when delivering the response.

6. Fill out the **Create a synthetic response** fields as follows:
    - In the **Name** field, enter a name for the response you're creating (e.g., `Custom 404`).
    - From the **Status** menu, select the appropriate status (e.g., `404 Not Found`).
    - In the **MIME Type** field, specify the Content-Type of the response (e.g., `text/html`).
    - In the **Response** field, enter the content to be served when delivering a response.
7. Click **Create**.
8. Click the **Attach a condition** link to the right of the name of your new response.



Create a new condition

Learn the basics in our conditions tutorial

Type: Cache

Learn more about the different types of conditions.

Name: 404 Not Found

Apply if...

```
beresp.status == 404
```

The expression to decide whether this is run. Maximum length is 512 characters.

9. Fill out the **Create a new condition** fields as follows:
   - From the **Type** menu, select the type of condition you're creating (e.g., `Cache`).
   - In the **Name** field, enter a name for the condition you're creating (e.g., `404 Not Found`).
   - In the **Apply if** field, enter the condition under which the new response occurs in the following format:

```
beresp.status == ###
```

where `###` equals the status condition you're creating the response for. For example, using the value of `beresp.status == 404` in the **Apply if** field here tells Fastly to use this response object whenever origin servers return a 404 status. (See the Conditions guides for more detailed information on conditions.)

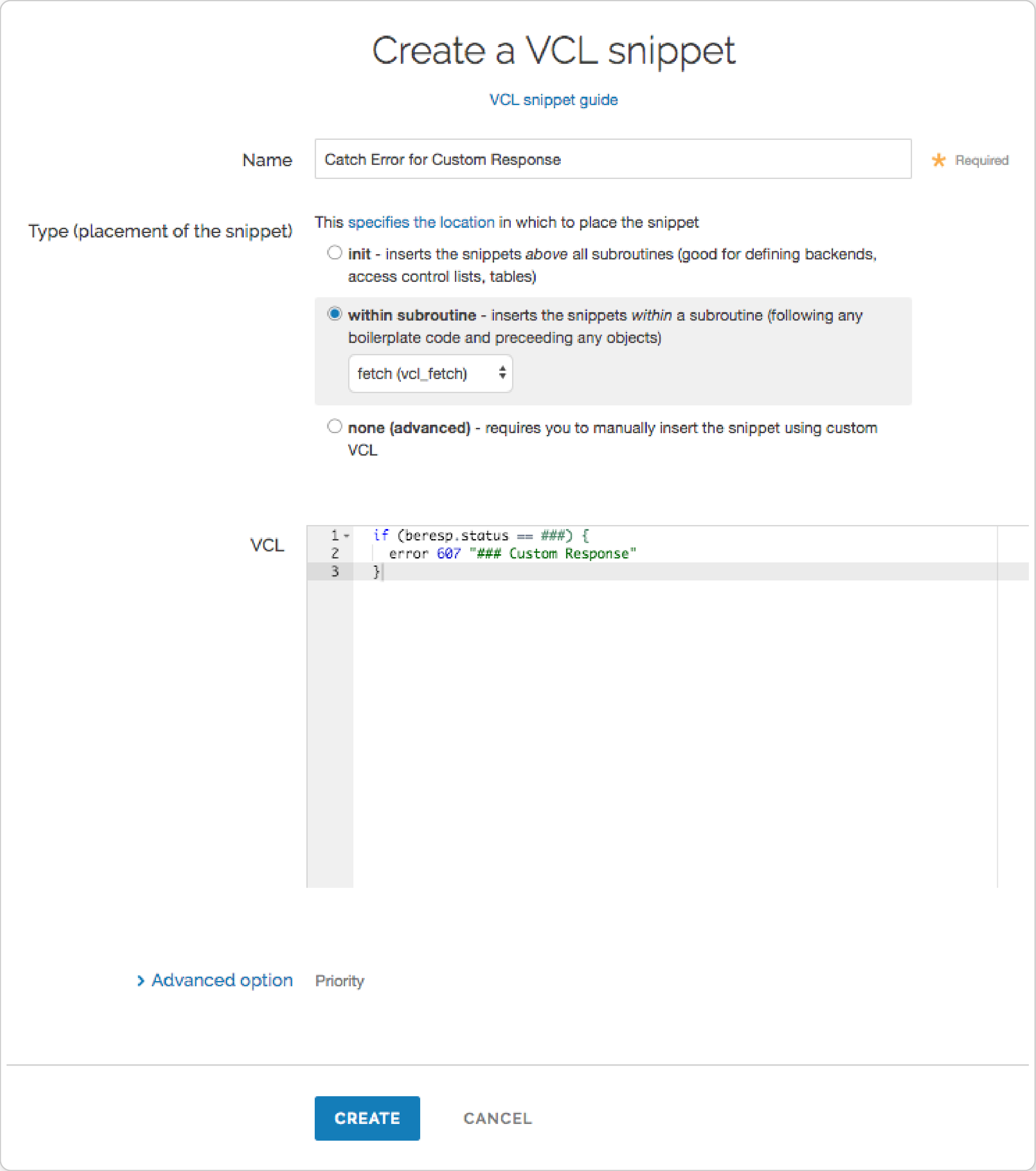
10. Click **Save and apply to**.
11. Click **Activate** to deploy your configuration changes. Fastly will now serve your custom HTML error page when required.

Creating custom responses using VCL Snippets

To create the custom response using VCL Snippets, create two separate snippets: one to trigger the condition for an internal Fastly error and the second to create the response to that error.

Create a VCL Snippet for a condition

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create Snippet**.



Name: Catch Error for Custom Response

Type (placement of the snippet): This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

fetch (vcl_fetch)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
if (beresp.status == ###) {
  error 607 "### Custom Response"
}
```

6. In the **Name** field, enter an appropriate name (e.g., `Catch Error for Custom Response`).

7. From the **Type** controls, select **within subroutine**.

8. From the **Select subroutine** menu, select **fetch (**`vcl_fetch`**)**.

9. In the **VCL** field, add the following condition:

```
if (beresp.status == ###) {
  error 600 "### Custom Response"
}
```

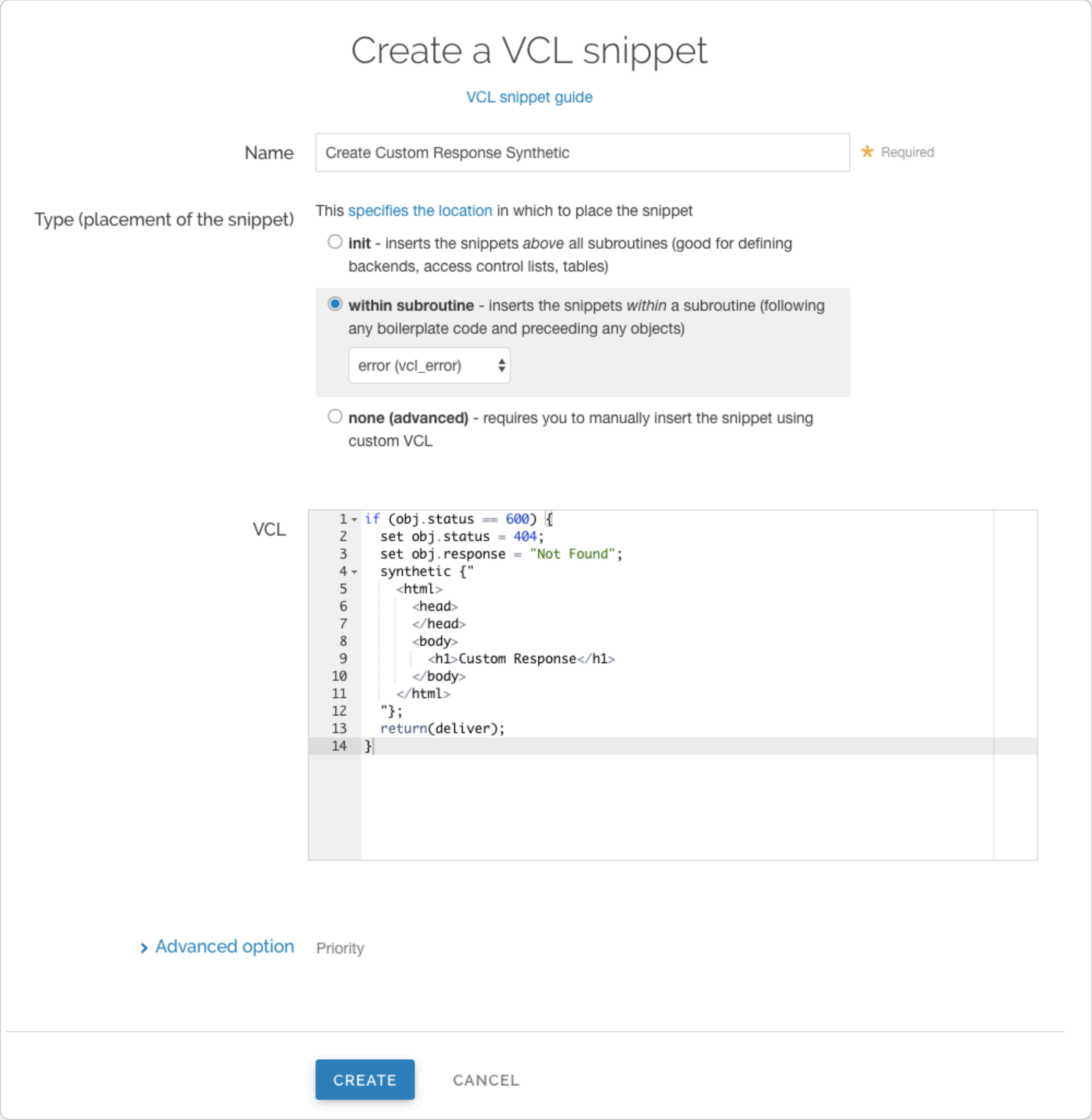
where `###` is the status condition you're creating the response for. The error code used here, `600`, is a random number that doesn't conflict with standard HTTP error codes. Consider using custom error code numbers in the 600’s or 700’s to avoid confusion.

10. Click **Create** to create the snippet.

Create a VCL Snippet for a synthetic response

1. Click **VCL Snippets**.

2. Click **Create Snippet**.



Create a VCL snippet

VCL snippet guide

Name: Create Custom Response Synthetic ★ Required

Type (placement of the snippet): This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

error (vcl_error)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL:

```
if (obj.status == 600) {
  set obj.status = 404;
  set obj.response = "Not Found";
  synthetic {"
    <html>
      <head>
      </head>
      <body>
        <h1>Custom Response</h1>
      </body>
    </html>
  "};
  return(deliver);
}
```

3. In the **Name** field, enter an appropriate name (e.g., `Create Custom Response Synthetic`).

4. From the **Type** controls, select **within subroutine**.

5. From the **Select subroutine** menu, select **error (**`vcl_error`**)**.

6. In the **VCL** field, add the following condition:

```
if (obj.status == 600) {
  set obj.status = 404;
  set obj.response = "Not Found";
  synthetic {"
    <html>
      <head>
      </head>
      <body>
        <h1>Custom Response</h1>
      </body>
    </html>
  "};
  return(deliver);
}
```

replacing `Custom Response` with your custom, synthetic response. This VCL tells Fastly to respond with your custom response if a request for an object meets the condition you created in `vcl_fetch`.

> NOTE
>
> Synthetic responses don't have a character limit, but including them in the custom VCL file may push that file over its size limit.

7. Click **Create** to create the snippet.
8. Click **Activate** to deploy your configuration changes.

* * *

Redirecting apex domains, wildcard domains, and subdomains

/en/guides/redirecting-apex-domains-wildcard-domains-and-subdomains

---

You can use the **Redirect traffic to www subdomains** setting to redirect traffic for apex domains, wildcard domains, or subdomains to a www subdomain so that users always arrive in a consistent location. For example, you could redirect requests from `example.com` and `users.example.com` to `www.example.com`.

Prerequisites

Before you redirect traffic to a www subdomain, you must add the domain to your Fastly service and update the domain's DNS records to point to Fastly. For example, to redirect `example.com` to `www.example.com`, you would add `example.com` to your Fastly service and create a DNS A record to point that domain at Fastly.

Adding the redirects

To redirect traffic for an apex domain, wildcard domain, or subdomain, follow the steps below:

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. In the **Redirect traffic to www subdomains** area, click **Add apex redirect**.
6. From the **Domains** menu, select the apex domain, wildcard domain, or subdomain you want to redirect.
7. From the **Status** menu, select the HTTP status code to send when redirecting traffic from the domains you selected.
8. Click **Add**.
9. Repeat these steps for any other apex domains, wildcard domains, or subdomains you want to redirect.
10. Click **Activate** to deploy your configuration changes.

* * *

Responses tutorial

/en/guides/responses-tutorial

Fastly allows you to create custom HTTP responses that are served directly from the cache without storing the page on a server. Responses are commonly used to serve small static assets that seldom change and maintenance pages that are served when origins are unavailable. This tutorial shows you how to create your own responses.

> NOTE
>
> We assume that you already know how to edit and deploy configurations using the web interface. If you are not familiar with basic editing using the application, see our help guides to learn more.

Creating a quick response

Fastly provides features that allow you to quickly enable and configure responses for a robots.txt file and 404 and 503 errors. For more information, see our guides on creating and customizing a robots.txt file and creating error pages with custom responses.

Creating an advanced response

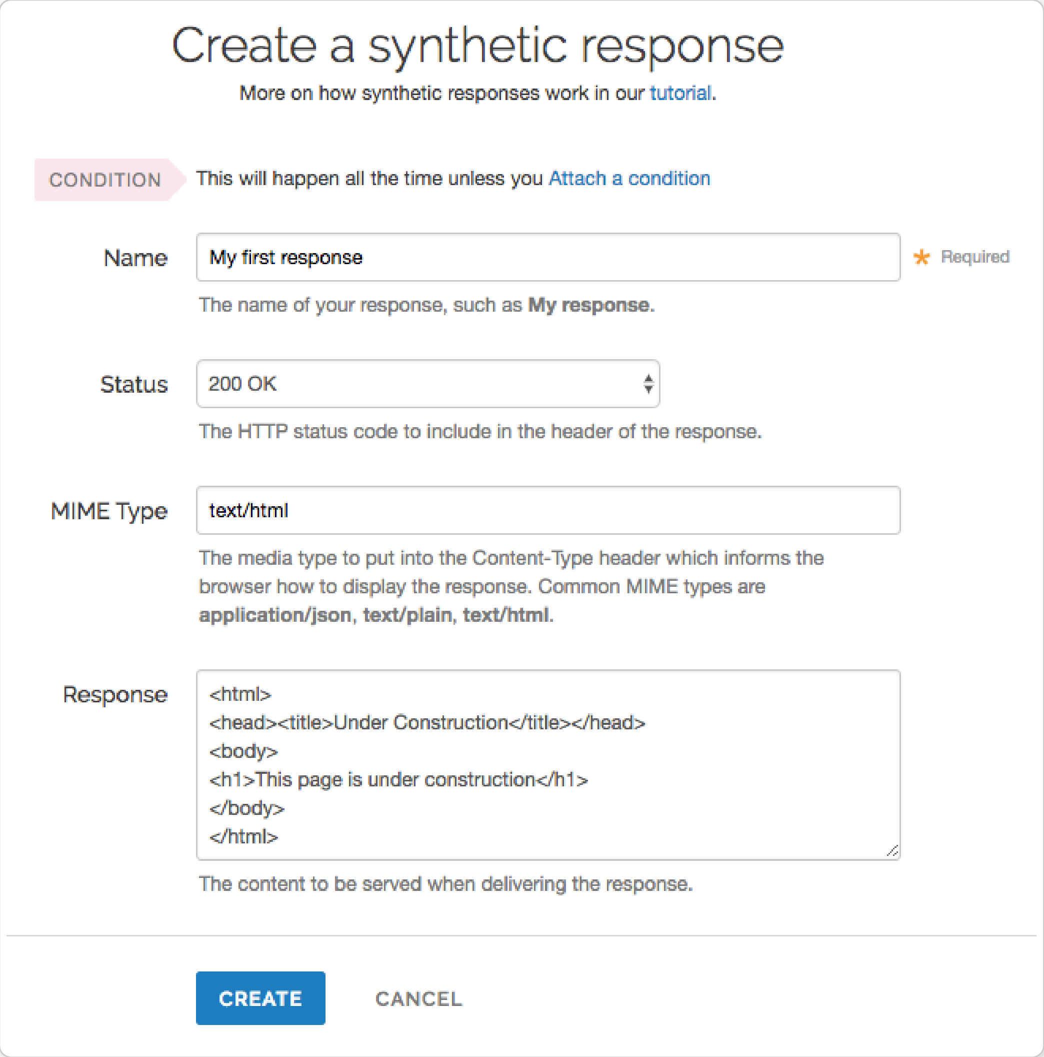
You can create an advanced response to specify the HTTP status code, MIME type, and content of the response. An advanced response has three basic attributes:

- Status - An HTTP status code to include in the header of the response
- Response - The content to be served when delivering the response

- Description - A human readable identifier for the response

By setting these three attributes and adding a condition to the response, you can very quickly get one up and running on your service. To create an advance response, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Set up advanced response**.



Create a synthetic response

More on how synthetic responses work in our tutorial.

CONDITION This will happen all the time unless you Attach a condition

Name: My first response ★ Required

The name of your response, such as **My response**.

Status: 200 OK

The HTTP status code to include in the header of the response.

MIME Type: text/html

The media type to put into the Content-Type header which informs the browser how to display the response. Common MIME types are **application/json**, **text/plain**, **text/html**.

Response:

```
<html>
<head><title>Under Construction</title></head>
<body>
<h1>This page is under construction</h1>
</body>
</html>
```

The content to be served when delivering the response.

6. Fill out the **Create a synthetic response** fields as follows:
   - In the **Name** field, enter a human-readable name for the response (e.g., `My first response`).
   - From the **Status** menu, select the appropriate status (e.g., `200 OK`).
   - In the **MIME Type** field, enter the content type of the response (e.g., `text/html`).
   - In the **Response** field, enter the response you want to appear when the conditions are met.

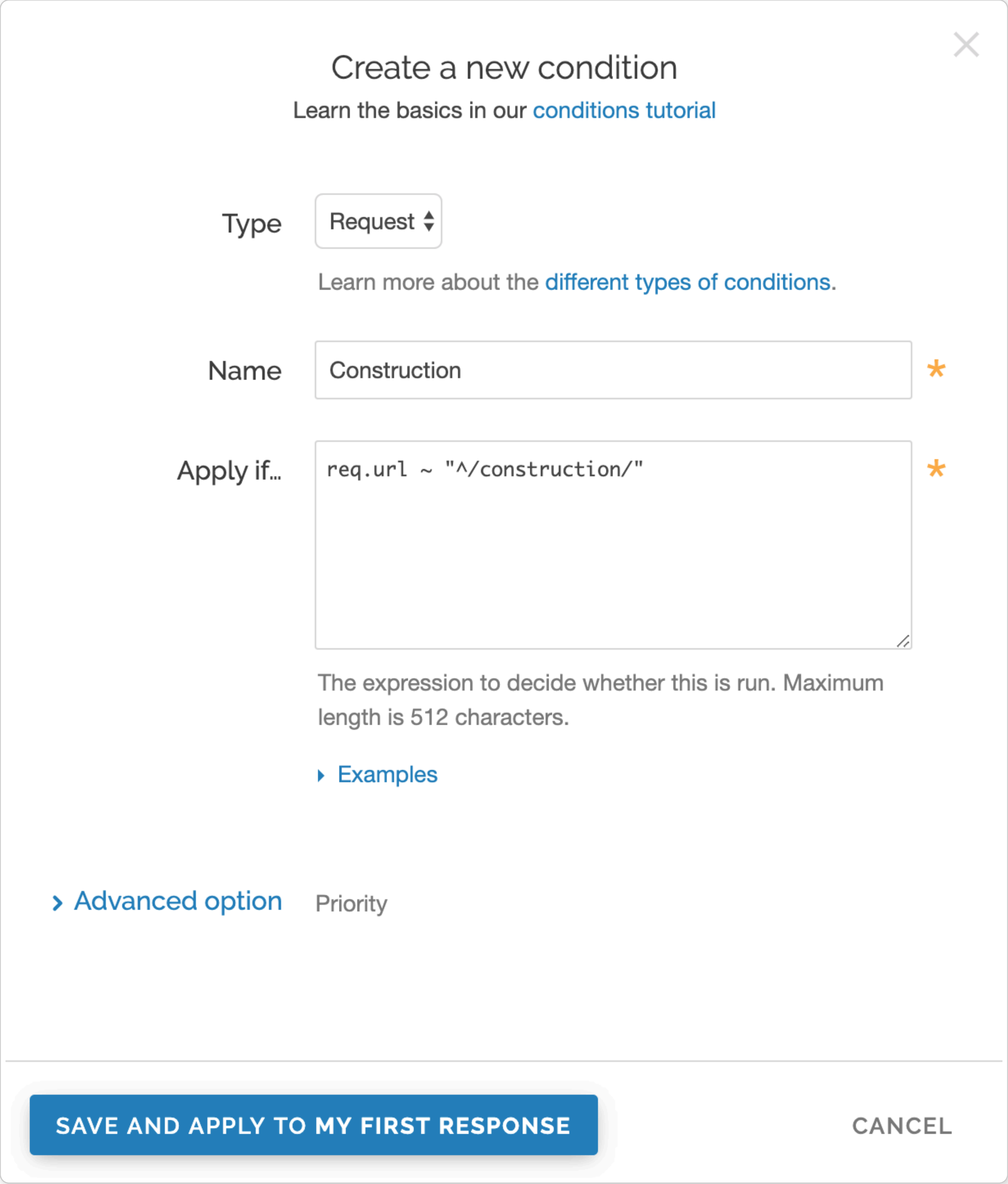
7. Click **Create** to create your custom response.

Your new response appears in the list of responses.

Adding conditions

To add a condition, follow the steps below:

1. Click **Attach condition** to the right of the new response.



Create a new condition

Learn the basics in our conditions tutorial

Type: Request

Learn more about the different types of conditions.

Name: Construction

Apply if...

```
req.url ~ "^/construction/"
```

The expression to decide whether this is run. Maximum length is 512 characters.

Examples
Advanced option
Priority
SAVE AND APPLY TO MY FIRST RESPONSE
CANCEL

2. Fill out the **Create a new condition** fields as follows:
    - From the **Type** menu, select the type of condition you want to create.
    - In the **Name** field, enter a human-readable name for the condition so that it can be easily identified in the future.
    - In the **Apply if** field, enter the condition under which the new response occurs. The condition should take the following format: `req.url ~ "^/construction/"` equals the request condition you're creating the response for.
    - In the **Priority** field, enter a priority if needed. Condition priorities are only needed in "interesting" cases, and can usually be left at the default "10" for all response conditions.
3. Click **Save and apply to**.
4. Click **Activate** to deploy your configuration changes.

Fastly now serves your custom response page when the condition is met.

* * *

Category: Video

These articles describe setup and configuration guidelines for setting up live stream delivery or video on-demand.

Adaptive bitrate playback URL guidelines

🔗 /en/guides/adaptive-bitrate-playback-url-guidelines

Fastly's On-the-Fly Packaging (OTFP) service supports any directory structure you might use to store different quality levels of a video. To construct adaptive bitrate (ABR) playback URLs for a video, make directory paths to that video unique. Ensure all the files associated with a particular video (e.g., quality levels, subtitles) exist under a single directory.

IMPORTANT

If you aren't sure how to configure OTFP, contact support before making any changes.

For example, say you had a video called Example Video. Assuming you had multiple quality levels and associated files for Example Video, the following directory structure would provide the best start to constructing ABR playback URLs:

Directory path example	**Description**
`/foo/bar/example-video/`	Base folder unique to this video
`/foo/bar/example-video/480p_30fps.mp4`	Quality level 480p with 30 frames per sec with audio

Directory path example	Description
`/foo/bar/example-video/720p_30fps.mp4`	Quality level 720p with audio with 30 frames per sec with audio
`/foo/bar/example-video/720p_60fps.mp4`	Quality level 720p with audio with 60 frames per sec with audio
`/foo/bar/example-video/1080p_30fps.mp4`	Quality level 1080p with audio with 30 frames per sec with audio
`/foo/bar/example-video/1080p_60fps.mp4`	Quality level 1080p with audio with 60 frames per sec with audio
`/foo/bar/example-video/4k_30fps.mp4`	Quality level 4k with audio with 30 frames per sec with audio

With this directory structure, the ABR playback URL for all videos in the base directory would follow this template:

```
http://example.com/path/to/dir/<video_id>/<quality_file1_name_wo_ext>,<quality_file2_nam wo
```

For example, the ABR playback URLs for Example Video in every format would be:

Format	Example URL
HDS	`http://example.com/foo/bar/example-video/480p_30fps,720p_30fps,720p_60fps,1080p_30fps,1080p_60fps,4k_30fps/master.f4m`
HLS	`http://example.com/foo/bar/example-video/480p_30fps,720p_30fps,720p_60fps,1080p_30fps,1080p_60fps,4k_30fps/master.m3u8`
MPEG-DASH	`http://example.com/foo/bar/example-video/480p_30fps,720p_30fps,720p_60fps,1080p_30fps,1080p_60fps,4k_30fps/master.mpd`

You can reduce the duplication in ABR playback URLs separating out the repeated prefix and suffix info as follows:

```
<filename_prefix><filename_variable><filename_suffix_wo_ext>.mp4
```

and the template would change to one of the following:

```
http://example.com/path/to/dir/<video_id>/<filename_prefix><quality_file1_variable_name_ _e
```

```
http://example.com/path/to/dir/<video_id>/<filename_prefix><quality_file1_variable_name_ _e
```

```
http://example.com/path/to/dir/<video_id>/<quality_file1_variable_name>,<quality_file2_v ia
```

IMPORTANT

To use token validation with ABR manifest URLs, special modifications must be made using custom VCL. Contact support for assistance.

* * *

Collecting OTFP metrics

/en/guides/collecting-otfp-metrics

Fastly allows you to collect and process On-the-Fly Packaging (OTFP) service metrics for analysis using a combination of custom VCL updates and specific log streaming settings. Once you've set up OTFP metrics collection through remote log streaming you can use any of a number of third-party and open source software options to aggregate your logging data for visualization and further analysis.

IMPORTANT

If you aren't sure how to configure OTFP, contact support before making any changes.

Upload custom VCL

1. Before uploading custom VCL, review the caveats of mixing and matching Fastly VCL with custom VCL.
2. Add the following custom VCL to your Fastly VCL:

```
sub vcl_deliver {
  # Identify Request type
  if (req.url.ext ~ "m3u8|ts|aac|webvtt") {
    set resp.http.Otfp-Format = "HLS";
  } else if (req.url.ext ~ "mpd|m4s") {
    set resp.http.Otfp-Format = "DASH";
  } else {
    set resp.http.Otfp-Format = "OTHER";
  }

  # Extract name-value pairs Otfp Info herder
  if (resp.http.X-Fastly-Otfp-Info) {
    set resp.http.Otfp-SS = regsub(resp.http.X-Fastly-Otfp-Info, ".*ss=(\S+).*", "\
    set resp.http.Otfp-SL = regsub(resp.http.X-Fastly-Otfp-Info, ".*sl=(\S+).*", "\
    set resp.http.Otfp-VL = regsub(resp.http.X-Fastly-Otfp-Info, ".*vl=(\S+).*", "\

    # Resolution (rs name-value) not available for audio-only segments
    if (resp.http.X-Fastly-Otfp-Info ~ ".*rs=(\S+).*") {
      set resp.http.Otfp-RS = re.group.1;
    } else {
      set resp.http.Otfp-RS = "-";
    }
  }
#FASTLY deliver

  return(deliver);
}
```

Create a logging endpoint

Follow the instructions to set up remote log streaming for your account and when creating your specific logging endpoint, set the **Format String** field to the following:

```
%h now.sec %r %>s %b resp.http.Otfp-Format resp.http.Otfp-SS resp.http.Otfp-SL resp.http tf
```

Control log file timing with a logging endpoint condition

To avoid excess log files, consider attaching a condition to the logging endpoint so logs are only sent when video segments are requested so that logging specifically exclude those files sent from Fastly's Origin Shield.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**.
5. In the list of logging endpoints, find the endpoint you enabled when setting up remote log streaming, then click **Attach a condition.**
6. Fill out the **Create a new condition** window as follows:
    - In the **Name** field, enter a human-readable name for the condition.
    - In the **Apply if** field, enter `resp.http.X-Fastly-Otfp-Info && !req.http.Fastly-FF`.
7. Click **Save and apply to.**

Analyze logging data

In addition to any Varnish variable, and a variety of Fastly's extensions to VCL, log files include the following video-specific fields:

- `ss` - video segment start presentation time in seconds
- `sl` - video segment duration in seconds
- `vl` - video duration in seconds
- `rs` - video track display resolution in pixels

You can use these fields to run queries for analysis and use what you discover to refine your video delivery settings.

* * *

Common OTFP errors

/en/guides/common-otfp-errors

This page lists some possible error values that Fastly's On-The-Fly Packaging service (OTFP) service will send in the `X-Fastly-Package-Error` response header when attempting to fetch video data from your origin server or when repackaging your MP4 files. For help troubleshooting, send the full URL of the response to our customer support team.

NOTE

There are certain cases where you may see one of these messages as a warning instead of an error.

404 responses

In a 404 response from OTFP, the following header value is returned:

`Origin 404` - This error occurs if the source MP4 does not exist on origin.

5xx responses

In a 5xx response from OTFP, header values might include the following.

500 responses

- `Service configuration error` - This error typically occurs when something is wrong with one of the cache control headers that determine the behavior of an OTFP service.
- `Incompatible or corrupt origin media` - This error typically occurs when the remote MP4 being requested is corrupt or the requested file doesn't actually use the MP4 container format.
- `Slice out of bounds` - This error occurs when the start and end markers of a video do not align with the actual duration of that video.
- `Video track has no key frames` - This error occurs when the source video MP4 file does not include an `stss` box, which has information about key frames.
- `Flapping origin entity` - This error occurs if conflicting entity tags (ETags) were seen when handling a request. It's possible that the file changed legitimately during the request, and a follow-up request should work as expected. If the problem persists, it indicates that the origin can't handle ETags correctly.
- `File is not a WebVTT` - This error occurs when Fastly is unable to parse the remote WebVTT file.
- `Unexpected EOF on origin fetch` - This error occurs because an attempt to fetch segment media failed due to the file's size being smaller than its index indicated.
- `Track media is not in order` - This error occurs if Fastly cannot handle the MP4 because the media in its media data box (`mdat` box) is out of order, as indicated by the `stco` or `co64` box, which has information about the location of data in the media track's data stream.
- `moov box is too big` - This error occurs when the index portion of the MP4 is larger than 500MiB.

- `Mp4 has too many frames` - This error occurs when the total number of samples (between both the audio and video track) exceeds 40,000,000.
- `ftyp box is too big` - This error occurs if the file type box (`ftyp`) is larger than necessary.
- `Track lacked valid track id` - This error occurs if the Track ID was not set to a value greater than 0 in the `tkhd` box.
- `Track lacks valid timescale` - This error occurs if the `timescale` field in the media header box (`mdhd` box) did not have valid settings.
- `Track lacks valid duration` - This error occurs if the `duration` field in the media header box (`mdhd` box) did not have valid settings.
- `Track XXXX and XXXX do not agree` - This error indicates that the MP4 index is corrupt.
- `Empty XXXX, missing XXXX, illegal XXXX, ....` - This error indicates that the MP4 index is corrupt.
- `Unable to unmarshal expressplay response` - This error occurs when ExpressPlay is responding unexpectedly.
- `expressplay: params unable to match expected format` - This error indicates that the `X-Fastly-Drm` header has invalid base64 encoded data after `expressplay:`.
- `Origin segment media too large` - This error indicates that the file must be repackaged.
- `Unsupported audio codec 0x<##>` - This error indicates that the video/audio is encoded with an audio codec that's not supported by OTFP. Example: `Unsupported audio codec 0x69`.
- `No audio/video samples found. Fragmented or malformed MP4 detected.` - This error indicates that the source MP4 provided is likely a fragmented MP4 (fmp4) or not an MP4.
- `HEVC in HLS transport stream is not supported` - This error indicates that you need to use fragmented MP4 segments (.fmp4) instead of transport stream (.ts) requests. (Transport stream requests are not compatible with HEVC source video files.) You can specify use of fragmented MP4 segments by including the `X-Fastly-Hls-Fragmented-Mp4: true` header in your VCL request. If, after setting that header, you are experiencing playback issues on Apple devices using HEVC, you can try changing the header value to `force-hvc1`, which will ensure that OTFP is using `hvc1` sample boxes instead of `hev1`: `X-Fastly-Hls-Fragmented-Mp4: force-hvc1`. This can sometimes fix mislabeled streams.

502 responses

- `Invalid response from origin` - This error typically occurs when the response from the server is something other than an expected 206, 404, or 5xx response. For example, returning a 403 error instead of a 404 when a file is missing.
- `Invalid Content-Range from origin` - This error occurs if the origin responded to a Range request with a Content-Range header that was illegal or did not match the request Fastly made.
- `Failed to reach origin` - This error typically indicates that a path or bucket is incorrectly set in X-Fastly-Origin.
- `Validator supplied but ignored` - This error indicates that the origin is not handling ETags correctly. Specifically, when asked If-None-Match, instead of responding with a 304 status, the origin responded with a 2xx status that had the same ETag. Check out RFC 7232, section 3.2 for details.

504 responses

`Origin timeout` - This error occurs if the OTFP was unable to connect or fetch content from origin inside of 15 seconds.

* * *

Streaming configuration guidelines

/en/guides/streaming-configuration-guidelines

The Fastly network can deliver live streams for any HTTP streaming technology, archived or recorded, on any public or private cloud storage service. When configuring VCL to deliver live streams, we recommend following these guidelines, which Customer Support can help you with.

Configure shielding

Configure shielding by designating a specific shield POP for your origin to ensure live streams remain highly available within the Fastly network. If your setup includes primary and alternate origins (e.g., for high profile live streams), be sure to select a shield POP close to each origin, one for each origin you define.

Configure video manifest and segment caching TTLs

In live streams, video manifests are periodically refreshed when new segments become available, specially for HLS. We recommend setting manifest file TTLs to less than half of the video segment duration, typically 1-2 seconds for 5-second video segments. For long DVRs and live-to-VOD transitions, set segment TTLs longer on shields and shorter on edge POPs such that they are served from memory (that is, less than 3600s).

The following VCL sample may help you implement different TTLs for video manifest and segments. It can also be added to your service using VCL Snippets:

```
sub vcl_fetch {
#FASTLY fetch

  # Set 1s ttls for video manifest and 3600s ttls for segments of HTTP Streaming format
  # Microsoft Smooth Streaming format manifest and segments do not have file extensions
  # Look for the keywords "Manifest" and "QualityLevel" to identify manifest and segmen
  if (req.url.ext ~ "m3u8|mpd" || req.url.path ~ "Manifest") {
    set beresp.ttl = 1s;
    return (deliver);
  }
  else {
    if (req.url.ext ~ "aac|dash|m4s|mp4|ts" || req.url.path ~ "QualityLevel") {
      set beresp.ttl = 3600s;
      return (deliver);
    }
  }

  return (deliver);
}
```

Optionally, identify video manifests and segments using the MIME type.

Configure lower TTLs for errors

By default, Fastly honors the `Cache-Control` header from the origin to set TTLs for cacheable objects. However, origins may not send `Cache-Control` headers for non-200 or 206 HTTP status code responses. As a result, Fastly will only cache few status code responses with default TTLs configured, usually 3600s, to prevent large numbers of requests from hitting the origin. Uncacheable status code responses can be enabled for caching by setting `beresp.cacheable` flag to `true`.

For live streams, new video segments are added every few seconds. Typically, live stream transcoders are configured to generate 5s segments and manifests are refreshed after each new segment is available. Frequently, video players can make requests to segments not yet available or requests can return errors like 500 or 503 status codes. In such cases, status code responses should be made cacheable and should only be cached with TTLs small enough to give sufficient time for origins to recover (around 1s).

The following VCL sample may help you implement this and can also be added to your service using VCL Snippets:

```
sub vcl_fetch {
#FASTLY fetch

  # Set 1s ttl if origin response HTTP status code is anything other than 200 and 206
  if (!http_status_matches(beresp.status, "200,206")) {
    set beresp.ttl = 1s;
    set beresp.cacheable = true;
    return (deliver);
  }

  return (deliver);
}
```

Configure Streaming Miss

Configure Streaming Miss to reduce the time clients (players) must wait to begin downloading streams when Fastly's edge servers must fetch content from your origin. Streaming Miss should be enabled for video or audio objects only (these are sometimes called *chunks* or *segments*).

The following VCL sample may help you implement this. It can also be added to your service using VCL Snippets:

```
sub vcl_fetch {
#FASTLY fetch

  # Enable Streaming Miss only for video or audio objects.
  # Below conditions checks for video or audio file extensions commonly used in
  # HTTP Streaming formats.
  if (req.url.ext ~ "aac|dash|m4s|mp4|ts") {
    set beresp.do_stream = true;
  }

  return (deliver);
}
```

Configure automatic compression

Configure automatic compression for manifest files based on their file extension or content-type using the following table as a guide:

HTTP streaming format	file extension	content-type
Apple HLS	m3u8	application/x-mpegurl, application/vnd.apple.mpegurl
MPEG-DASH	mpd	application/dash+xml
Adobe HDS	f4m, bootstrap	application/f4m (for manifest), application/octet-stream (for bootstrap)
Microsoft HSS	N/A	application/vnd.ms-sstr+xml

Configure CORS

Configure a CORS header on your service to play audio or video content on a different domain. Clients can potentially send CORS pre-flight requests to determine if the main request is OK to send.

By default, OPTIONS requests will pass to origin. Instead of this, we can respond with a synthetic response from the CDN. We also have an example for handling OPTIONS specifically with a synthetic. If allowing OPTIONS requests to traverse to the origin, we recommend caching OPTIONS requests.

Configure methods

With a media origin, the requests are typically only for GET and HEAD methods, so it's worth blocking other requests that are not expected by the origin. By default, Fastly will pass on GET, HEAD, and FASTLYPURGE methods. We can block all other requests by using custom VCL to send a 405 error back to the client:

```
sub vcl_recv {
#FASTLY recv

  if (req.method != "HEAD" && req.method != "GET" && req.method != "FASTLYPURGE") {
    error 405; # This is a `return(pass);` by default
  }

return(lookup);
}
```

NOTE

Consider the logic here carefully. The PUT/PATCH and DELETE methods are likely only required when dealing with the origin's hostname directly versus the domain used for clients. OPTIONS requests should be handled synthetically if you intend to block them from going to the origin. Failing to do so could cause failure scenarios.

Enable the experimental BBR congestion algorithm

The BBR TCP congestion control algorithm is an optional, TCP-related configuration that can help improve a client's experience. Unlike the default CUBIC congestion control algorithm, which is packet-loss-based and latency-insensitive, BBR is designed to maximize bandwidth while controlling latency.

WARNING

While expected to perform better than CUBIC (especially under transient packet losses), BBR is still a work-in-progress and implementing it may cause performance degradation for some users.

You can implement this algorithm by adding the following VCL to your service using VCL Snippets:

```
sub vcl_deliver {
#FASTLY deliver

  # set congestion algorithm for client requests
  if (!req.http.Fastly-FF && client.requests == 1) {
    set client.socket.congestion_algorithm = "bbr";
  }

  return(deliver);
}
```

TIP

TCP optimizations can be applied conditionally rather than applying them to all clients. For example, enable BBR only for clients within a specific ASN or ISP network like a mobile or wireless network.

Configure origin timeouts

Set appropriate origin timeouts to ensure new live stream segments are downloaded from origin in a timely manner. For example, for a live stream with 5s video segments, set the Origin Connect value to 1s and the First Byte and Between Bytes timeout values to 2s. Typically, these values should be configured such that Fastly can also retry another origin (if configured) before sending the appropriate response on client requests.

Consider setting up failover (fallback) origins

Consider configuring your VCL to allow your origins to failover from high-profile primary streams to alternate streams in case of encoder failures or other issues (e.g., high resource utilization).

Configure real-time log streaming

For troubleshooting and debugging live streaming delivery issues, configure real-time log streaming and include TCP connection, caching, and different time-related metrics in `vcl_log`. For example, consider including:

- `fastly_info.state` (cache hits or misses)
- `client.socket.tcpi_rtt` (client round-trip time)
- `time.to_first_byte` (time from client request to the first byte being received)
- `time.elapsed` (time since the request started, which can be used to calculate response time or time-to-last-byte for both origin and clients)
- `client.as.number` and `client.as.name` (autonomous system number and name associated with client IP)
- `client.socket.tcpi_delta_retrans` (number of packets re-transmitted to the client)
- `client.socket.tcpi_snd_mss` (maximum segment size used to send responses to client)
- `client.requests` (number of requests on a connection so far)
- `client.socket.nexthop` (network path Fastly is sending the client response)
- `req.restarts` (number of request restarts typically indicates retry attempts)
- `server.datacenter` (the Fastly POP that served the request)
- `resp.http.content-length` and `resp.body_bytes_written` (actual bytes sent to client compared to what was expected to be sent)

These metrics can help you analyze throughput and may help you determine reasons a video player might switch quality levels during ABR playback.

Take advantage of surrogate key purging

All video segments and the manifest for a live stream can be purged using a single API call by using Fastly's surrogate key feature.

Manage live-to-VOD smoothly

Most encoders generate a separate video manifest when making the same live stream available for VOD. If your VOD manifest has the same URL as the live one, purge the live stream video manifest or wait for the caches to invalidate (as they will be set with low TTLs). If your setup archives the live stream as progressive mp4s, consider delivering them using Fastly's OTFP service.

NOTE

Wowza integrations. When configuring your Wowza origin server, be sure to select the Live HTTP Origin application type. If you select Live Edge, Wowza will always return a unique URL for manifest requests, resulting in extremely low cache hit.

* * *

Getting started guides

Category: Basics

These articles provide basic information and instructions for configuring Fastly services after getting started.

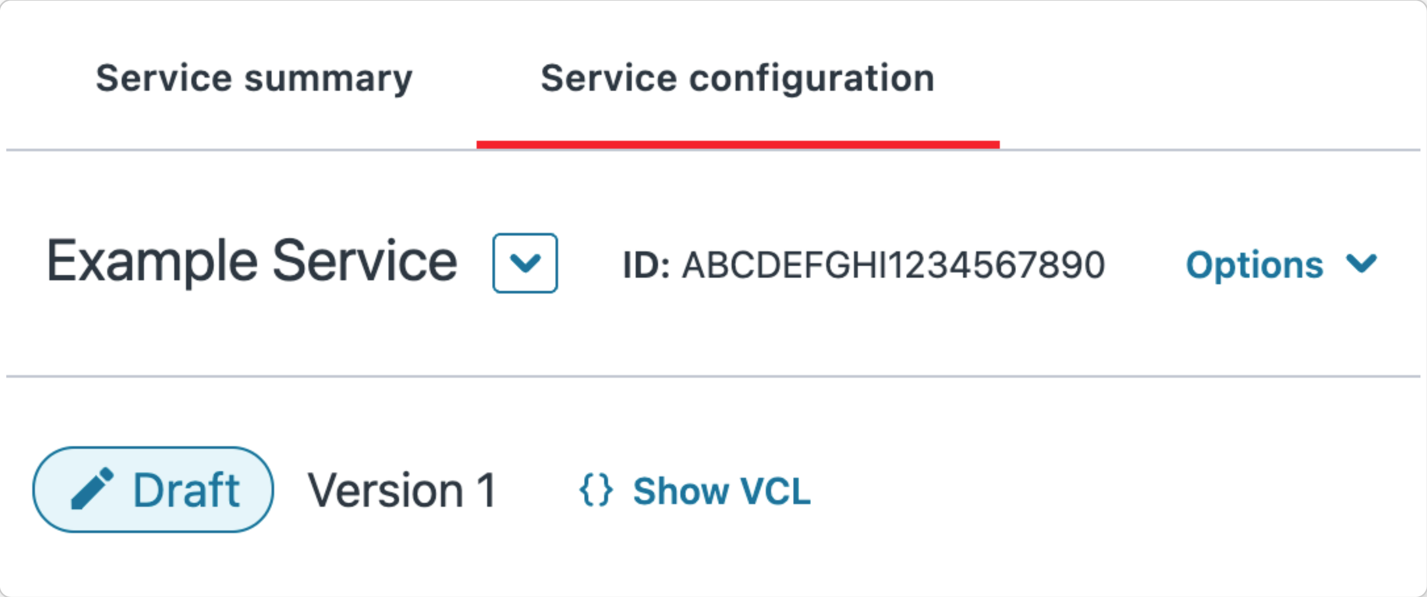
About services

/en/guides/about-services

A Fastly service is a user-defined set of caching rules and behaviors for a website or application. You might create new Fastly services to do things like:

- add a new website under your control to your list of web properties
- add a new domain to your growing list of existing domains already served by Fastly
- isolate traffic metrics for specific digital assets, like a site's images

Every service tied to your account contains a unique ID that appears next to the name of your service on any page.



Once created, you can edit and activate new versions of your Fastly services that include refinements and updates to your configurations. The web interface also allows you to do other things with existing Fastly services, like compare them to each other, deactivate or reactivate them, and delete them.

Before you begin

Before you begin working with Fastly services, be sure you understand how caching and CDNs work. You'll also need to understand the Fastly web interface controls before using them to work with your Fastly services.

Understanding Fastly services and versions

When you create a service, before it will do anything, you need to configure it. This configuration will tell Fastly how to handle traffic through your service. Officially, a configuration for a service is called a *version* of that service. You'll notice that your new service has a single, initial version (all new services start their existence this way). The initial version of a service is an inactive blank slate, waiting for you to configure and *activate* it. Once you activate your first version, your service will begin handling traffic according to your configurations.

Your service can have many versions. Each version has its own status:

- **Draft** is the status that every version starts with when it's first created. This status indicates that the version hasn't been *activated* and is still being edited. Only versions with draft status can be edited.
- **Active** is the status displayed by a version after you activate it. When you activate a version, you are telling Fastly to use that configuration for your service's traffic. Only one version of a service can be active at a time. Once a service becomes active, it also becomes *locked*, so it can no longer be edited, only *deactivated* or *reactivated*.
- **Locked** is the status displayed by a version after you deactivate it. This status primarily indicates that a version can't be edited. Active versions are also locked automatically when they first become active, but they display as active, not as locked.

Eventually, your service will probably have more versions displaying the locked status than any other. At some point, you may decide that one of these locked versions has a configuration you prefer to what is currently active. In that case, you can simply reactivate that version. Or, if some aspects of that version aren't viable anymore, you can *clone* it and make changes to the clone, which will have the draft status. Fastly maintains a complete record of the version history of your service to ensure that you can always roll a service back to any version.

Service creation limits

- **Accounts are limited to a set number of services.** We set a limit on the number of services you can create per account by default based on your account type and any packaged offering you've purchased. Reach out to sales@fastly.com for details on how to increase this limit.
- **Each service is limited to a set number of domains.** We set a limit on the number of domains you can create per service by default based on your account type and any packaged offering you've purchased. However, if you contact support, we may be able to adjust this number for you by working with you to set up and fine-tune domain handling in your service.

What's next

Once you understand how services and versions work and their limits, create your first CDN service or Compute service.

* * *

	Content and its delivery
🔗	/en/guides/content-and-its-delivery

Content types delivered by Fastly

The underlying protocol used by the World Wide Web to define how content is formatted and transmitted is called the Hypertext Transfer Protocol (HTTP). Fastly's CDN Service delivers all HTTP-based file content (e.g., HTML, GIF, JPEG, PNG, JavaScript, CSS) including the following:

- Static content
- Dynamic content
- Video content

Each content type is described below.

Static content

Static content includes content that remains relatively unchanged. Fastly can control static content in two ways:

- using the time to live (TTL) method, where Fastly's cache re-validates the content after expiration of the TTL, or
- using Fastly's Instant Purge functionality, in which content remains valid until the cache receives a purge request that invalidates the content.

Examples of static content include images, CSS, and JavaScript files.

Dynamic content

Dynamic content includes content that changes at unpredictable intervals, but can still be cached for a fraction of time. We serve this dynamic content by taking advantage of Fastly's Instant Purge functionality. Using this functionality, dynamic content remains valid only until a Fastly cache receives a purge request that invalidates the content. Fastly understands that the rate of those purge requests cannot be predicted. Dynamic content may change frequently as a source application issues purge requests in rapid succession to keep the content up to date. Dynamic content can, however, remain valid for months if there are no changes requested.

Examples of dynamic content include sports scores, weather forecasts, breaking news, user-generated content, and current store item inventory.

Video content

Video content includes:

- Live video streams
- Video on Demand (VOD) content libraries

Video content can be served using standard HTTP requests. Specifically, Fastly supports HTTP Streaming standards, including HTTP Live Streaming (HLS), HTTP Dynamic Streaming (HDS), HTTP Smooth Streaming (HSS), and MPEG-DASH. For Fastly's CDN Service to deliver video, the video must be packaged.

Content sources supported by Fastly

Fastly caches deliver various types of content from many different sources. Supported sources include:

- Websites
- Internet APIs
- Internet Applications

- Live and Live Linear Video
- Video on Demand (VOD) Libraries

Regardless of the content source, the content's source server must communicate using HTTP. HTTP defines specific types of *methods* that indicate the desired action to be performed on content. The manner in which those HTTP methods are used (the standard, primary methods being GET, POST, PUT, and DELETE) can be labeled as being RESTful or not. Fastly supports RESTful HTTP by default, but also can support the use of non-RESTful HTTP as long as the method used is mapped to its appropriate cache function. Each of the content sources supported by Fastly are described in more detail below.

Websites

Websites are servers that provide content to browser applications (e.g., Google's Chrome, Apple's Safari, Microsoft's Internet Explorer, Opera Software's Opera) when end users request that content. The content contains both the requested data and the formatting or display information the browser needs to present the data visually to the end user.

With no CDN services involved, browsers request data by sending HTTP GET requests that identify the data with a uniform resource locator (URL) address to the origin server that has access to the requested data. The server retrieves the data, then constructs and sends an HTTP response to the requestor. When a CDN Service is used, however, the HTTP requests go to the CDN rather than the origin server because the customer configures it to redirect all requests for data to the CDN instead. Customers do this by adding a CNAME or alias for their origin server that points to Fastly instead.

Internet APIs

Application program interfaces (APIs) serve as a language and message format that defines exactly how a program will interact with the rest of the world. APIs reside on HTTP servers. Unlike the responses from a website, content from APIs contain only requested data and identification information for that data; no formatting or display information is included. Typically the content serves as input to another computing process. If it must be displayed visually to an end user, a device application (such as, an iPad, Android device, or iPhone Weather application) does data display instead.

Legacy internet applications

Legacy internet applications refer to applications not originally developed for access over the internet. These applications may use HTTP in a non-RESTful manner. They can be incrementally accelerated without caching, benefiting only from the TCP Stack optimization done between edge Fastly POPs and the Shield POP, and the Shield POP to the origin. Then caching can be enabled incrementally, starting with the exchanges with the greatest user-experienced delay.

Live and live linear video streams & video on demand libraries

Live and live linear video content (for example, broadcast television) is generally delivered as a *stream* of information to users, which they either choose to watch or not during a specific broadcast time. Video on demand (VOD), on the other hand, allows end users to select and watch video content when they choose to, rather than having to watch at a specific broadcast time.

Regardless of which type of video content an end user experiences, a video player can begin playing before its entire contents have been completely transmitted. End users access the video content from a customer's servers via HTTP requests from a video player application that can be embedded as a part of a web browser. Unlike other types of website content, this content does not contain formatting or display information. The video player handles the formatting and display instead.

When the video content is requested, the customer's server sends the content as a series of pre-packaged file chunks along with a manifest file required by the player to properly present the video to the end user. The manifest lists the names of each file chunk. The video player application needs to receive the manifest file first in order to know the names of the video content chunks to request.

Pre-packaging in this context refers to the process of receiving the video contents, converting or *transcoding* the stream into segments (chunks) for presentation at a specific dimension and transmission rate, and then packaging it so a video player can identify and request the segments of the live video a user wants to view.

To request video delivery on your account, contact your Fastly Account Representative at sales@fastly.com.

* * *

Logging in and using Fastly

🔗 /en/guides/logging-in-and-using-fastly

IMPORTANT

This guide only applies to customers with Fastly accounts or with Signal Sciences accounts linked to Fastly accounts. If you have a Signal Sciences account that isn't linked to a Fastly account, log in as usual.

The Fastly web interface allows you to use a web browser to log in from anywhere. If you have trouble logging in, refer to our guide on troubleshooting your login

Before you begin

Keep in mind the following:

- **Keep password managers up to date.** Most password managers associate entries in their databases according to specific website domains. If you've stored your username and password combination there for either your Fastly or Signal Sciences accounts, your password manager won't have any way of knowing when you've added or updated information. You must always manually enter and update the information in your password manager yourself.
- **Single sign-on (SSO) is enforced per your organization's settings.** If your organization has single sign-on (SSO) enabled for Fastly accounts or Signal Sciences accounts, you will be prompted to authenticate with SSO or will be automatically signed-in if you're already authenticated.
- **Two-factor authentication is enforced per you or your organization's settings.** If you or your organization has enabled two-factor authentication for Fastly accounts or Signal Sciences accounts, you'll need to enter a code generated by an authenticator app as part of the login process.

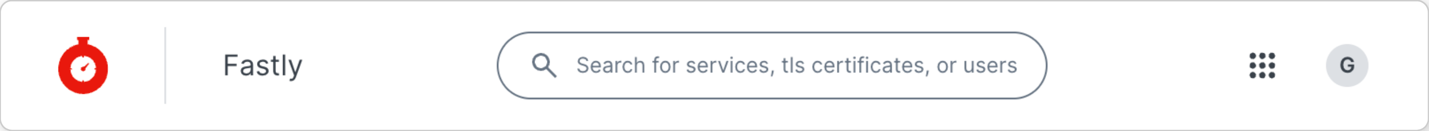
Logging in to your account with your email and password

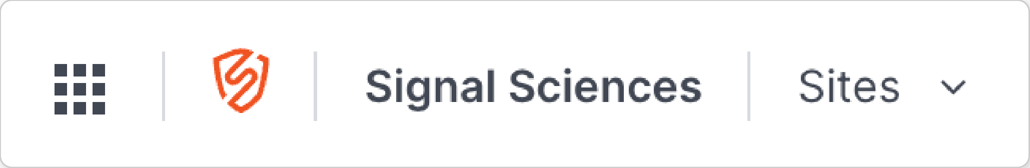
1. Navigate to the appropriate login page for your account, either https://manage.fastly.com/ or https://dashboard.signalsciences.net/.
2. In the **Email** field, enter the email address associated with your Fastly account

3. Click **Continue**.
4. In the **Password** field, enter the password associated with the email address you entered.
5. Click **Log in**.
6. *(Optional)* If your account has two-factor authentication enabled then, in the **Authentication code** field, enter the code generated by the authenticator app on your mobile device.
7. Click **Complete login**.

Switching between application control panels

Regardless of which account you log in to first, you can switch between the Fastly and Next-Gen WAF control panels via the switcher menu. Once you've logged in, it appears as nine squares in a three by three grid at the of the browser:





Clicking the switcher menu with the Fastly control panel active will open the Next-Gen WAF control panel in a new browser tab. Likewise, clicking the switcher menu with the Next-Gen WAF control panel active will open the Fastly control panel in a new browser tab.

Logging out of your account

To log out of your account, do one of the following:

- To log out of your Fastly account, select **Sign out** from the **user quick links menu**.
- To log out of your Signal Sciences account, select **Sign out** from the **My Profile menu**.

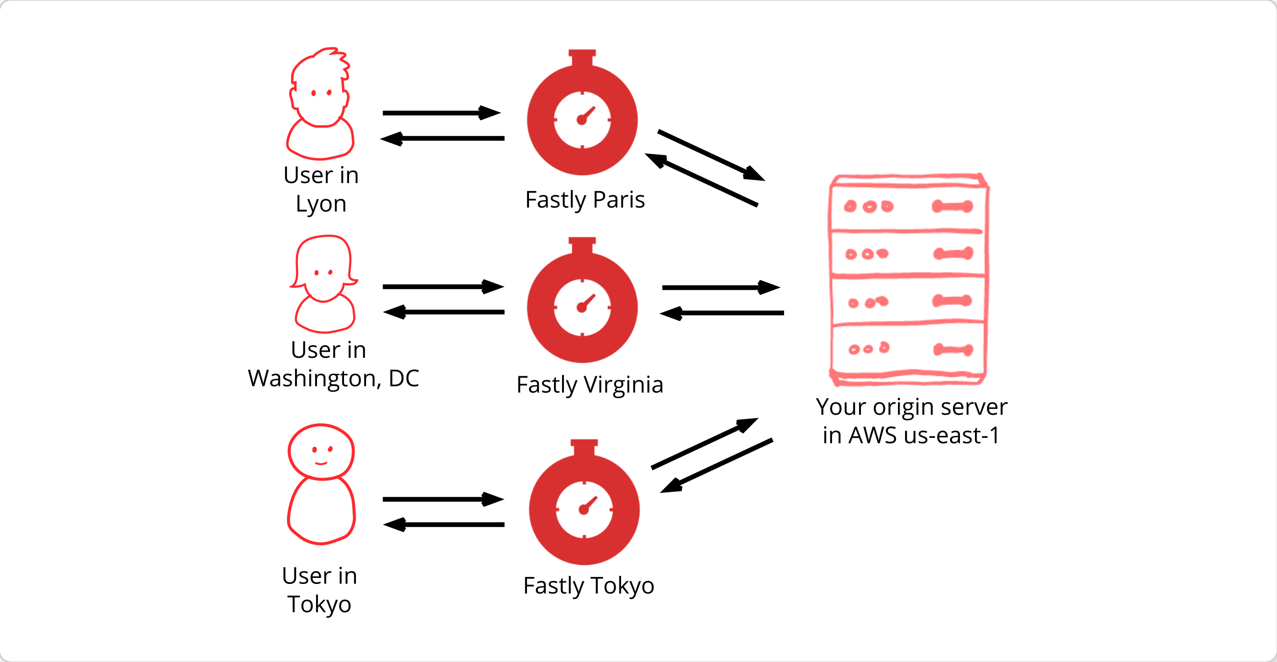
* * *

Shielding

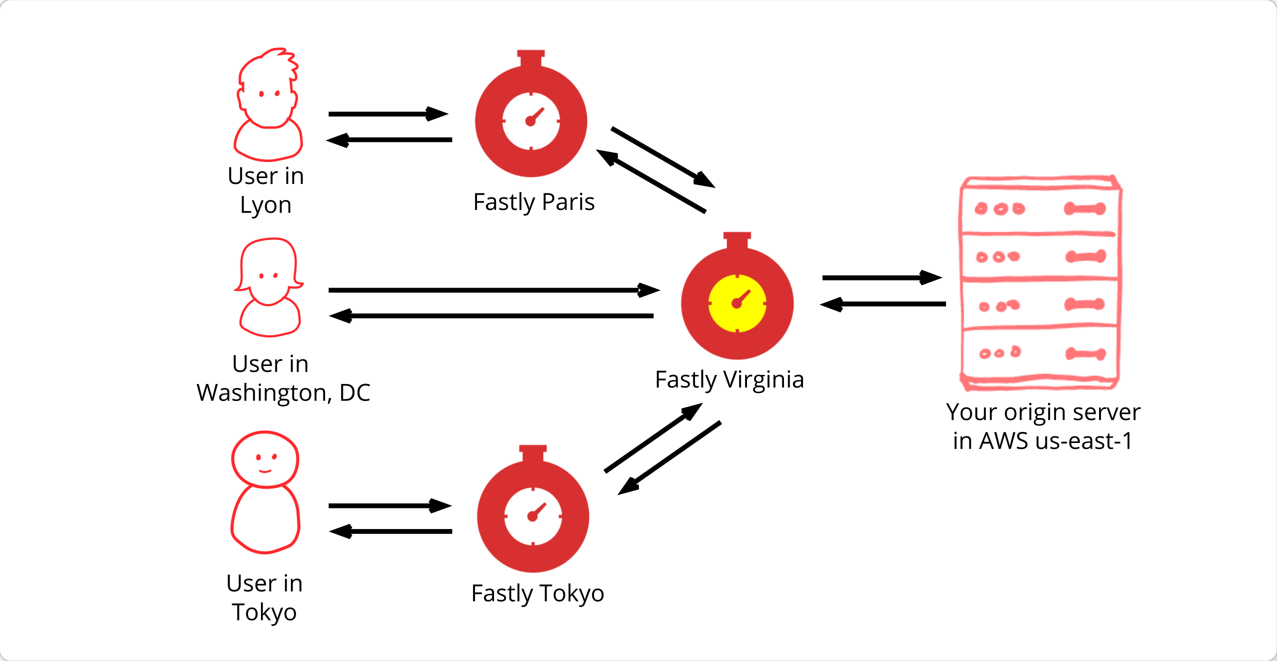
/en/guides/shielding

As a content delivery network, Fastly works by having any one of our global points of presence (POP) respond to requests that would otherwise be sent directly to your origin server. Because each POP acts independently, any time it doesn't have a cached version of the requested resource, it will make a request directly to your origin, even if another POP may be in the process of making the same request. You can reduce the load on your origin servers if you specify one of Fastly's POPs as an origin "shield." Designating a shield POP ensures requests to your origin will come from a single POP, thereby increasing the chances of an end user request resulting in a cache `HIT`.

For example, consider end users in three regions: Lyon (France), Washington DC (United States), and Tokyo (Japan). Each of these regions has a local Fastly POP available to them that caches information by default. If the user in Lyon requests a resource that is not cached on the Fastly Paris POP, however, that POP will make a request to the origin server, cache the resources in the response, and return the cached resource to the user.


User in Lyon
Fastly Paris
User in Washington, DC
Fastly Virginia
Your origin server in AWS us-east-1
User in Tokyo
Fastly Tokyo

With shielding enabled, however, the POP you designate collects all requests to your origin server instead. For example, the Fastly Virginia POP was designated as the shield for a server located in the AWS `us-east-1` region (as show in the illustration below).


User in Lyon
Fastly Paris
User in Washington, DC
Fastly Virginia
Your origin server in AWS us-east-1
User in Tokyo
Fastly Tokyo

In this scenario, if the user in Lyon requests a resource that isn't cached on the Fastly Paris POP, then that POP will forward the request to the shield POP, in this case in Virginia. If the resource had been previously cached on the shield POP, then it would be returned to the regional POP where it is then cached and returned to the user. Otherwise, the shield POP would make a request to the origin server for the resource, cache the response and return it to the regional POP where it is also cached before being returned to the user.

For more advanced use cases, see Advanced shielding scenarios.

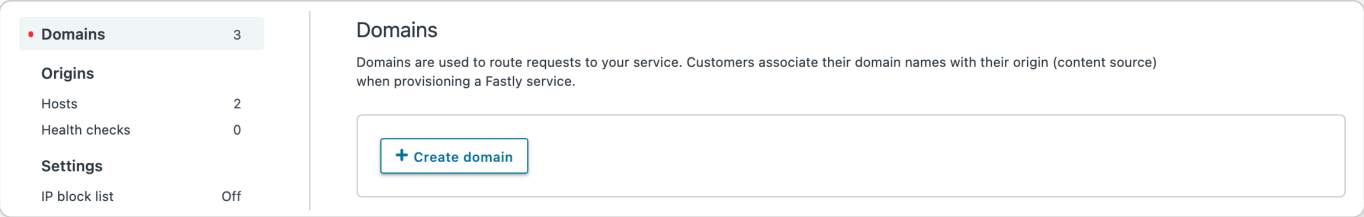
Enabling shielding

IMPORTANT

If you are using Google Cloud Storage as your origin, you need to follow the steps in our GCS setup guide instead of the steps below.

Enable shielding with these steps:

1. Read the caveats of shielding information below for details about the implications of and potential pitfalls involved with enabling shielding for your organization.
2. Log in to the Fastly web interface.
3. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
4. Click **Edit configuration** and then select the option to clone the active version.
5. Click **Origins**.
6. Click the name of the Host you want to edit.
7. From the **Shielding** menu, select the data center to use as your shield keeping the following in mind:
    - Generally, we recommend selecting a data center close to your backend. Doing this allows faster content delivery because we optimize requests between the shield POP you're selecting (the one close to your server) and the edge POP (the one close to the user making the request). Read our guidance on choosing a shield location for more information.
    - With multiple backends, each backend will have its own shield defined. This allows flexibility if your company has backends selected geographically and different shield POPs are desired.
8. Click **Update** to save your changes.
9. If you have changed the default Host or have added a header to change the Host, add the modified hostname to your list of domains. Do this by clicking **Domains** and checking to make sure the Host in question appears on the page. If it isn't included, add it by clicking **Create domain**.



Domains are used to route requests to your service. Customers associate their domain names with their origin (content source) when provisioning a Fastly service.

With shielding enabled, queries from other POPs appear as incoming requests to the shield. If the shield doesn't know about the modified hostname, it doesn't know which service to match the request to. Including the origin's hostname in the domain list eliminates this concern.

10. Click **Activate** to deploy your configuration changes.

Caveats of shielding

Shielding not only impacts traffic and hit ratios, it affects configuration and performance. When you configure shielding, be aware of the following caveats.

Inbound traffic billing

Inbound traffic to a shield will be billed as regular traffic, including requests to populate remote POPs. Enabling shielding will incur some additional Fastly bandwidth charges, but those are likely to be offset by savings of your origin bandwidth (and origin server load). Pass-through requests will not go directly to the origin, they will go through the shield first.

Global HIT ratio calculation

Global HIT ratio calculation may seem lower than the actual numbers. Shielding is not taken into account when calculating the global hit ratio. If an edge node doesn't have an object in its cache, it reports a miss. Local MISS/Shield HIT gets reported as a miss and a hit in the statistics, even though there is no call to the backend. It will also result in one request from the edge node to the shield. Local MISS/Shield MISS will result in two requests, because we will subsequently fetch the resource from your origin. For more information about caching with shielding, see our shielding developer documentation.

Shield failover

If a specified shield POP is inaccessible for a request (e.g., because of intervening network issues), that request will go directly from the edge node to your origin server, bypassing the shield.

Backends manually defined using VCL

You will be unable to manually define backends using VCL. Shielding at this level is completely dependent on backends being defined as actual objects through the web interface or API. Other custom VCL will work just fine.

Automatic load balancing

If you've selected auto load balancing, you can only select one shield total. You must use custom VCL to use multiple shields when auto load balancing is set.

Sticky load balancing

Enabling sticky load balancing and shielding at the same time requires custom VCL. Sticky load balancers use `client.identity` to choose where to send the session. The `client.identity` defaults to the IP request header. That's fine under normal circumstances, but if you enable shielding, the IP will be the original POP's IP, not the client's IP. Thus, to enable shielding and a custom sticky load balancer, you want to use the following:

```
if (req.http.fastly-ff) {
  set client.identity = req.http.Fastly-Client-IP;
}
```

Host header

You'll need to use caution when changing the Host header before it reaches the shield. Fastly matches a request with a Host header. If the Host header doesn't match to a domain within the service, an error of 500 is expected. To ensure consistent behavior of Fastly customer services and origins, we normalize the host header’s value to all lowercase in the `vcl_hash` subroutine. This means that no matter how your site’s domain name is capitalized in the request, the hash subroutine will behave predictably. This does not apply to any other parts of the URL, which remain case-sensitive.

Purging conflicts can occur if the Host header is changed to a domain that exists in a different service. For example, say Service A has hostname `a.example.com` and Service B has hostname `b.example.com`. If Service B changes the Host header to `a.example.com`, then the edge will think the request is for Service B but the shield will think the request is for Service A. As a precaution, you'll want to purge the object from both Service A and Service B to ensure that Fastly is retrieving the newest version on the next request. If you're changing the Host header before it reaches the shield, the object is split across both services because `a.example.com` and `b.example.com` are treated as separate objects. For information about the caveats to be aware of when you change a Host header, see our article on Specifying an override host.

VCL execution

VCL gets executed twice: once on the edge POP and again on the shield POP. Changes to `beresp` and `resp` can affect the caching of a URL on the shield and edge. Consider the following examples.

Say you want Fastly to cache an object for one hour (3600 seconds) and then ten seconds on the browser. The origin sends `Cache-Control: max-age=3600`. You unset `beresp.http.Cache-Control` and then reset `Cache-Control` to `max-age=10`. With shielding enabled, however, the result will not be what you expect. The object will have `max-age=3600` on the shield and reach the edge with `max-age=10`.

A better option in this instance would be to use `Surrogate-Control` and `Cache-Control` response headers. `Surrogate-Control` overrides `Cache-Control` and is stripped after the edge node. The `max-age` from `Cache-Control` will then communicate with the browser. The origin response headers would look like this:

```
Surrogate-Control: max-age=3600
Cache-Control: max-age=10
```

Another common pitfall involves sending the wrong `Vary` header to an edge POP. For example, there's VCL that takes a specific value from a cookie, puts it in a header, and that header is then added to the `Vary` header. To maximize compatibility with any caches outside of your control (such as with shared proxies as commonly seen in large enterprises), the `Vary` header is updated in `vcl_deliver`, replacing the custom header with `Cookie`. The code might look like this:

```
vcl_recv {
  # Set the custom header
  if (req.http.Cookie ~ "ABtesting=B") {
    set req.http.X-ABtesting = "B";
  } else {
    set req.http.X-ABtesting = "A";
  }
  ...
}

...

```

```
sub vcl_fetch {
  # Vary on the custom header
  if (beresp.http.Vary) {
    set beresp.http.Vary = beresp.http.Vary ", X-ABtesting";
  } else {
    set beresp.http.Vary = "X-ABtesting";
  }
  ...
}

...

sub vcl_deliver {
  # Hide the existence of the header from downstream
  if (resp.http.Vary) {
    set resp.http.Vary = regsub(resp.http.Vary, "X-ABtesting", "Cookie");
  }
}
```

When combined with shielding, however, the effect of the above code will be that edge POPs will have `Cookie` in the `Vary` header, and thus will have a terrible hit rate. To work around this, amend the above VCL so that `Vary` is only updated with `Cookie` when the request is not coming from another Fastly cache. The `Fastly-FF` header is a good way to tell. The code would look something like this (including the same `vcl_recv` from the above example):

```
# Same vcl_recv from above code example

sub vcl_fetch {
  # Vary on the custom header, don't add if shield POP already added
  if (beresp.http.Vary !~ "X-ABtesting") {
    if (beresp.http.Vary) {
      set beresp.http.Vary = beresp.http.Vary ", X-ABtesting";
    } else {
      set beresp.http.Vary = "X-ABtesting";
    }
  }
  ...
}

...

sub vcl_deliver {
  # Hide the existence of the header from downstream
  if (resp.http.Vary && !req.http.Fastly-FF) {
    set resp.http.Vary = regsub(resp.http.Vary, "X-ABtesting", "Cookie");
  }
}
```

POP maintenance

As part of our standard maintenance procedures, Fastly may perform maintenance on a POP that you have designated as an origin shielding location. When this happens, the POP will remain in service, but we may bring individual machines within that POP in or out of service as needed. Fastly may also decommission a POP. When this happens, Fastly will move your shielding location to a neighboring data center on your behalf. Before standard maintenance starts, we will post a notification update to our service status page, but if you prefer to be individually alerted before the maintenance happens, you can open a support ticket by visiting https://support.fastly.com/, letting us know your customer ID, and requesting to be notified of upcoming shielding migrations at locations where you shield. We will then contact you via email to announce upcoming migrations. If necessary, we'll ask you to perform certain actions at a time that's convenient for you.

* * *

Using Fastly's global POP network

🔗 /en/guides/using-fastlys-global-pop-network

Our points of presence (POPs) on the internet are strategically placed at the center of the highest density Internet Exchange Points around the world. Fastly's Network Map shows a detailed view of current and planned locations for all Fastly POPs. In addition, our data centers API endpoint provides a list of all Fastly POPs, including their latitude and longitude locations.

Geographic distribution is just one of the factors Fastly considers when building its global infrastructure. Other factors include connectivity, provider diversity, and our ability to build a scalable, performant modern network centered around internet infrastructure hubs to best support our customers' markets. Fastly's focus on automation, operational redundancy, and global delivery when building our infrastructure means our POPs often combine multiple physical sites to better serve densely populated markets.

Can Fastly host my content?

We accelerate your site by caching both static assets and dynamic content by acting as a reverse proxy to your origin server (also known as *Origin Pull*), but we do not provide services for uploading your content to our servers.

In addition to using your own servers as the source, we also support various *cloud storage* services as your origin, such as Amazon Simple Storage Service (S3), Google Cloud Storage (GCS), and Google Compute Engine (GCE) as your file origin. Our partnership with Google in particular enables us to have direct connectivity to their cloud infrastructure.

Directing your traffic through Fastly's global network

Fastly operates a domain name system (DNS) service specifically written to optimize getting your traffic to Fastly. This optimization automatically routes your traffic to the nearest Fastly POP (in terms of network proximity) on our global network and reroutes around internet outages and other disturbances.

To take advantage of Fastly's global network, use a fully qualified domain name (FQDN) when you create domains in the Fastly web interface and make sure you've properly configured your CNAME DNS records.

You can use an apex domain (e.g., `example.com` instead of `www.example.com`) as your canonical domain and still take advantage of Fastly's global network by pointing your apex DNS record at IPs on Fastly’s anycast network.

Limiting traffic to a subset of POPs

Fastly allows you to configure traffic routing choices that best suit your specific needs by prioritizing a subset of POPS through which your traffic travels. For example, you can specify that Fastly prioritizes the use of only North American and European Union (EU) POPs. For more information, read about our Billing zone anycast options.

Will Fastly ever adjust POP locations or service regions? How will I be notified?

Fastly continues to grow its network footprint, adding and combining new service POPs in the process. At times, expansion may result in the addition of new billable regions to our network. We'll announce new POP locations and new billable regions in advance through our network status page at fastlystatus.com. Contact sales@fastly.com with specific contract or billing questions.

* * *

Working with CDN services

/en/guides/working-with-cdn-services

Once you create a CDN service, you can edit and activate new versions of your Fastly services that include refinements and updates to your configurations. The web interface also allows you to do other things with existing Fastly services, like compare them to each other, deactivate or reactivate them, and delete them.

Before you begin

Before you create your first Fastly service, be sure you understand how services and versions work and service creation limits. You'll also need to understand the Fastly web interface controls before using them to work with your Fastly services.

TIP

This guide includes instructions for creating and interacting with CDN services. Our Compute documentation provides more information on working with services that take advantage of Compute.

Creating a new CDN service

You can create new CDN and Compute services through the web interface. Use the steps below to quickly create a CDN service that uses standard defaults common to many customers. If you prefer to create a draft service immediately, the **Skip and go to service configuration** link will automatically create a draft version of a new service with no recommended defaults.

1. Log in to the Fastly web interface.
2. Click **Create service** at the top right of the page, just below the primary navigation controls.
3. In the **Service name** field, enter a descriptive name for your service (e.g., `My Example Website Service`).

4. In the **Domain** field, enter the name of the domain you want Fastly to use to route requests to. Our guide to working with domains provides additional details.

5. In the **Host** field, enter the name of the host you plan to use as the backend or origin for your site. Our guide to working with hosts provides additional details.

6. Review the **Recommended settings** that are enabled by default for your service and decide whether or not to disable them by clicking the appropriate setting switch to **Off**.
   - The **Override default host** setting enables host override at the origin level. Many customers creating services specify an override host at the same time, especially when using an origin that requires a specific hostname to be passed to it.
   - The **Default compression** setting enables automatic compression using a default compression policy. Automatic compression on our edge servers can help you reduce the size of your assets so traffic can flow faster.
   - The **Force TLS & HSTS** setting enables the HTTP Strict Transport Security (HSTS) security enhancement specification through Fastly and forces modern web browsers to communicate only via the TLS protocol. If you leave this default setting enabled, be sure to create a TLS certificate for use with your domains. For more information, check out our TLS documentation.

7. Click **Activate**. Your service will be created using the service name, domain, and host you specify and the default settings you've left enabled.

8. Test your service configurations by opening `https://www.example.com.global.prod.fastly.net` in a new browser window, replacing `www.example.com` with your own website's domain name. Your website should appear, though it may take up to 60 seconds for new configuration settings to take effect. You can continue to explore various configuration settings for as long as you like before starting to serve traffic.

9. Once you've reviewed everything and are ready, complete your service setup and start serving traffic through Fastly by setting your domain's CNAME DNS record to point to Fastly.

Editing your services

You might want to edit a version of an existing service to do things like:

- change the amount of time information is retained in cache memory for a service
- configure a service to temporarily serve stale content should your origin server need to be unavailable for an extended period of time (for example, taken offline for maintenance)
- decrease the amount of time Fastly will wait for your origin server to respond to a request for content

Editing and activating versions of services

Fastly locks versions of services you've already activated to make rollbacks safer and provide version control. You can duplicate (*clone*) any existing service version, active or inactive, and edit that cloned version. You must *activate* new versions of services to deploy their configurations. Configuration changes are never automatically activated.

To make changes to a service and activate a new version, follow the steps below:

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

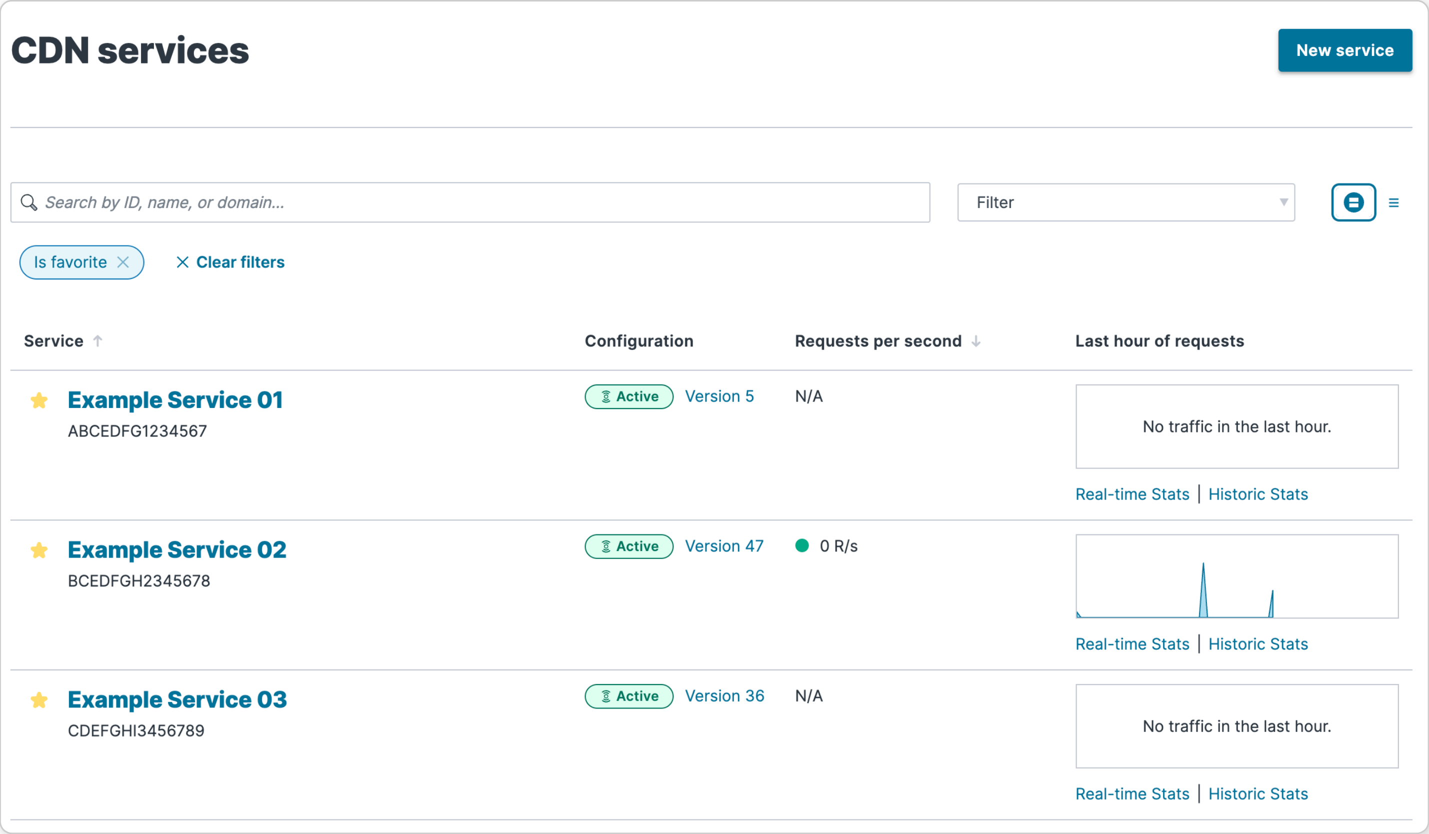
3. From the **Edit configuration** menu, select the appropriate service configuration action:
    - Select **Clone version *[version number]* (active)** to clone the active version of the service for editing.
    - Select **Edit version *[version number]* (latest draft)** to edit the latest draft of the service.
4. Click **Activate**. The new version of the service is activated and appears in the event log.

Viewing all services

You can view all the services associated with your account or filter them to show only specific types of services.

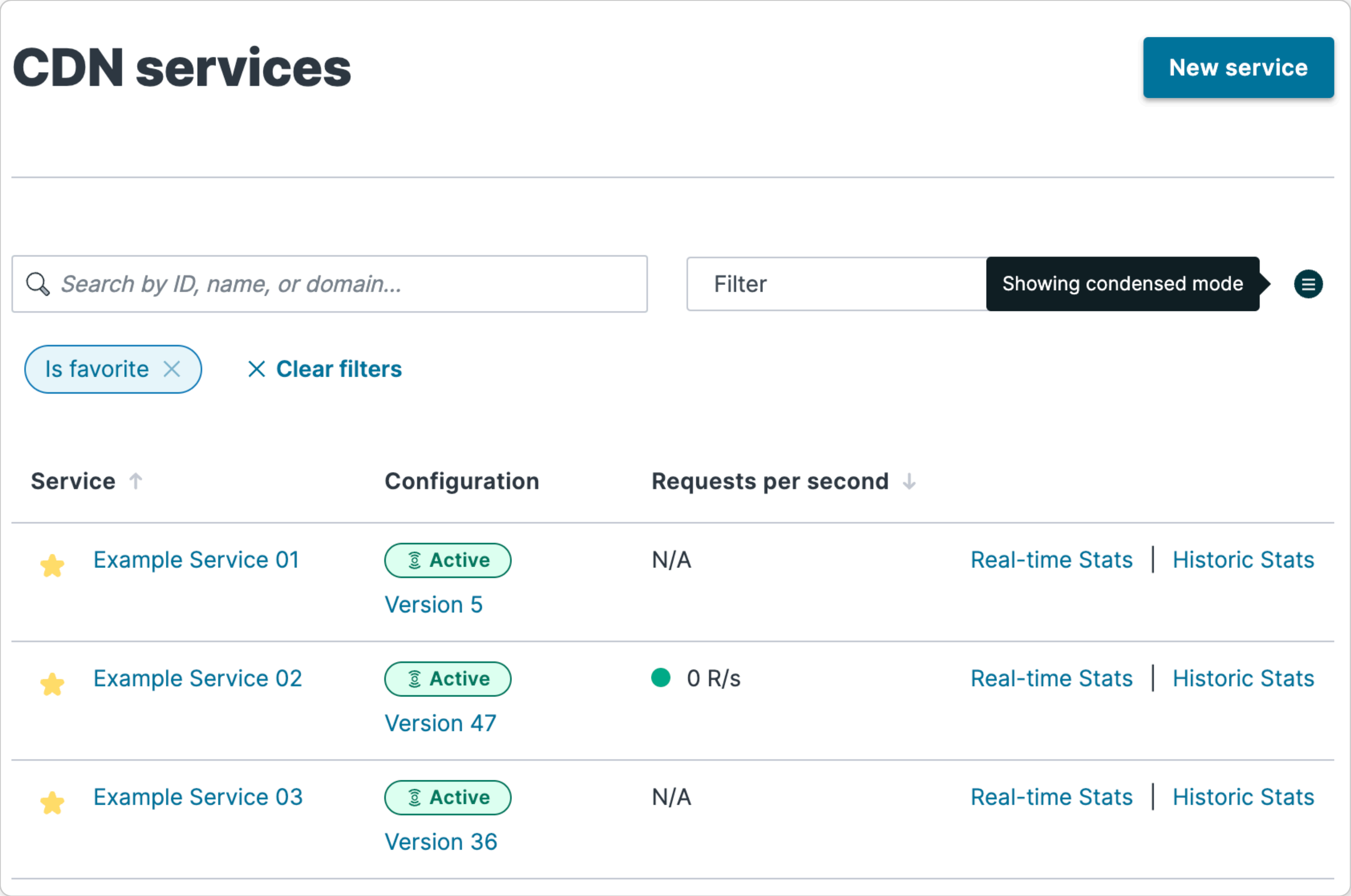
Viewing a complete list of services

To view all your services once you're logged in, click either **Compute services** or **CDN services** in the **Your services** card of the **Home** page to display a list of your services of a particular type. Clicking on each column in the displayed list will sort the entire list based on that content type.


CDN services
New service
Search by ID, name, or domain...
Filter
Is favorite
Clear filters
Real-time Stats | Historic Stats

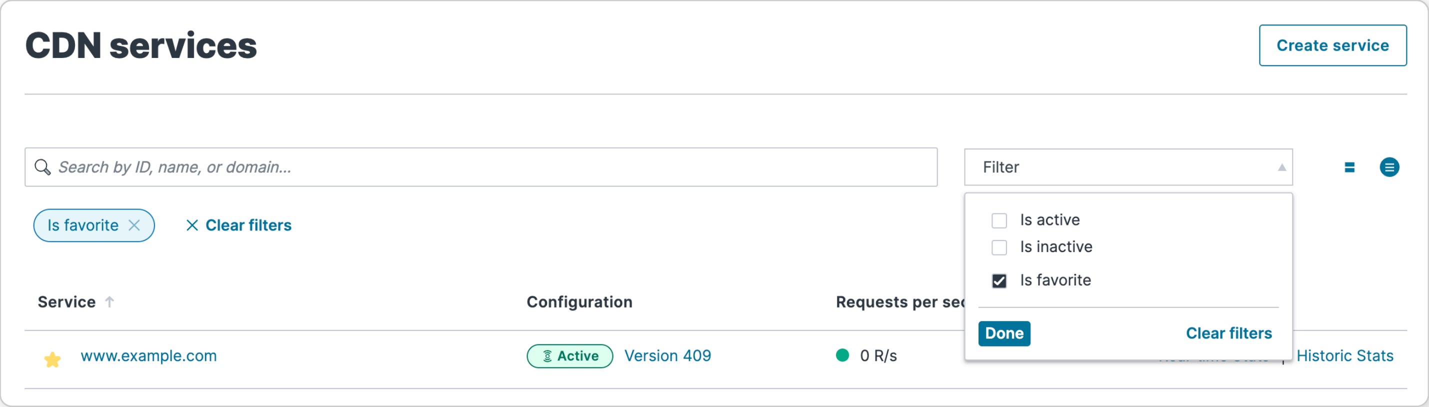
Service	Configuration	Requests per second	Last hour of requests
Example Service 01 ABCEDFG1234567	Active Version 5	N/A	No traffic in the last hour.
Example Service 02 BCEDFGH2345678	Active Version 47	● 0 R/s	
Example Service 03 CDEFGHI3456789	Active Version 36	N/A	No traffic in the last hour.

If you have a lot of services, you can view a condensed version of the list by clicking the icon with three lines above the list of services.


CDN services
New service
Search by ID, name, or domain...
Filter
Showing condensed mode
Is favorite
Clear filters
Service
Configuration
Requests per second
Example Service 01
Active
Version 5
N/A
Real-time Stats | Historic Stats
Example Service 02
Active
Version 47
0 R/s
Real-time Stats | Historic Stats
Example Service 03
Active
Version 36
N/A
Real-time Stats | Historic Stats

Viewing a subset of all services

You can filter your list of services to show services that are active or inactive, or that you've designated as favorites. To filter your services, select the appropriate filters from the **Filter** menu above the list of services:


CDN services
Create service
Search by ID, name, or domain...
Filter
☐ Is active
☐ Is inactive
☑ Is favorite
Done
Clear filters
Is favorite
Clear filters
Service
Configuration
Requests per sec
www.example.com
Active
Version 409
0 R/s
Historic Stats

The filters you've applied to the list will appear as oval shapes above the list of services. You can remove a filter at any time by clicking the `x` on any filter or the **Clear filters** option.

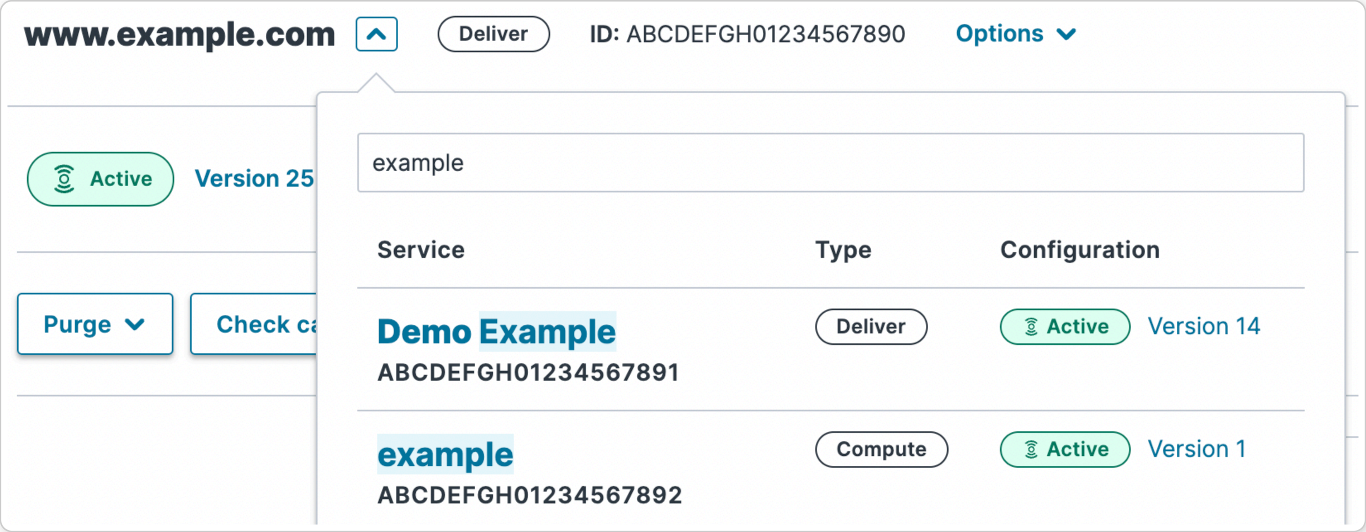
TIP

If you have a lot of services, consider clicking the star icon ⭐ to the left of a service name to mark it as a favorite. You can view a subset of services using filters and include only your favorites.

Switching between services and service versions

To switch between services associated with your account, follow these steps:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Service summary** or **Service configuration** page, click the switcher to the right of the account name and ID.


www.example.com
Deliver
ID: ABCDEFGH01234567890
Options
Active
Version 25
Purge
Check ca

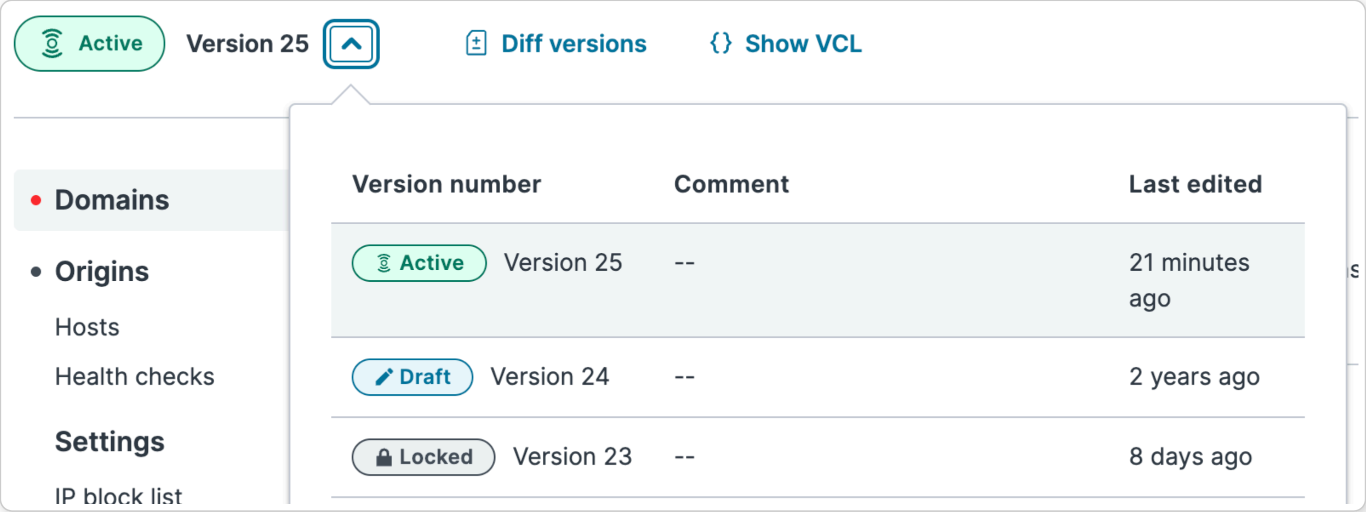
example

Service	Type	Configuration
Demo Example ABCDEFGH01234567891	Deliver	Active Version 14
example ABCDEFGH01234567892	Compute	Active Version 1

4. Select the appropriate service from the list that appears.

To switch between versions of a specific service, follow these steps:

1. From the **Service configuration** page, click the switcher to the right of the version number.


Active
Version 25
Diff versions
Show VCL
Domains
Origins
Hosts
Health checks
Settings
IP block list

Version number	Comment	Last edited
Active Version 25	--	21 minutes ago
Draft Version 24	--	2 years ago
Locked Version 23	--	8 days ago

2. Select the appropriate version from the list that appears.

Deleting a service

Fastly allows you to delete any service you create, along with all of its versions. Fastly does not offer a way to delete specific versions of a service, however. Service versions are meant to be a historic log of the changes that were made to a service. To undo changes introduced by a particular service version, you can always go back to a previous version and reactivate or clone a new service version based on any old version.

To delete any service along with all of its versions, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Deactivate**.
4. Click **Confirm and deactivate** to confirm you want to deactivate your service and acknowledge that you no longer want to serve traffic with it.
5. From the **Options** menu, select **Delete** and then click **Confirm and delete** to delete the service.

> IMPORTANT
>
> To minimize the risk of unauthorized use of your domains, we strongly recommend modifying or deleting any DNS CNAME records pointing to the Fastly hostname associated with the deleted service. Follow the instructions on your DNS provider's website.

Other things you can do

In addition to creating or editing services, you can view all your services, rename them, compare versions of them, deactivate or reactivate specific versions of them, and delete them.

Renaming services

To rename your service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Edit service name**.
4. In the **Service name** field, enter a new service name.
5. Click **Apply**. The newly renamed service name appears.

Adding comments to service versions

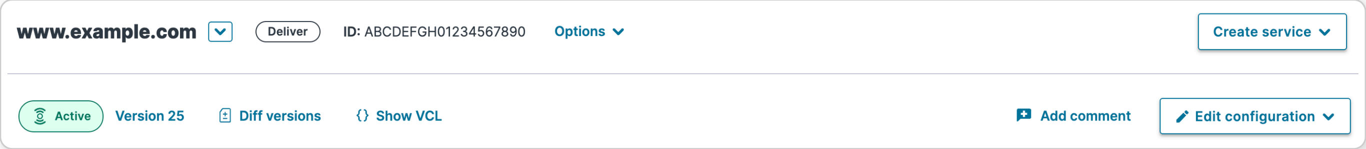
Service versions can include comments to label them (e.g., to identify reasons for changes in that version). You can add and update version comments on both locked and activated service versions.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by

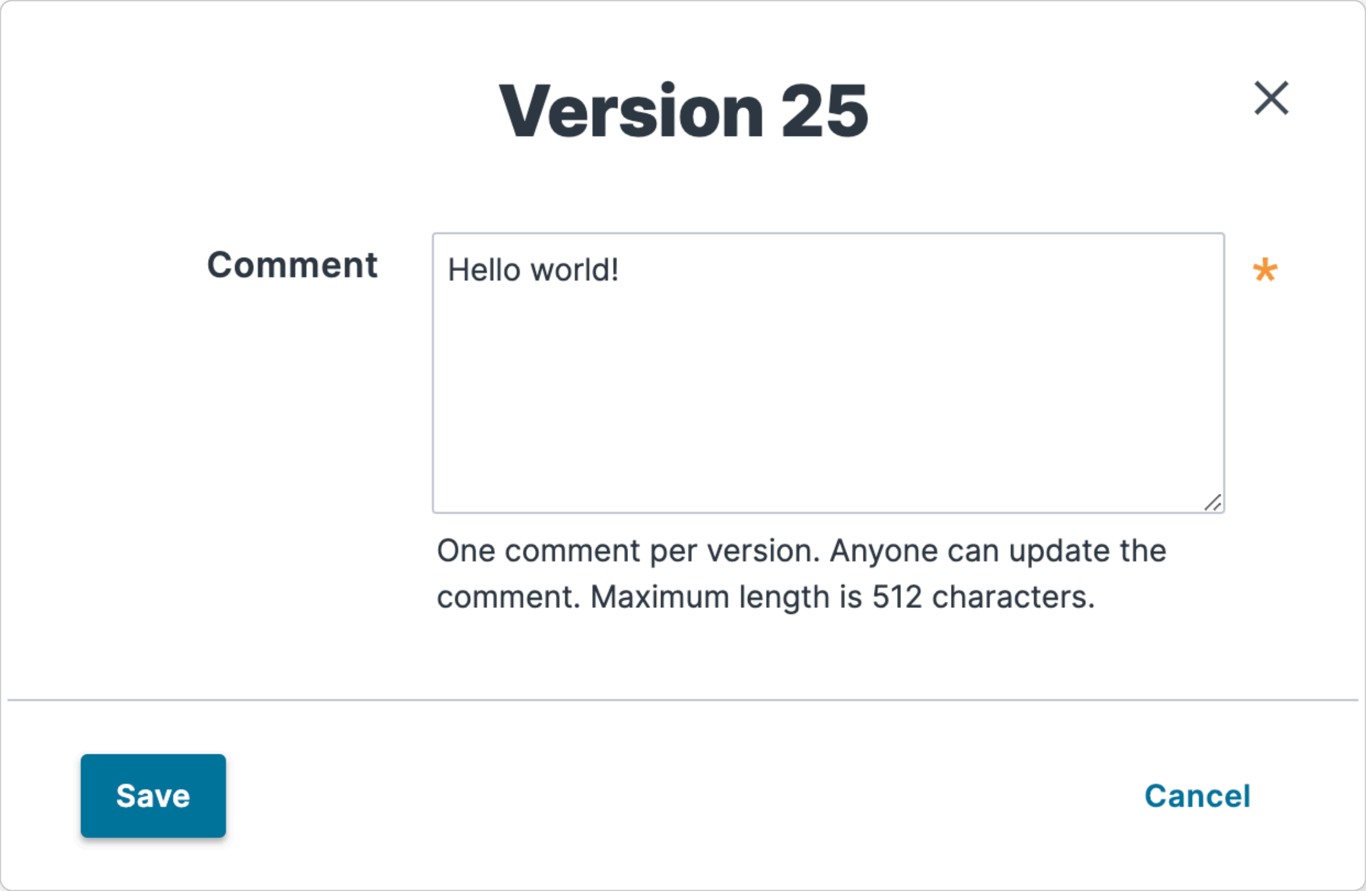
type.

3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Add comment** in the upper right corner of the web interface.


www.example.com
Deliver
ID: ABCDEFGH01234567890
Options
Create service
Active
Version 25
Diff versions
Show VCL
Add comment
Edit configuration

The comment window appears.

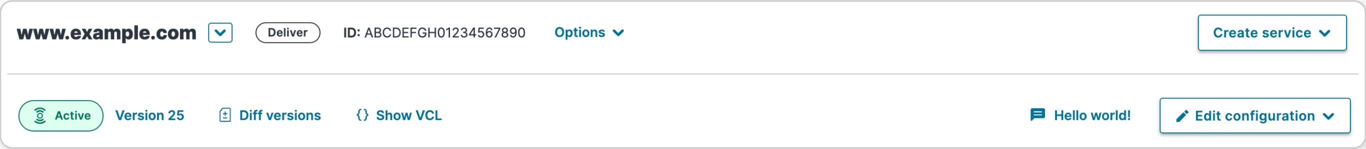

Version 25
Save
Cancel

Comment: Hello world!

One comment per version. Anyone can update the comment. Maximum length is 512 characters.

5. In the **Comment** field, enter a meaningful comment for the version.

6. Click **Save**. The truncated version of the comment appears where the Add comment link used to be.


www.example.com
Deliver
ID: ABCDEFGH01234567890
Options
Create service
Active
Version 25
Diff versions
Show VCL
Hello world!
Edit configuration

TIP

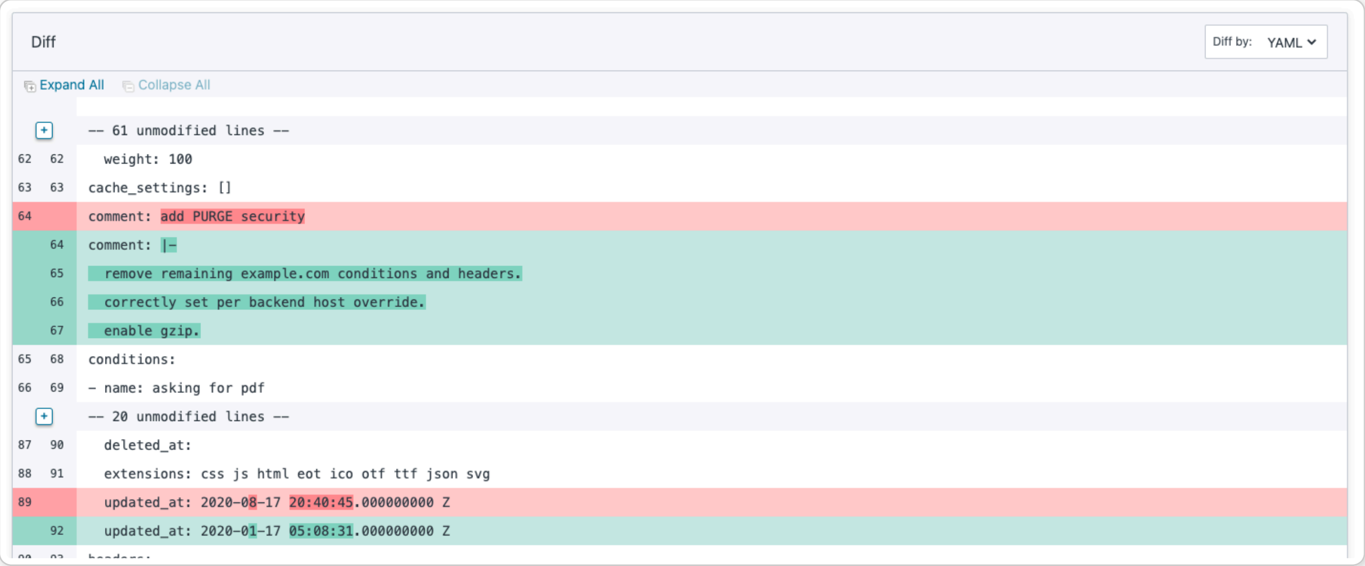
You can view service version comments at any time by clicking the service version number to display the version selection menu or by clicking the version comment icon to display the version comment in a separate window. Version comments also appear in the event log to help with account activity monitoring.

Comparing different service versions

To compare two versions of a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Diff versions**. The Diff versions page appears.

   Removals are highlighted in red and the additions and changes are highlighted in green. Any large blocks of unaffected configuration lines can be expanded and viewed or collapsed and hidden by clicking on the plus (+) sign to the left of the actual changes, next to the line numbers.



```
      -- 61 unmodified lines --
62 62   weight: 100
63 63 cache_settings: []
64    comment: add PURGE security
   64 comment: |-
   65   remove remaining example.com conditions and headers.
   66   correctly set per backend host override.
   67   enable gzip.
65 68 conditions:
66 69 - name: asking for pdf
      -- 20 unmodified lines --
87 90   deleted_at:
88 91   extensions: css js html eot ico otf ttf json svg
89      updated_at: 2020-08-17 20:40:45.000000000 Z
   92   updated_at: 2020-01-17 05:08:31.000000000 Z
```

> TIP
>
> You can change the compared service versions by clicking **Switch versions** and selecting a different version number in the menu that appears.

Deactivating a service

To deactivate a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Deactivate**.
4. In the **Enter service name** field, enter the exact service name to deactivate.
5. Click **Confirm and deactivate** to confirm you want to deactivate your service and acknowledge that you no longer want to serve traffic with it.

> IMPORTANT

To minimize the risk of unauthorized use of your domains, we strongly recommend modifying or deleting any DNS CNAME records pointing to the Fastly hostname associated with the deactivated service. Follow the instructions on your DNS provider's website.

You can also activate or deactivate a service via the API. Did you accidentally delete a service? We can help.

Reactivating a service

To reactivate a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Activate**. The service is reactivated.
5. If you removed the DNS CNAME records for the service's domains when you deactivated the service, you should add new DNS CNAME records now.

Getting help with accidental service deletions

Services can be deactivated or deleted. Deactivated services can be reactivated at any time, but once they've been deleted you must contact Customer Support to have them restored. When sending your request, remember to include:

- your customer ID
- your company name
- your service ID (the name of the service you want restored)

Customer Support will notify you when your service has been restored.

What's next

Learn more about working with domains, hosts, and health checks as you continue to refine versions of your service configurations.

* * *

Working with CNAME records and your DNS provider

/en/guides/working-with-cname-records-and-your-dns-provider

To route traffic to a Fastly service using your domain name, you must associate the domain with your Fastly service and update your domain's CNAME record to point to Fastly. This guide helps you update your domain's CNAME record with your DNS provider and describes how to choose the right hostname.

IMPORTANT

Fastly is not a DNS provider. The steps you follow when creating a CNAME record for your domain will vary depending on your DNS provider. You must have access privileges to modify DNS records for your domain.

Before you begin

Keep in mind the following CNAME configuration limitations. Specifically:

- **Shared TLS hostname incompatibility.** You can't add a CNAME record for a shared TLS hostname.
- **Apex domain incompatibility.** You can't use a CNAME record if you plan to use Fastly on your apex domain (e.g., `example.com` rather than `www.example.com`). Check out our guide to using Fastly with apex domains for more details.

Choosing the right Fastly hostname for your CNAME record

To successfully update your DNS CNAME record, you must choose the right Fastly hostname to use. Choosing the appropriate CNAME record is the final step required before Fastly can start acting as a reverse proxy and begin routing client traffic through Fastly services.

The hostname you choose will differ based on:

- the standard HTTPS (TLS) support requirements for your domain, including whether or not HTTP/2 is enabled.
- any custom TLS options purchased for your domain.
- whether or not you choose to limit your traffic to the North American and EU network or use Fastly's global network.

We've provided recommendations below based on these criteria.

Non-TLS hostnames and limiting traffic

If you don't require TLS support and only need to accept HTTP (Port 80) connections, use one of the following hostnames:

- Use `dualstack.nonssl.global.fastly.net.` to route traffic through Fastly's entire global network.
- Use `dualstack.nonssl.us-eu.fastly.net.` to route traffic through Fastly's North American and EU POPs only.

IMPORTANT

Fastly's non-TLS hostnames refuse HTTPS connections (port 443) to prevent TLS certificate mismatch errors.

TLS-enabled hostnames

If using Fastly TLS, use `<letter>.sni.global.fastly.net` to route traffic through Fastly's entire global network. The `letter` value depends on your TLS configuration:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.

3. Click **View details** for the domain you would like to route to Fastly. The domain's details page appears.

The CNAME records section contains the value for your CNAME record. You can prefix the value with `dualstack` to enable IPv6 support (e.g., `dualstack.<letter>.sni.global.fastly.net`).

IMPORTANT

You must use the assigned Fastly TLS hostname provided in the Fastly web interface. Using the incorrect Fastly hostname will cause a TLS Certificate mismatch error for HTTPS (Port 443) traffic.

Updating the CNAME record with your DNS provider

Once you've determined the appropriate Fastly hostname for your domain, the next step is to create a CNAME record for your domain. Refer to your DNS provider's documentation for exact instructions on how to create or update a CNAME record. If you run your own DNS server or are familiar with the format of BIND zone files, the CNAME record would look similar to this:

```
www.example.com.    3600    IN    CNAME   nonssl.global.fastly.net.
```

In the above example, the domain set up on Fastly is `www.example.com.`, with a time-to-live (TTL) of `3600` seconds (1 hour), the Record Type is `CNAME`, and the Fastly hostname is `nonssl.global.fastly.net.` because TLS support isn't required and traffic will be routed through Fastly's entire global network.

Best practices when updating a DNS CNAME record

- Be sure you've added all domains you want served by Fastly to the appropriate service. If you don't and you point your domain to Fastly, an `unknown domain` error will occur.
- Make sure your service is properly configured. You can test a Fastly service on your local machine by using curl and testing setup before changing domains.
- If you have multiple hostnames on the same domain (e.g., `api.example.com`, `www.example.com`, `app.example.com`), you can use a DNS wildcard record (`*.example.com`) at your DNS provider so only a single CNAME record is created and maintained. You should also add either a matching `*.example.com` domain or the individual domains to your Fastly service.
- Before changing a CNAME to point to a Fastly hostname, change your service configuration to lower the CNAME's TTL to a small number (we suggest 60 seconds) and wait for the old TTL to expire. Creating a DNS CNAME record for your domain after the TTL expiration ensures you have an easy way to roll back changes if you encounter an issue. Once you confirm everything is working properly using Fastly, you can increase the TTL to its original value.

Checking your CNAME record

To check your CNAME record, run the following command in a terminal window:

```
$ dig www.example.com +short
```

Your output should appear similar to the following:

```
nonssl.global.fastly.net.
```

```
151.101.117.57
```

In most cases, the hostname displayed first will be your current Fastly hostname (in this case, `nonssl.global.fastly.net.`). If you don't see a Fastly hostname in the output or if you see an incorrect Fastly hostname, then either your CNAME isn't properly set at your DNS provider or an older CNAME record is still cached by your local DNS resolver.

You can use various online DNS query tools like OpenDNS Cache Check or whatsmydns.net to test the current DNS responses from the different DNS resolvers worldwide.

Removing CNAME records

If you deactivate a service, delete a service, or cancel your account, we strongly recommend modifying or deleting any CNAME records pointing to Fastly hostnames. Follow the instructions on your DNS provider's website. Doing so will minimize the risk of unauthorized use of your domains.

* * *

Working with Compute services

/en/guides/working-with-compute-services

Once you create a Compute service, you can edit and activate new versions of your Fastly services that include refinements and updates to your configurations. The web interface also allows you to do other things with existing Fastly services, like compare them to each other, deactivate or reactivate them, and delete them.

Before you begin

Before you create your first Fastly service, be sure you understand how services and versions work and service creation limits. You'll also need to understand the Fastly web interface controls before using them to work with your Fastly services.

Some steps require using the Fastly CLI. If you haven't already, follow the steps for creating an API token, making sure it has `global` scope. Then, download and install the Fastly CLI and use that token to authenticate your account before continuing.

Creating a new Compute service

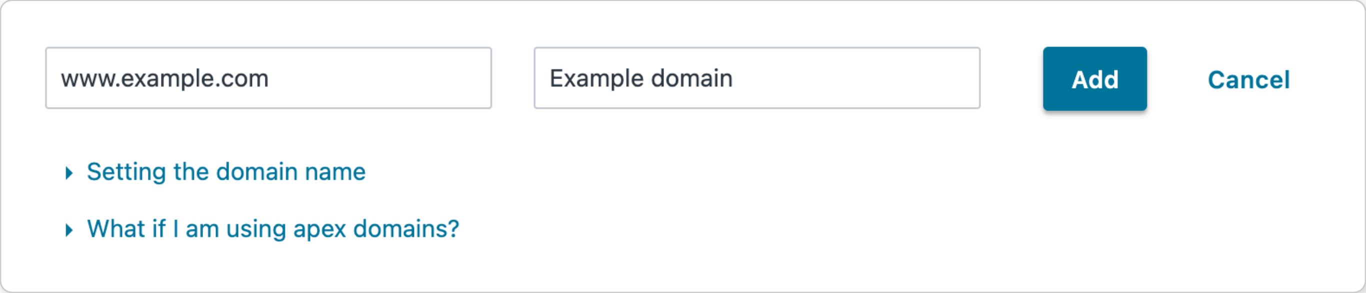
To create a new Compute service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Create service** menu, select **Compute** to create a new Wasm-based service. A new, unnamed Wasm service's configuration page appears.
3. Rename the service as necessary.
4. *(Optional)* Add a comment to help you identify what you're working on.

TIP

After creating your Compute service, you can choose to complete the remaining configuration steps via the Fastly CLI.

5. Use the **Domains** field to add a domain to the service.



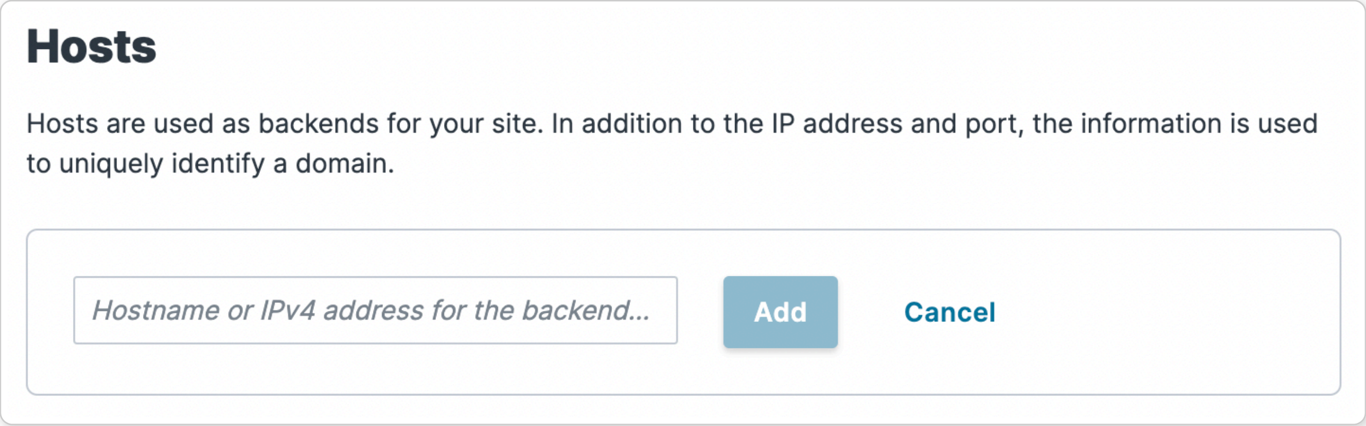
www.example.com

Example domain

NOTE

Compute requires you to secure your domain using TLS.

6. Use the **Hosts** field to add a host to the service.



Hosts

Hosts are used as backends for your site. In addition to the IP address and port, the information is used to uniquely identify a domain.

Hostname or IPv4 address for the backend...

TIP

Many customers setting up services specify an override host at the same time. See our guidance on overriding a host at the origin level if you're interested in exploring this functionality.

7. Using the Fastly CLI, create a new Compute project. A local development environment and a `fastly.toml` package manifest will be generated.
8. Open `fastly.toml` and add your service ID into the `service_id` property.
9. Compile the project into a Wasm binary packaged for Fastly use. The package will be named `<name>.tar.gz`.
10. Return to your service in the Fastly web interface.
11. Click **Package**.
12. Click **Browse for Package** to navigate to the package file on your system.
13. Test your service configurations by opening `http://www.example.com.global.prod.fastly.net` in a new browser window, replacing `www.example.com` with your own website's domain name. Your website should

appear, though it may take up to 60 seconds for new configuration settings to take effect.

> ✓ TIP
>
> You can continue to explore various configuration settings for as long as you like before starting to serve traffic.

14. Click **Activate** at the top right of the screen and then **Confirm and Activate** to activate your new service. The Compute page appears with details about the configuration settings you've applied.
15. Once you're ready, complete your service setup and start serving traffic through Fastly by setting your domain's CNAME DNS record to point to Fastly.

Editing your services

You might want to edit a version of an existing service to do things like:

- change the amount of time information is retained in cache memory for a service
- configure a service to temporarily serve stale content should your origin server need to be unavailable for an extended period of time (for example, taken offline for maintenance)
- decrease the amount of time Fastly will wait for your origin server to respond to a request for content

Editing and activating versions of services

Fastly locks versions of services you've already activated to make rollbacks safer and provide version control. You can duplicate (*clone*) any existing service version, active or inactive, and edit that cloned version. You must *activate* new versions of services to deploy their configurations. Configuration changes are never automatically activated.

To make changes to a service and activate a new version, follow the steps below:

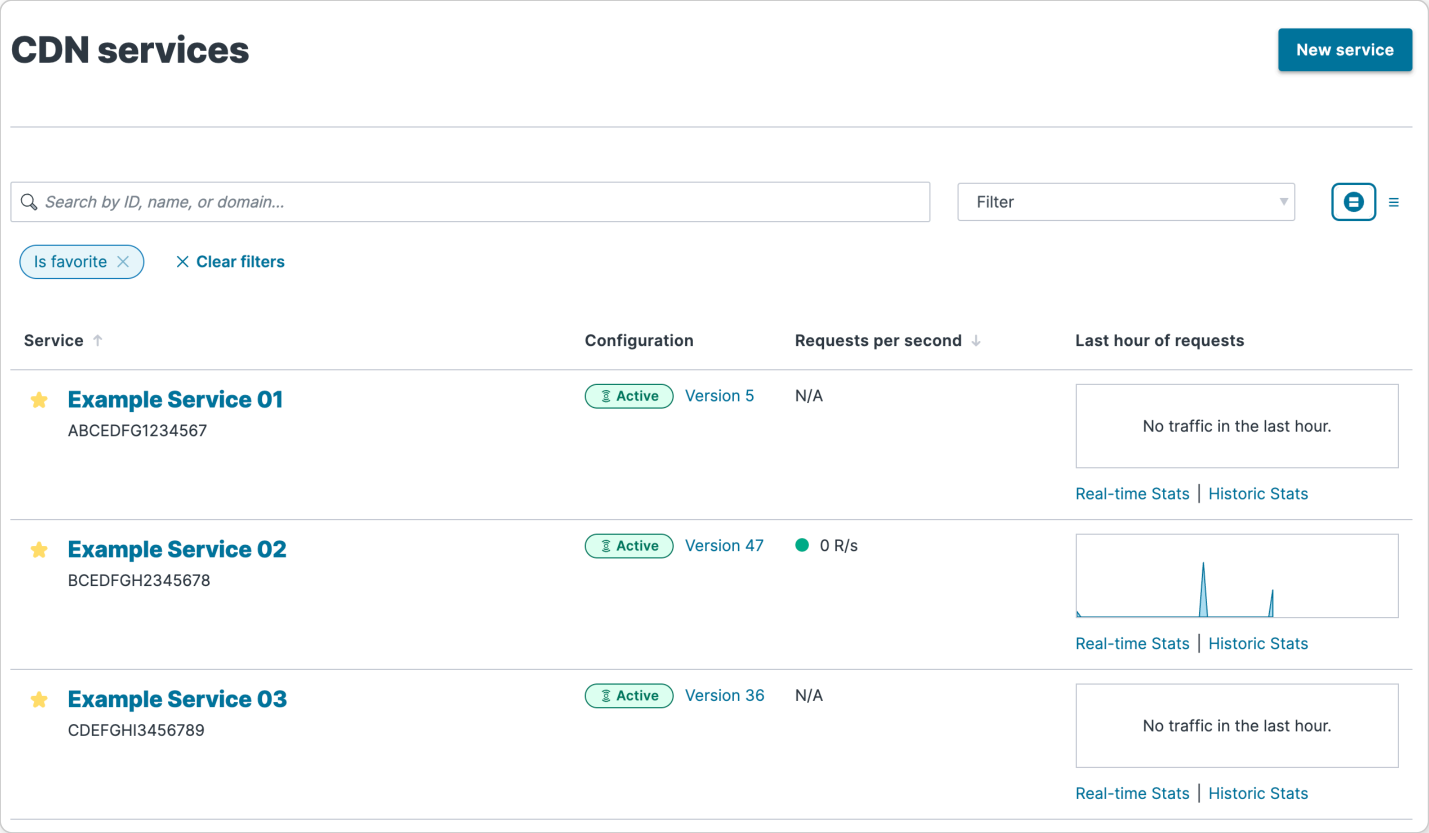
1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Edit configuration** menu, select the appropriate service configuration action:
    - Select **Clone version *[version number]* (active)** to clone the active version of the service for editing.
    - Select **Edit version *[version number]* (latest draft)** to edit the latest draft of the service.
4. Click **Activate**. The new version of the service is activated and appears in the event log.

Viewing all services

You can view all the services associated with your account or filter them to show only specific types of services.

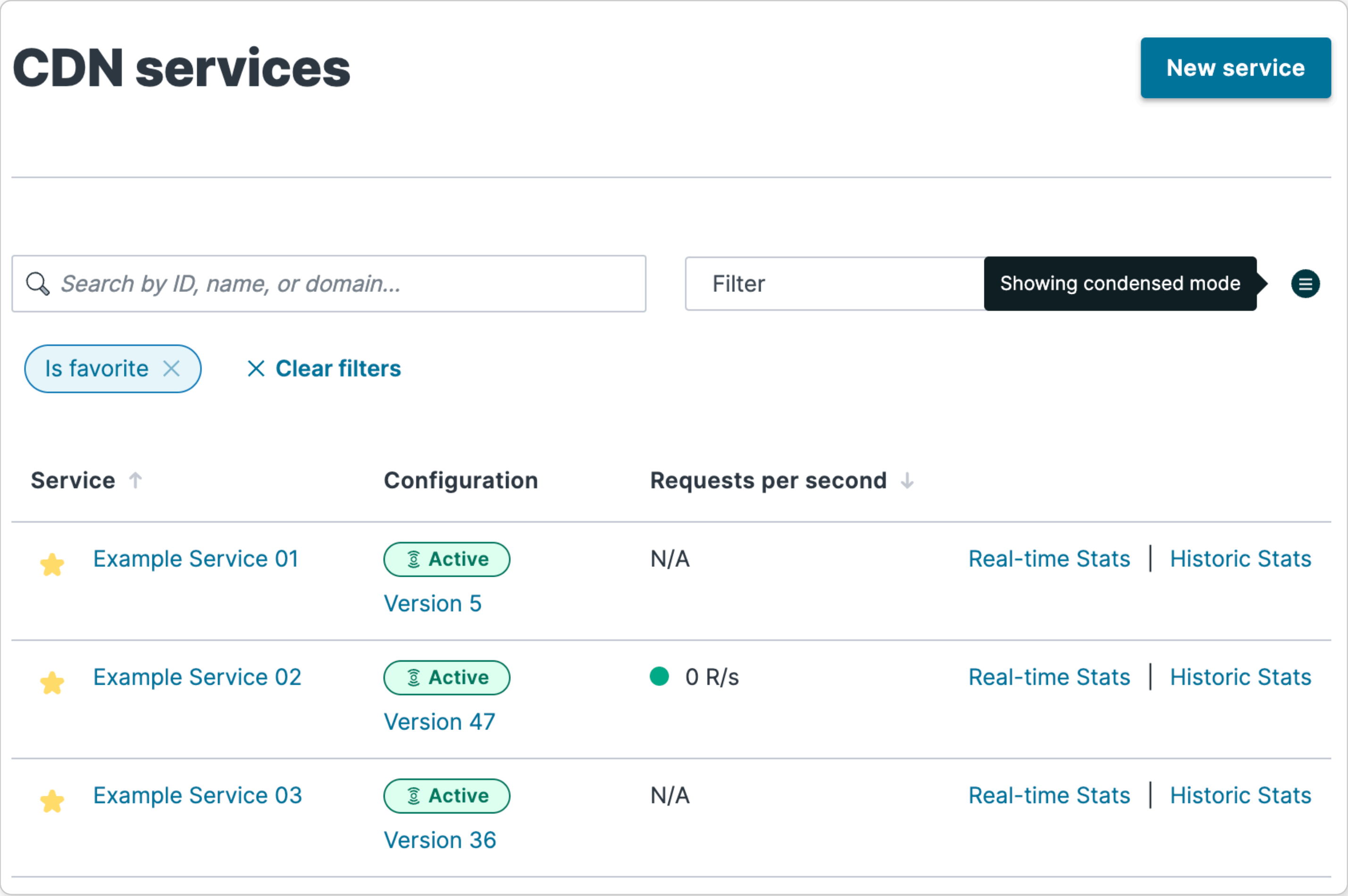
Viewing a complete list of services

To view all your services once you're logged in, click either **Compute services** or **CDN services** in the **Your services** card of the **Home** page to display a list of your services of a particular type. Clicking on each column in the displayed list will sort the entire list based on that content type.


CDN services
New service
Search by ID, name, or domain...
Filter
Is favorite
Clear filters

Service	Configuration	Requests per second	Last hour of requests
★ Example Service 01 ABCEDFG1234567	Active Version 5	N/A	No traffic in the last hour. Real-time Stats | Historic Stats
★ Example Service 02 BCEDFGH2345678	Active Version 47	● 0 R/s	Real-time Stats | Historic Stats
★ Example Service 03 CDEFGHI3456789	Active Version 36	N/A	No traffic in the last hour. Real-time Stats | Historic Stats

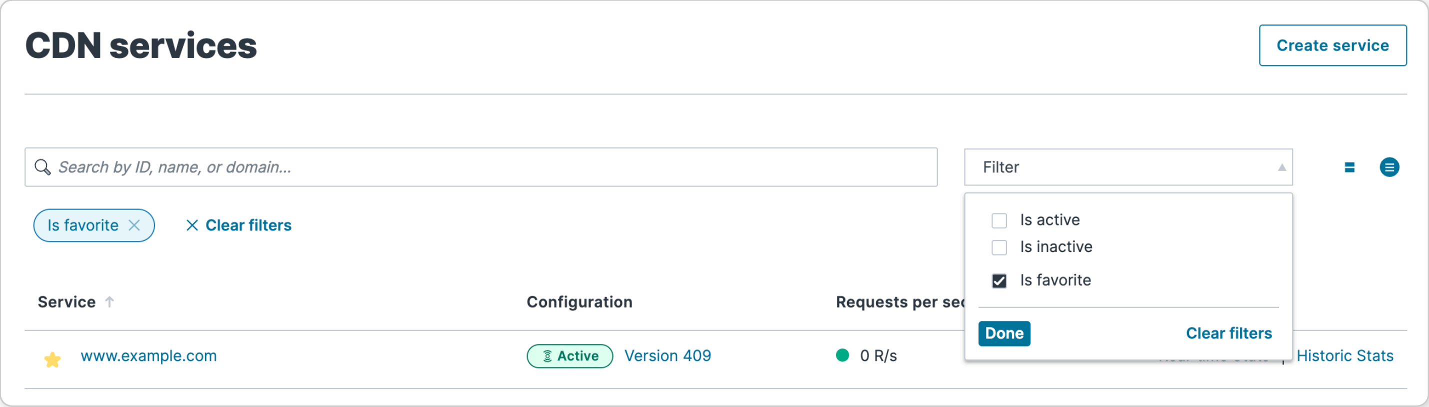
If you have a lot of services, you can view a condensed version of the list by clicking the icon with three lines above the list of services.


CDN services
New service
Search by ID, name, or domain...
Filter
Showing condensed mode
Is favorite
Clear filters

Service	Configuration	Requests per second	
★ Example Service 01	Active Version 5	N/A	Real-time Stats | Historic Stats
★ Example Service 02	Active Version 47	● 0 R/s	Real-time Stats | Historic Stats
★ Example Service 03	Active Version 36	N/A	Real-time Stats | Historic Stats

Viewing a subset of all services

You can filter your list of services to show services that are active or inactive, or that you've designated as favorites. To filter your services, select the appropriate filters from the **Filter** menu above the list of services:


CDN services
Create service
Search by ID, name, or domain...
Filter
☐ Is active
☐ Is inactive
☑ Is favorite
Done
Clear filters
Is favorite ✕
✕ Clear filters
Historic Stats

Service	Configuration	Requests per sec
www.example.com	Active Version 409	0 R/s

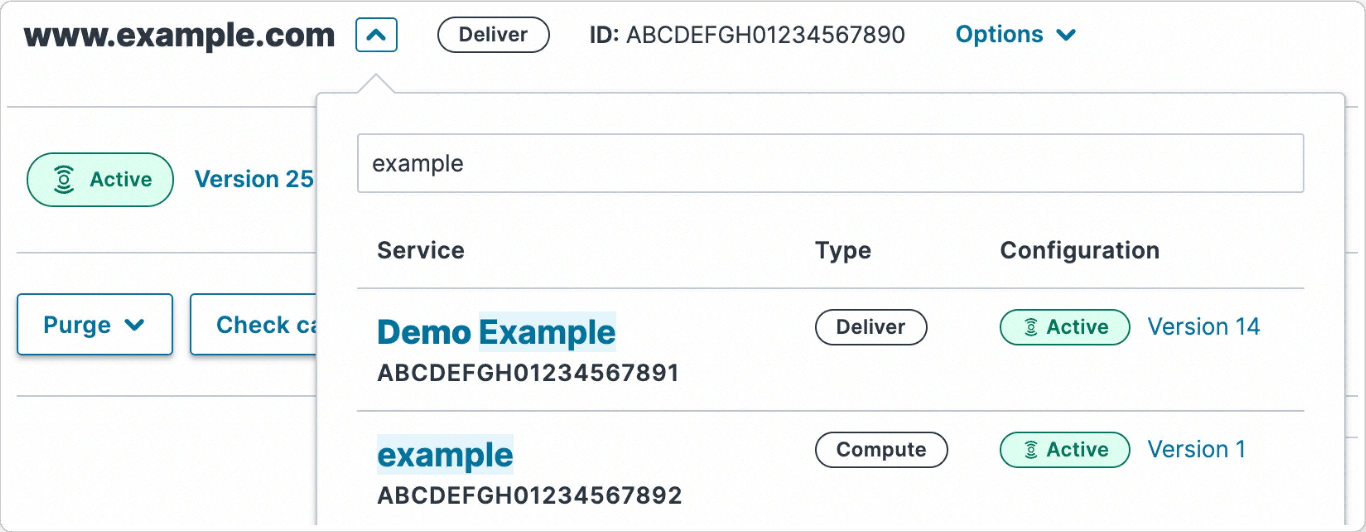
The filters you've applied to the list will appear as oval shapes above the list of services. You can remove a filter at any time by clicking the `x` on any filter or the **Clear filters** option.

> TIP
>
> If you have a lot of services, consider clicking the star icon ⭐ to the left of a service name to mark it as a favorite. You can view a subset of services using filters and include only your favorites.

Switching between services and service versions

To switch between services associated with your account, follow these steps:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Service summary** or **Service configuration** page, click the switcher to the right of the account name and ID.


www.example.com
Deliver
ID: ABCDEFGH01234567890
Options
Active
Version 25
Purge
Check ca

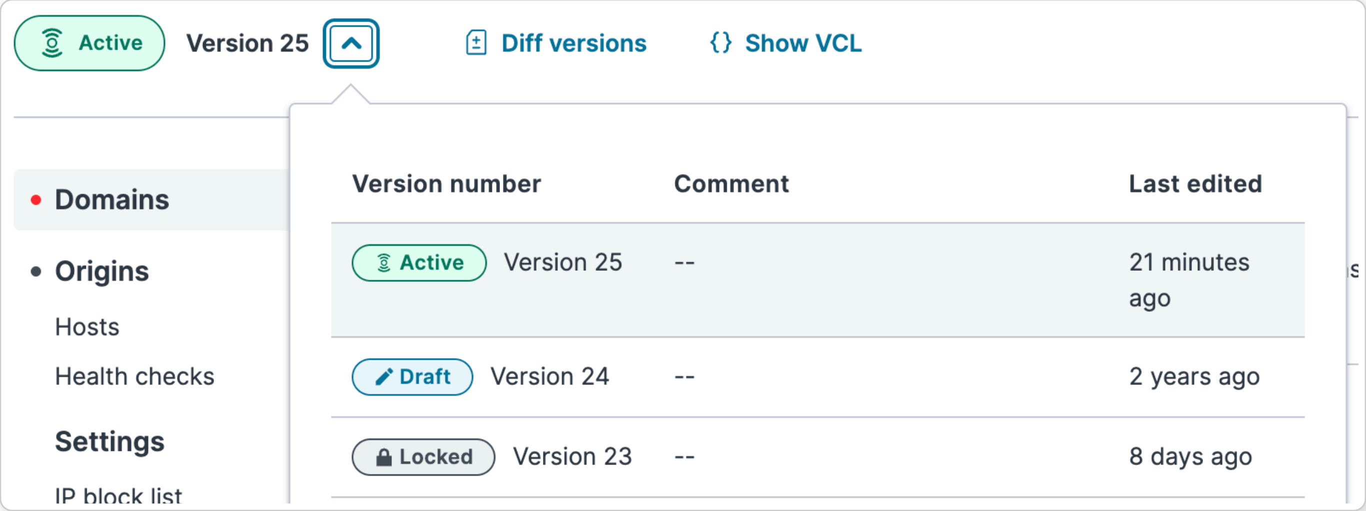
example

Service	Type	Configuration
Demo Example ABCDEFGH01234567891	Deliver	Active Version 14
example ABCDEFGH01234567892	Compute	Active Version 1

4. Select the appropriate service from the list that appears.

To switch between versions of a specific service, follow these steps:

1. From the **Service configuration** page, click the switcher to the right of the version number.



Version number	Comment	Last edited
Active Version 25	--	21 minutes ago
Draft Version 24	--	2 years ago
Locked Version 23	--	8 days ago

2. Select the appropriate version from the list that appears.

Deleting a service

Fastly allows you to delete any service you create, along with all of its versions. Fastly does not offer a way to delete specific versions of a service, however. Service versions are meant to be a historic log of the changes that were made to a service. To undo changes introduced by a particular service version, you can always go back to a previous version and reactivate or clone a new service version based on any old version.

To delete any service along with all of its versions, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Deactivate**.
4. Click **Confirm and deactivate** to confirm you want to deactivate your service and acknowledge that you no longer want to serve traffic with it.
5. From the **Options** menu, select **Delete** and then click **Confirm and delete** to delete the service.

> IMPORTANT
>
> To minimize the risk of unauthorized use of your domains, we strongly recommend modifying or deleting any DNS CNAME records pointing to the Fastly hostname associated with the deleted service. Follow the instructions on your DNS provider's website.

Other things you can do

In addition to creating or editing services, you can view all your services, rename them, compare versions of them, deactivate or reactivate specific versions of them, and delete them.

Renaming services

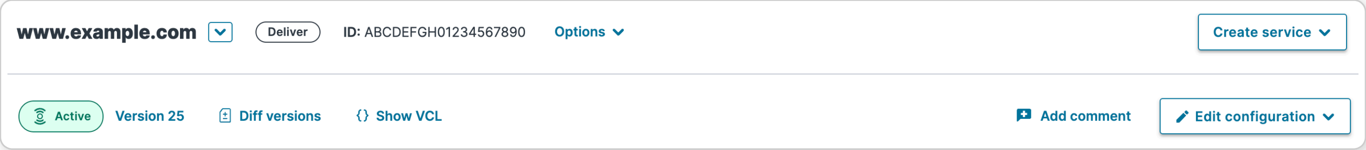
To rename your service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Edit service name**.
4. In the **Service name** field, enter a new service name.
5. Click **Apply**. The newly renamed service name appears.

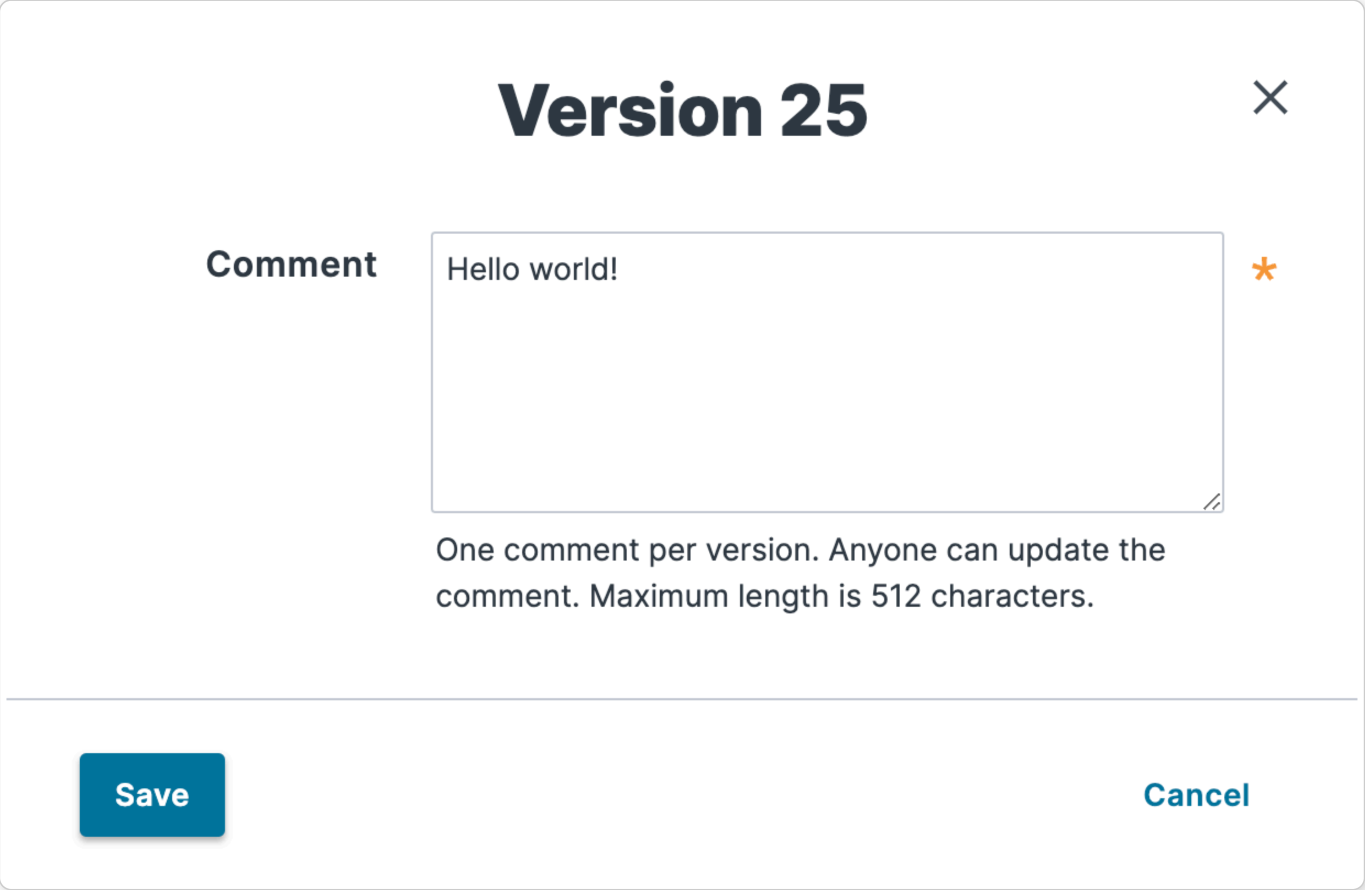
Adding comments to service versions

Service versions can include comments to label them (e.g., to identify reasons for changes in that version). You can add and update version comments on both locked and activated service versions.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Add comment** in the upper right corner of the web interface.



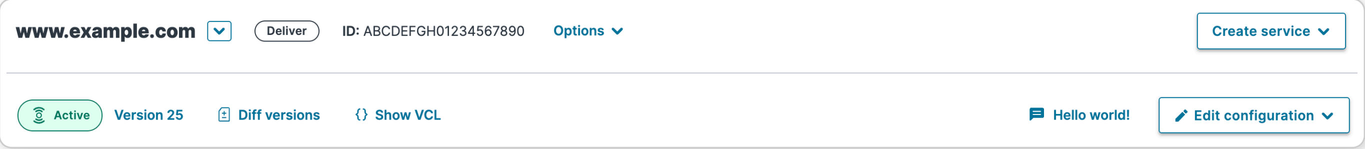
The comment window appears.


Version 25
Save
Cancel

Comment: Hello world!

One comment per version. Anyone can update the comment. Maximum length is 512 characters.

5. In the **Comment** field, enter a meaningful comment for the version.
6. Click **Save**. The truncated version of the comment appears where the Add comment link used to be.


www.example.com
Deliver
Options
Create service
Active
Version 25
Diff versions
Show VCL
Hello world!
Edit configuration

ID: ABCDEFGH01234567890

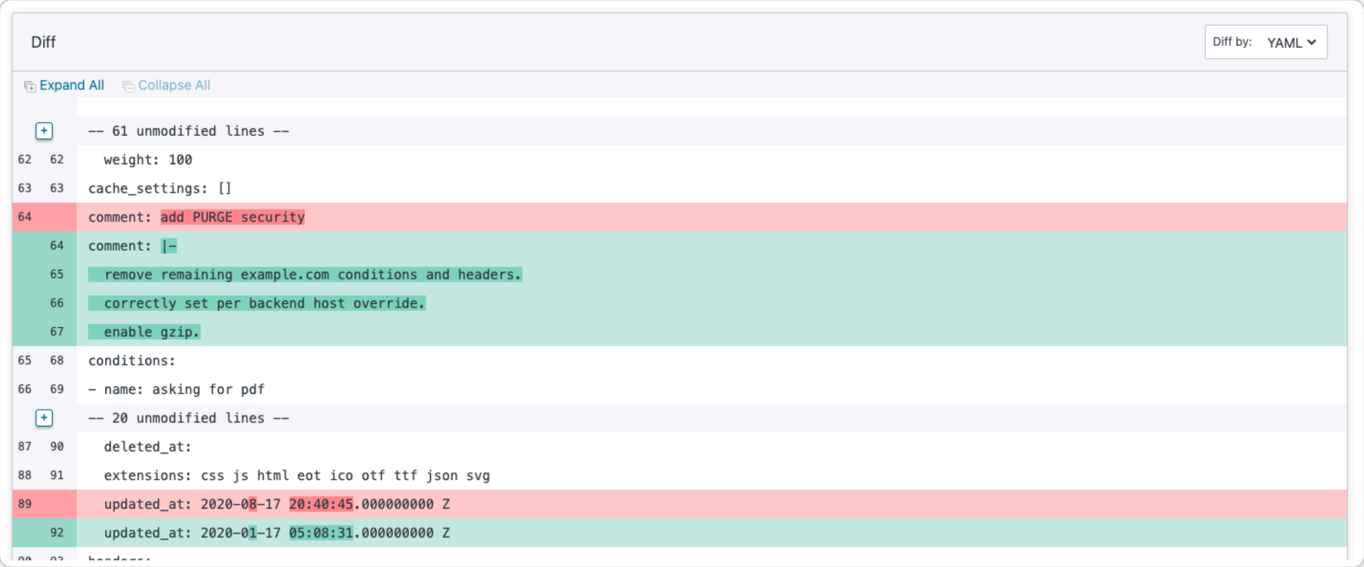
> TIP
>
> You can view service version comments at any time by clicking the service version number to display the version selection menu or by clicking the version comment icon to display the version comment in a separate window. Version comments also appear in the event log to help with account activity monitoring.

Comparing different service versions

To compare two versions of a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Diff versions**. The Diff versions page appears.

Removals are highlighted in red and the additions and changes are highlighted in green. Any large blocks of unaffected configuration lines can be expanded and viewed or collapsed and hidden by clicking on the plus (+) sign to the left of the actual changes, next to the line numbers.



```
         -- 61 unmodified lines --
62  62     weight: 100
63  63   cache_settings: []
64       comment: add PURGE security
    64   comment: |-
    65     remove remaining example.com conditions and headers.
    66     correctly set per backend host override.
    67     enable gzip.
65  68   conditions:
66  69   - name: asking for pdf
         -- 20 unmodified lines --
87  90     deleted_at:
88  91     extensions: css js html eot ico otf ttf json svg
89         updated_at: 2020-08-17 20:40:45.000000000 Z
    92     updated_at: 2020-01-17 05:08:31.000000000 Z
```

> **TIP**
>
> You can change the compared service versions by clicking **Switch versions** and selecting a different version number in the menu that appears.

Deactivating a service

To deactivate a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. From the **Options** menu, select **Deactivate**.
4. In the **Enter service name** field, enter the exact service name to deactivate.
5. Click **Confirm and deactivate** to confirm you want to deactivate your service and acknowledge that you no longer want to serve traffic with it.

> IMPORTANT
>
> To minimize the risk of unauthorized use of your domains, we strongly recommend modifying or deleting any DNS CNAME records pointing to the Fastly hostname associated with the deactivated service. Follow the instructions on your DNS provider's website.

You can also activate or deactivate a service via the API. Did you accidentally delete a service? We can help.

Reactivating a service

To reactivate a service, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Activate**. The service is reactivated.
5. If you removed the DNS CNAME records for the service's domains when you deactivated the service, you should add new DNS CNAME records now.

Getting help with accidental service deletions

Services can be deactivated or deleted. Deactivated services can be reactivated at any time, but once they've been deleted you must contact Customer Support to have them restored. When sending your request, remember to include:

- your customer ID
- your company name
- your service ID (the name of the service you want restored)

Customer Support will notify you when your service has been restored.

What's next

Learn more about working with Compute services.

* * *

Working with domains

/en/guides/working-with-domains

Domains are used to route requests to your service. You associate your domain names with your origin when provisioning a Fastly service so you can properly route requests to your website and ensure that others cannot serve requests to that domain. For example, you could enter `www.example.com`, `blog.example.com`, or even use wildcards such as `*.example.com`. You can add, edit, or remove domains from your service at any time.

TIP

Due to limitations in the DNS specification, Fastly doesn't recommend using apex or second-level domains. An example of an apex domain is `example.com` rather than `www.example.com`.

Before you begin

Be sure you learn about the web interface controls and how to work with services before you start working with your domains.

Domain creation limits

Domains can only be associated with a single service at a time and each service has a limited number of domains that can be associated with it.

Domains can only be associated with one service at a time

If a domain is in use by another service within your account, you can delete the domain and add it to the other service. However, if a domain is used by a service in another Fastly account, it can't be used without being delegated.

If you try to create a domain that is already associated with another service, you may get error messages like:

`Domain [domain name] is already taken by service [service name].`

In this case, follow the steps to delete a domain from your service before creating the domain on your new service.

If you try to create a domain that is already owned by another customer, you may get error messages like:

`Domain [domain name] is taken by another customer.` `Domain [domain name] is owned by another customer.`

If you receive one of these errors when adding a domain that you rightfully should have access to, or if you need to delegate a domain to another account or customer (for example, delegating a domain from a test account to a production account), contact support to help you delegate the domain to the correct account.

Services are limited to a set number of domains

We set a limit on the number of domains you can create per service by default based on your account type and any packaged offering you've purchased.

Once you reach that limit, error messages may appear that look something like this:

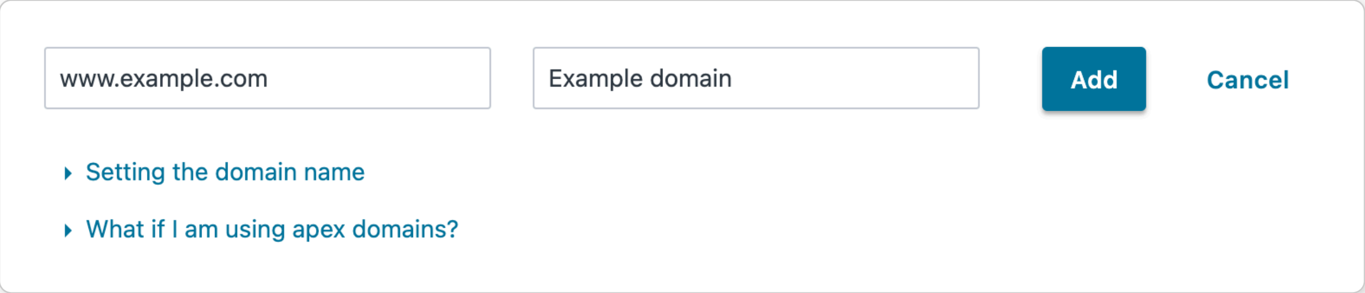
```
{
  "msg": "An error occurred while connecting to the fastly API, please try your request
  "detail": "Exceeding max number of domains: 10"
}
```

If you're receiving a limit message and need to create more services or domains, contact support for assistance. Fastly support engineers can not only increase the number of services that you can use, they can suggest other ways to design what you are trying to achieve.

Creating a domain for the first time

Follow the steps below to add a domain to your service for the first time:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.

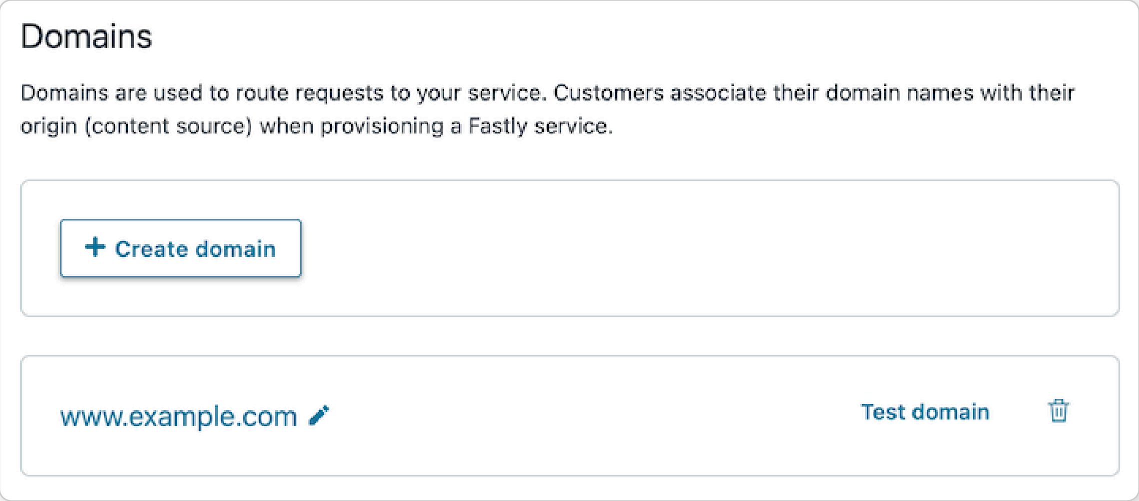

www.example.com
Example domain
Add
Cancel
Setting the domain name
What if I am using apex domains?

4. Fill out the domain creation fields as follows:
    - In the **Domain Name** field, enter your domain name.
    - *(Optional)* In the **Comment** field, enter a comment that describes the domain.
5. Click the **Add** button. Your new domain appears in the list of domains.
6. Click **Activate** to deploy your configuration changes.
7. If you haven't already, add a CNAME DNS record for your domain name to begin routing client traffic through Fastly services instead of directly to your origin.

Creating additional domains

Follow the steps below to add additional domains to your service:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.

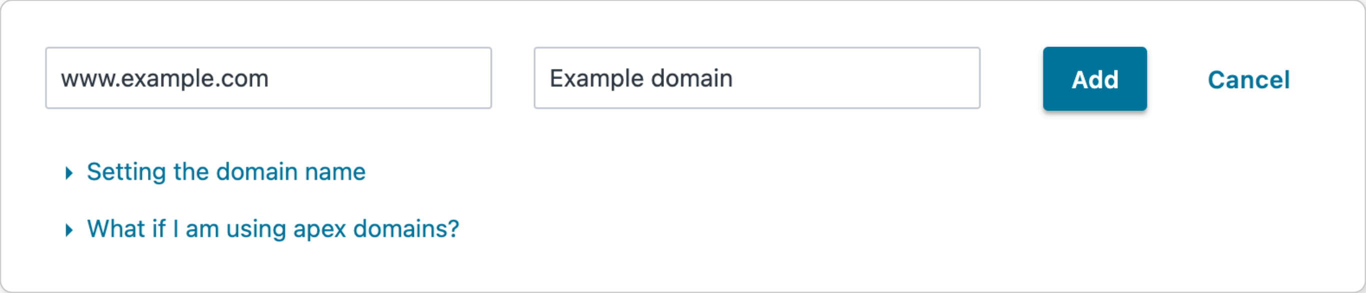


Domains

Domains are used to route requests to your service. Customers associate their domain names with their origin (content source) when provisioning a Fastly service.

+ Create domain
www.example.com
Test domain

4. Click **Create domain**.



5. Fill out the domain creation fields as follows:
    - In the **Domain Name** field, enter your domain name.
    - *(Optional)* In the **Comment** field, enter a comment that describes the domain.
6. Click **Add**. Your new domain appears in the list of domains.
7. Click **Activate** to deploy your configuration changes.
8. If you haven't already, add CNAME DNS records for your domain name to begin routing client traffic through Fastly services instead of directly to your origin.

Creating a domain using the API

You can use Fastly's API to programmatically add domains to your service. To add a domain to your service, make the following API call in a terminal application:

```
$ curl -H "Fastly-Key: FASTLY_API_TOKEN" -X POST https://api.fastly.com/service/<service d>
```

The response will look like this:

```
{
  "comment": "",
  "name": "www.example.com",
  "service_id": "<service_id>",
  "version": <version_id>
}
```

Testing a domain

After you activate your service configuration, but before you change your DNS entries to send your domain to our servers, you can check to see how your service is pulled through our network. Testing your domain can help you identify DNS issues or problems with your Fastly configuration.

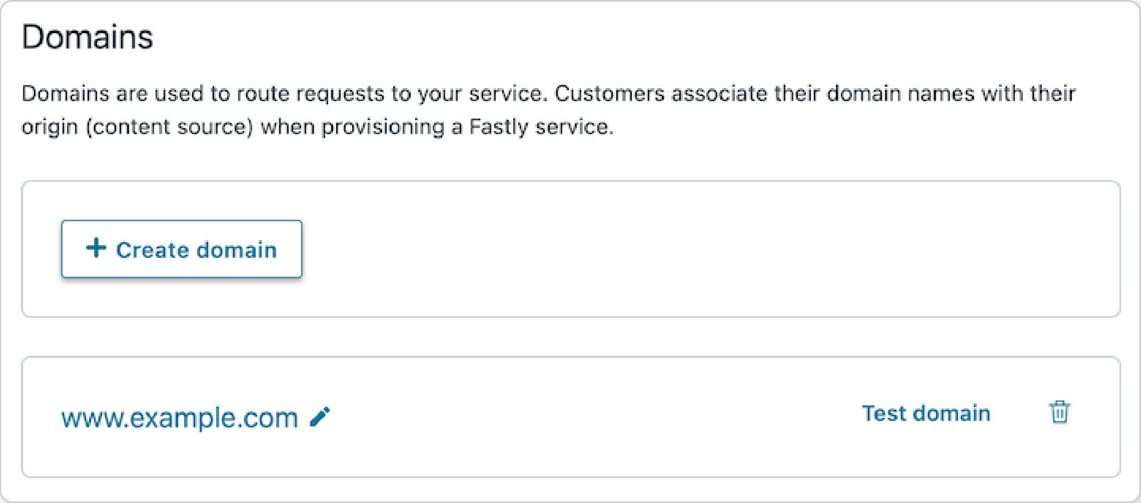
Using the web interface

To use the web interface to test your domain on Fastly before you make a final CNAME change, follow the steps below:

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.



Domains

Domains are used to route requests to your service. Customers associate their domain names with their origin (content source) when provisioning a Fastly service.

4. Click **Test domain** next to the domain you want to test.

5. Verify that your website appears in a new tab in your web browser.

Using command line utilities

To use command line utilities to test your domain on Fastly before you make a final CNAME change, follow the steps below:

1. Find the IP address of a Fastly POP.
2. Add a domain Host entry to your hosts file.
3. Test the domain in a web browser.

Determining the IP address of a Fastly POP

Use the `host`, `nslookup`, or `dig` command to determine the IP address of a Fastly POP. We return different IP addresses depending on the location and state of the network making the request. As a result, the IP address you receive for any specific query may differ from the below examples but should result in the best performance for you.

> ✓ TIP
>
> For non-TLS requests, use `dualstack.nonssl.global.fastly.net` for connection over IPv6. For TLS requests, use the custom TLS CNAME record provided in the web interface or by Fastly support. For more information about the Fastly TLS service, see our guide on TLS service options.

Running `host` for your domain returns both IPv4 addresses and IPv6 addresses:

```
$ host www.example.com.
www.example.com.is an alias for dualstack.nonssl.global.fastly.net.
dualstack.nonssl.global.fastly.net has address 199.232.144.204
dualstack.nonssl.global.fastly.net has IPv6 address 2a04:4e42:64::204
```

In this example, the IPv4 address is `199.232.144.204` and the IPv6 address is `2a04:4e42:64::204`.

Running `nslookup` for `dualstack.nonssl.global.fastly.net` also returns both IPv4 addresses and IPv6 addresses:

```
$ nslookup  dualstack.nonssl.global.fastly.net.
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   dualstack.nonssl.global.fastly.net
Address: 199.232.144.204
Name:   dualstack.nonssl.global.fastly.net
Address: 2a04:4e42:64::204
```

In this example, the IPv4 address is `199.232.144.204` and the IPv6 address is `2a04:4e42:64::204`.

Alternatively, you can run `dig` to determine the IP address, but you must specify an A query for IPv4 addresses or a AAAA query for IPv6 addresses.

Running `dig` for your domain with an A query returns:

```
$ dig +noall +answer -t A www.example.com.
www.example.com.        3600    IN      CNAME   dualstack.nonssl.global.fastly.net.
dualstack.nonssl.global.fastly.net. 30 IN A     199.232.144.204
```

The IPv4 address is `199.232.144.204`

Running `dig` for your domain with an AAAA query returns:

```
$ dig +noall +answer -t AAAA www.example.com.
www.example.com.        3600    IN      CNAME   dualstack.nonssl.global.fastly.net.
dualstack.nonssl.global.fastly.net. 30 IN AAAA  2a04:4e42:64::204
```

The IPv6 address is `2a04:4e42:64::204`.

Modifying your hosts file

You can temporarily add a static IP address and domain Host entry to the hosts file on your computer. Use the following anycast IPs for testing your sites served over `dualstack.nonssl.global.fastly.net`.

IPv4	IPv6
151.101.0.204	2a04:4e42::204
151.101.64.204	2a04:4e42:200::204
151.101.128.204	2a04:4e42:400::204
151.101.192.204	2a04:4e42:600::204

For example, if the domain you are testing is `www.example.com` and you are using the anycast IPv4 example `151.101.0.204`, you would add this entry to the file and save the changes:

```
151.101.0.204 www.example.com
```

TIP

On machines running macOS or Linux, your hosts file is `/etc/hosts`. On Windows-based machines, it's `C:\Windows\System32\Drivers\etc\hosts`.

Testing your domain

Test your domain to see how Fastly pulls it through our network by restarting your browser if it's already running, and then typing your domain in the address field. You should now see the updated domain in the address field indicating requests are being sent to the Fastly POP.

Alternatively, you can test the domain using a ping command to verify that your domain is being served by a Fastly POP address. In this case, `ping www.example.com` would display the Fastly POP address `151.101.56.204`.

Be sure to remove the Host entry from your hosts file after you make CNAME changes to point your domain to Fastly.

Deleting a domain

Follow the steps below to delete a domain from your service:

1. On the Domains page, click the trash next to the domain you want to delete.
2. Click **Confirm and delete** to confirm you want to delete your domain.
3. Click **Activate** to deploy your configuration changes.

IMPORTANT

To minimize the risk of unauthorized use of your domains, we strongly recommend modifying or deleting any DNS CNAME records pointing to the Fastly hostname associated with the deleted domain. Follow the instructions on your DNS provider's website.

What's next

Learn more about working with hosts and working with health checks as you continue to refine versions of your service configurations.

* * *

Working with health checks

/en/guides/working-with-health-checks

Health checks monitor the status of your hosts. Fastly performs health checks on your origin server based on the Check frequency setting you select in the Create a new health check page and the packaged offering you may have purchased. The Check frequency setting you select specifies approximately how many requests per minute Fastly POPs are checked to see if they pass. There is roughly one health check per Fastly POP per period. Any checks that pass will be reported as "healthy."

Before you begin

Be sure you learn about the web interface controls and how to work with services before you start setting up health checks.

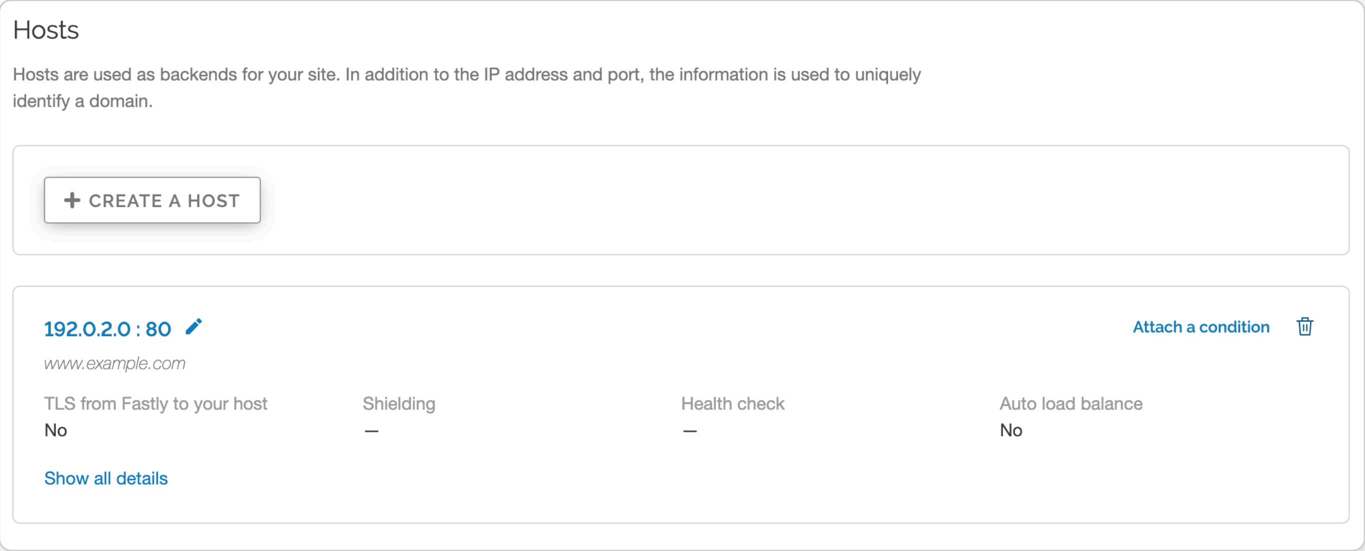
Limitations and considerations

When you create a health check, you have the option to create and attach a custom header value. If you elect to take advantage of this option, keep the following things in mind:

- The header value should only be used for the custom header check and should not be used for other purposes. You should always maintain appropriate security controls and secrets management procedures, including appropriate access limitations for customer header values.
- All headers combined must be less than 64KB. Each individual header must be less than 8190 bytes.
- There may not be more than 100 headers linked to an individual health check.

Creating a health check

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.



Hosts

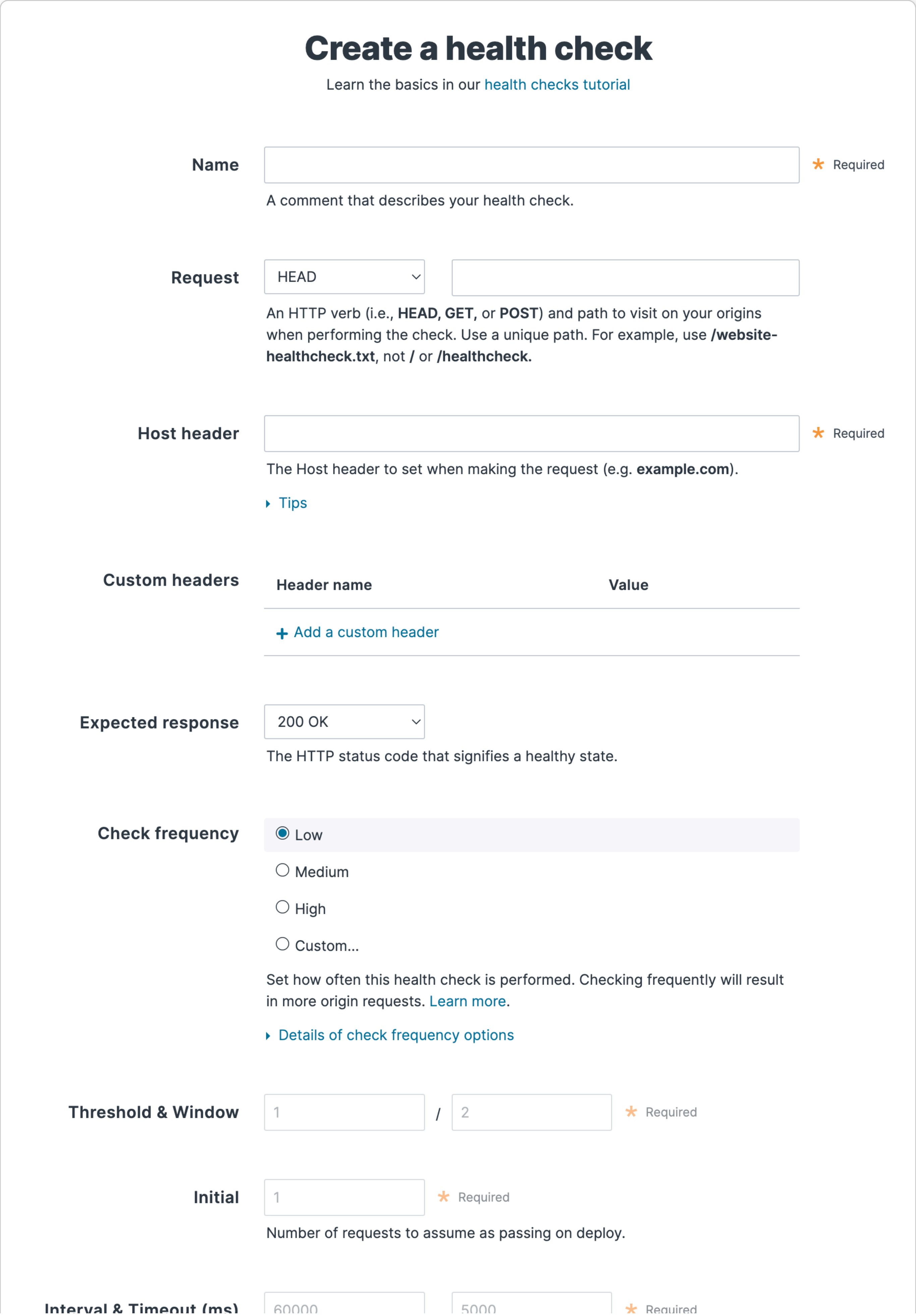
Hosts are used as backends for your site. In addition to the IP address and port, the information is used to uniquely identify a domain.

192.0.2.0 : 80

www.example.com

TLS from Fastly to your host	Shielding	Health check	Auto load balance
No	—	—	No

5. Click **Create health check**.



Create a health check

Learn the basics in our health checks tutorial

Name ☐ * Required

A comment that describes your health check.

Request HEAD ☐

An HTTP verb (i.e., **HEAD, GET,** or **POST**) and path to visit on your origins when performing the check. Use a unique path. For example, use **/website-healthcheck.txt**, not **/** or **/healthcheck.**

Host header ☐ * Required

The Host header to set when making the request (e.g. **example.com**).

▸ Tips

Custom headers

Header name	Value
+ Add a custom header	

Expected response 200 OK

The HTTP status code that signifies a healthy state.

Check frequency

● Low

○ Medium

○ High

○ Custom...

Set how often this health check is performed. Checking frequently will result in more origin requests. Learn more.

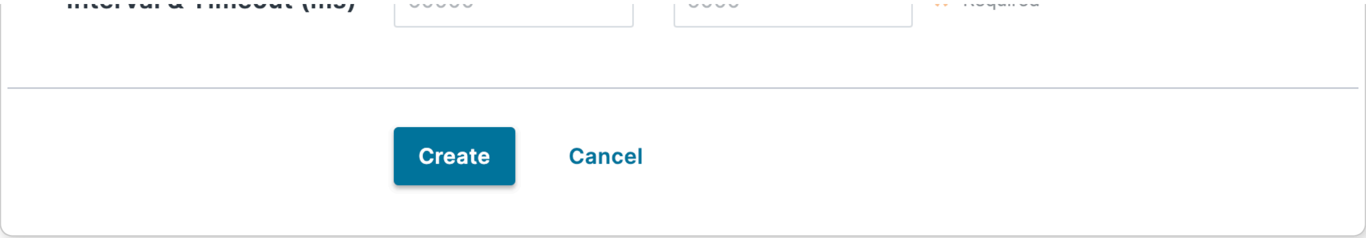
▸ Details of check frequency options

Threshold & Window 1 / 2 * Required

Initial 1 * Required

Number of requests to assume as passing on deploy.

Interval & Timeout (ms) 60000 5000 * Required



6. Fill out the **Create a health check** fields as follows:
    - In the **Name** field, enter a human-readable identifier for the health check (e.g., `West Coast Origin Check`).
    - From the **Request** menu, select an HTTP verb. In the **Request** field, enter the path to visit when performing the check. Use a unique path. For example, use `/website-healthcheck.txt`, not `/` or `/healthcheck`.
    - In the **Host header** field, enter the HTTP Host header to set when making the request (e.g., `example.com`).
    - In the **Custom header** field, click **Add a custom header** to add additional headers to your health check requests. You can specify up to 100 additional headers.
        - In the **Header name** column, enter a human-readable identifier for the header.
        - In the **Value** column, enter the value you want assigned to the header.

    > IMPORTANT
    >
    > The header value should only be used for the custom header check and should not be used for other purposes.

    - From the **Expected response** menu, select the HTTP status code the origin servers must respond with for the check to pass (usually `200 OK`).
    - In the **Check frequency** section, select a setting to control how often the health check is performed.
        - **Low:** One request every minute from each data center, where "healthy" means 1 out of 2 must pass.
        - **Medium:** One request every 15 seconds from each data center, where "healthy" means 3 out of 5 must pass.
        - **High:** One request every 2 seconds from each data center, where "healthy" means 7 out of 10 must pass.
        - **Custom:** A custom frequency you specify.
    - In the **Threshold & Window** fields, enter the number of successes per total number health checks. For example, specifying `3/5` means 3 out of 5 checks must pass to be reported as healthy.
    - In the **Initial** field, enter the number of requests to assume as passing on deploy. For example, if the Threshold & Window field is set to `3/5` and the Initial field is set to `1`, a backend would be marked as “unhealthy” until it passed two more health checks to reach the required minimum.
    - In the **Interval & Timeout (ms)** fields, enter times. Interval represents the period of time for the requests to run. Timeout represents the wait time until request is considered failed. Both times are specified in milliseconds.
7. Click **Create**.

Your new health check now appears in the list of checks.

Assigning a health check

Health checks do nothing on their own, but they can be added as a special parameter to an origin server in your configuration.

1. Edit one of your existing origin servers by clicking the origin server's name.
2. From the **Health checks** menu, select the health check you just created.
3. Click **Update**.

Fastly will now use the health check to monitor the selected origin server.

Creating and assigning health checks using VCL

Health checks can also be created and assigned using VCL. See our developer documentation on health checks for more information.

Troubleshooting

Fastly will periodically check your origin server based on the options chosen. Pay special attention to the HTTP Host header. A common mistake is setting the wrong host. If the origin server does not receive a host it expects, it may issue a 301 or 302 redirect causing the health check to fail. Also, Varnish requires the origin server receiving the health check requests to close the connection for each request. If the origin server does not close the connection, health checks will time out and fail.

If an origin server is marked unhealthy due to health checks, Fastly will stop attempting to send requests to it. Once all of your origin servers are marked unhealthy, Fastly will return a 503 error (service unavailable) to the client unless you tell it otherwise. You can configure Fastly to attempt to serve stale content instead until your origin servers become available again.

What's next

Learn more about working with domains and working with hosts as you continue to refine versions of your service configurations.

* * *

Working with hosts

🔗 /en/guides/working-with-hosts

A *host*, also referred to as a *backend* or *origin*, is a web server or cloud service that contains the content of your website or application. For example, a host could be a physical or virtual web server, an application running on a platform as a service, a serverless function running on a serverless platform, or even a static storage bucket.

Before Fastly can cache your content, you must add a host to your Fastly service configuration by specifying its hostname or IP address. When Fastly receives a request from a user, we fetch the objects from your host and return the response to the user.

NOTE

To learn more about configuring third-party services for use as hosts, refer to our integration guides and our developer documentation on integrating with backend technologies.

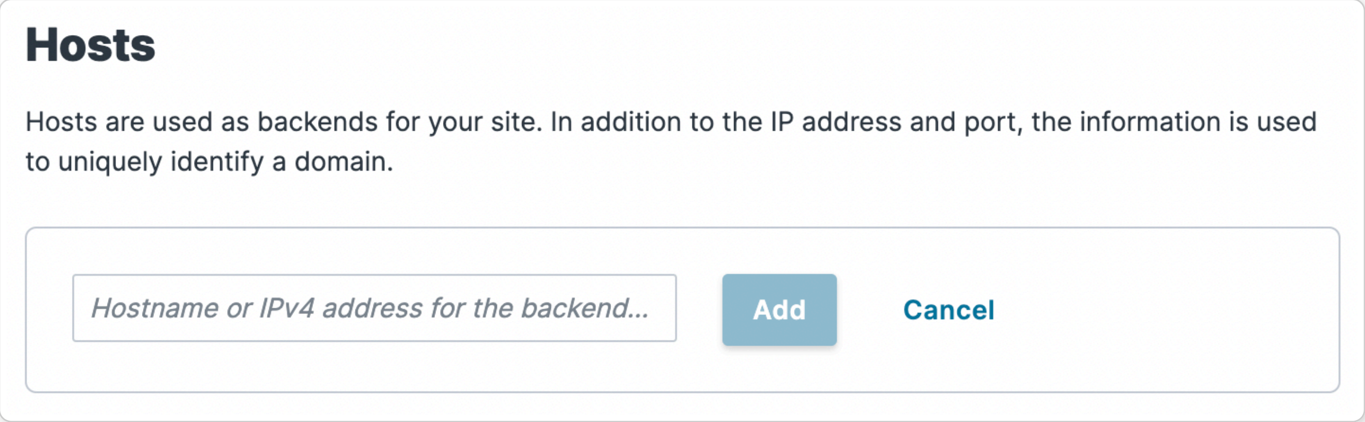
Before you begin

Be sure you learn about the web interface controls and how to work with services before you start working with your hosts.

Adding a host

To add a host to your Fastly service configuration, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. Click **Create a host**.



Hosts

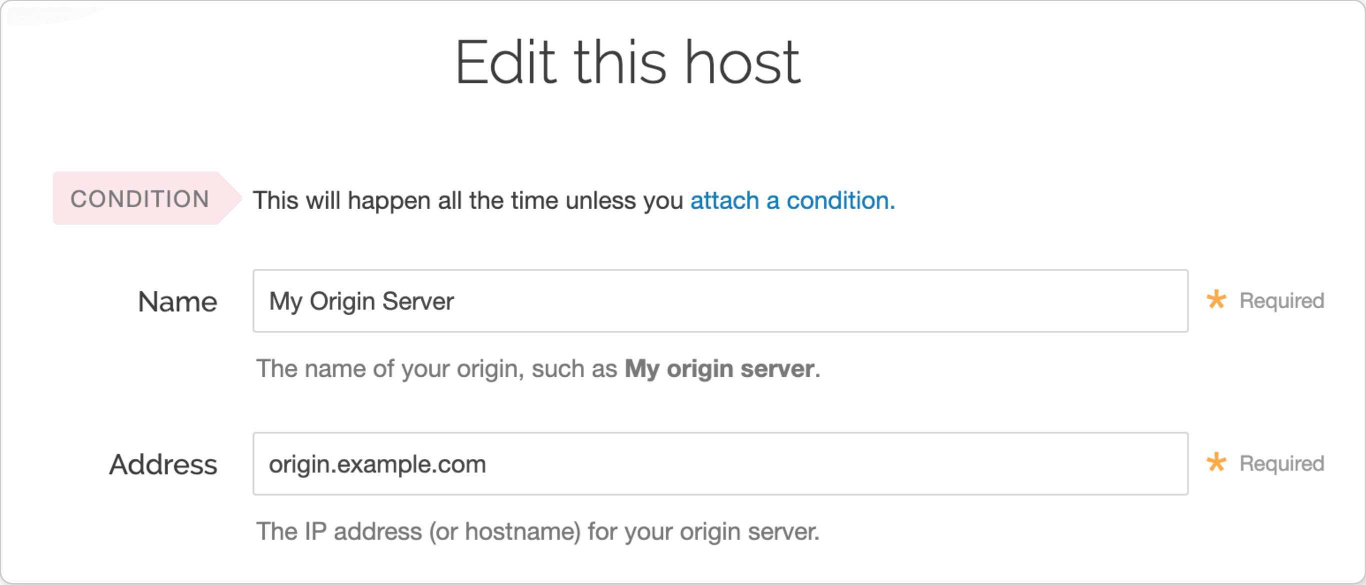
Hosts are used as backends for your site. In addition to the IP address and port, the information is used to uniquely identify a domain.

6. Fill out the **Hosts** field by entering the hostname or IP address of your origin server. Entering a hostname automatically enables Transport Layer Security (TLS) and assigns port 443. Entering an IP address disables TLS and assigns port 80.
7. Click **Add** to add your host.

Editing a host

After you've created your host, you can edit the settings by following the steps below:

1. In the **Hosts** area, click the pencil next to the Host you want to edit.



Edit this host

This will happen all the time unless you attach a condition.

Name: My Origin Server

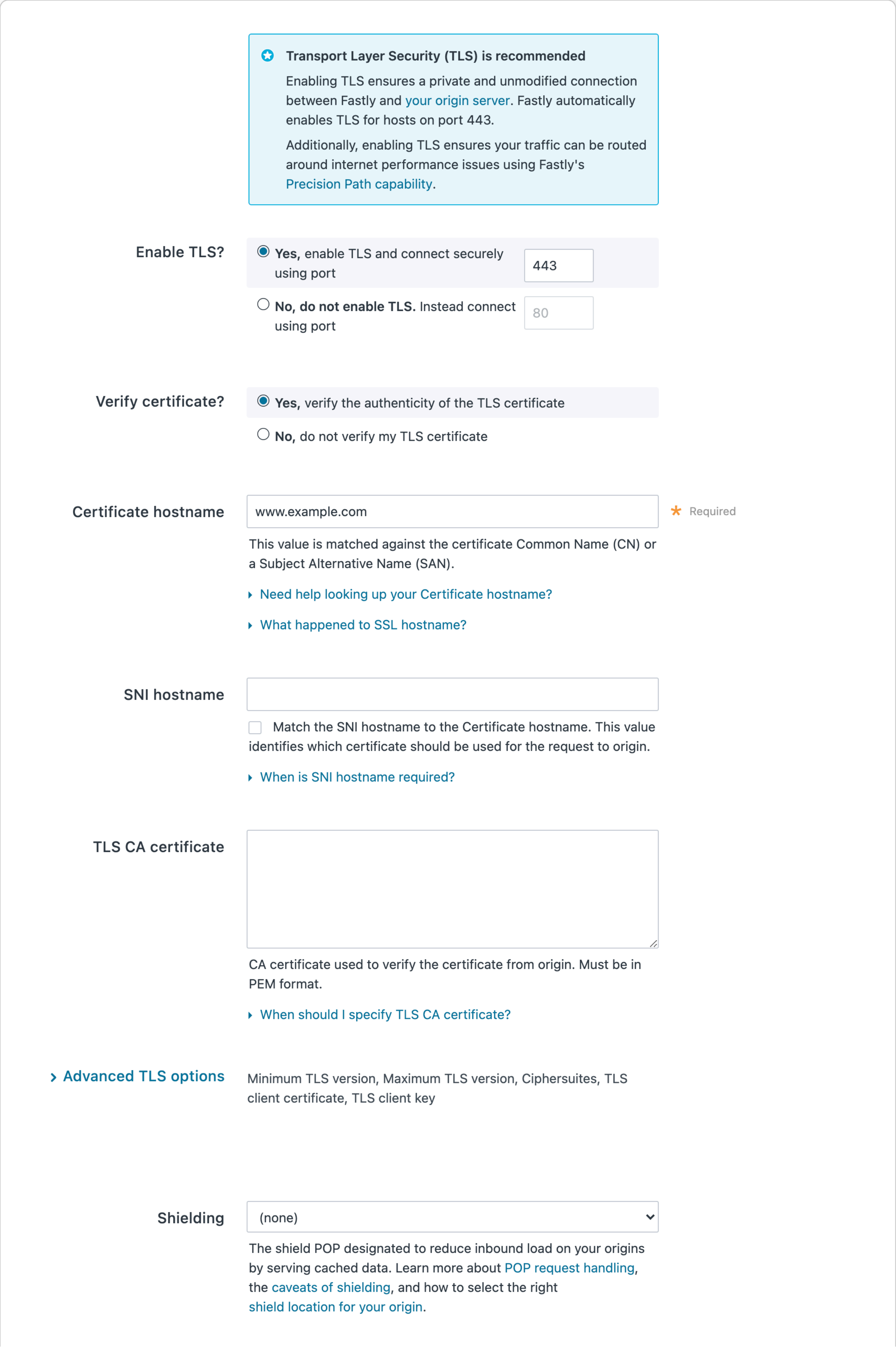
The name of your origin, such as **My origin server**.

Address: origin.example.com

The IP address (or hostname) for your origin server.

2. Fill out the **Edit this host** fields as follows:
   - In the **Name** field, enter the name of your server (for example, `My Origin Server`). This name is displayed in the Fastly web interface.
   - In the **Address** field, enter the IP address (or hostname) of your origin server.

     See Understanding the difference between certificate hostname and SNI hostname values for more information about hostnames.



Transport Layer Security (TLS) is recommended

Enabling TLS ensures a private and unmodified connection between Fastly and your origin server. Fastly automatically enables TLS for hosts on port 443.

Additionally, enabling TLS ensures your traffic can be routed around internet performance issues using Fastly's Precision Path capability.

Enable TLS?

● **Yes,** enable TLS and connect securely using port 443

○ **No, do not enable TLS.** Instead connect using port 80

Verify certificate?

● **Yes,** verify the authenticity of the TLS certificate

○ **No,** do not verify my TLS certificate

Certificate hostname: www.example.com

* Required

This value is matched against the certificate Common Name (CN) or a Subject Alternative Name (SAN).

- Need help looking up your Certificate hostname?
- What happened to SSL hostname?

SNI hostname:

☐ Match the SNI hostname to the Certificate hostname. This value identifies which certificate should be used for the request to origin.

- When is SNI hostname required?

TLS CA certificate:

CA certificate used to verify the certificate from origin. Must be in PEM format.

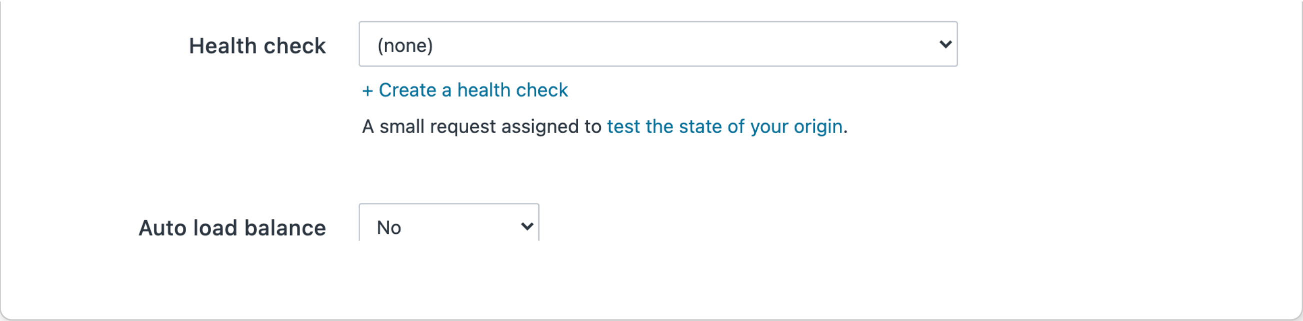
- When should I specify TLS CA certificate?

› Advanced TLS options

Minimum TLS version, Maximum TLS version, Ciphersuites, TLS client certificate, TLS client key

Shielding: (none)

The shield POP designated to reduce inbound load on your origins by serving cached data. Learn more about POP request handling, the caveats of shielding, and how to select the right shield location for your origin.



Health check (none)

A small request assigned to test the state of your origin.

Auto load balance No

3. Configure the **Transport Layer Security (TLS)** settings as follows:
    - From the **Enable TLS?** options, leave the default set to **Yes** if you want to enable TLS to secure the connection between Fastly and your origin. To enable TLS, a valid SSL certificate must be installed on your origin server and port 443 (or the specified port) must be open in the firewall. You can select **No** if you do not want to use TLS.
    - From the **Verify certificate?** options, leave default set to **Yes** if you want to verify the authenticity of the TLS certificate. Selecting **No** means the certificate will not be verified.

      > ⚠ WARNING
      >
      > Not verifying the certificate has serious security implications, including vulnerability to man-in-the-middle attacks. Consider uploading a CA certificate instead of disabling certificate validation.

    - In the **Certificate hostname** field, enter the hostname associated with your TLS certificate. This value is matched against the certificate common name (CN) or a subject alternate name (SAN) depending on the certificate you were issued.
    - *(Optional)* In the **SNI hostname** field, enter the hostname of a different certificate to be used for the request to origin if you are using Server Name Indication (SNI) to put multiple certificates on your origin. Alternatively, select the checkbox to match the SNI hostname to the Certificate hostname. This information also gets sent to the server in the TLS handshake. See our guidance on understanding the difference between certificate hostname and SNI hostname values for additional details.
    - *(Optional)* In the **TLS CA certificate** field, provide a certificate in PEM format if you're using a certificate that is either self-signed or signed by a certification authority (CA) not commonly recognized by major browsers. The PEM format looks like this:

      ```
      -----BEGIN CERTIFICATE-----
      MIIDrzCCApegAwIBAgIQCDvgVpBCRrGhdWrJWZHHSjANBgkqhkiG9w0BAQUFADBh
      MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMRkwFwYDVQQLExB3
      d3cuZGlnaWNlcnQuY29tMSAwHgYDVQQDExdEaWdpQ2VydCBHbG9iYWwgUm9vdCBD
      QTAeFw0wNjExMTAwMDAwMDBaFw0zMTExMTAwMDAwMDBaMGExCzAJBgNVBAYTAlVT
      MRUwEwYDVQQKEwxEaWdpQ2VydCBJbmMxGTAXBgNVBAsTEHd3dy5kaWdpY2VydC5j
      b20xIDAeBgNVBAMTF0RpZ2lDZXJ0IEdsb2JhbCBSb290IENBMIIBIjANBgkqhkiG
      9w0BAQEFAAOCAQ8AMIIBCgKCAQEA4jvhEXLeqKTTo1eqUKKPC3eQyaKl7hLOllsB
      CSDMAZOnTjC3U/dDxGkAV53ijSLdhwZAAIEJzs4bg7/fzTtxRuLWZscFs3YnFo97
      nh6Vfe63SKMI2tavegw5BmV/Sl0fvBf4q77uKNd0f3p4mVmFaG5cIzJLv07A6Fpt
      43C/dxC//AH2hdmoRBBYMql1GNXRor5H4idq9Joz+EkIYIvUX7Q6hL+hqkpMfT7P
      T19sdl6gSzeRntwi5m3OFBqOasv+zbMUZBfHWymeMr/y7vrTC0LUq7dBMtoM1O/4
      gdW7jVg/tRvoSSiicNoxBN33shbyTApOB6jtSj1etX+jkMOvJwIDAQABo2MwYTAO
      BgNVHQ8BAf8EBAMCAYYwDwYDVR0TAQH/BAUwAwEB/zAdBgNVHQ4EFgQUA95QNVbR
      TLtm8KPiGxvDl7I90VUwHwYDVR0jBBgwFoAUA95QNVbRTLtm8KPiGxvDl7I90VUw
      DQYJKoZIhvcNAQEFBQADggEBAMucN6pIExIK+t1EnE9SsPTfrgT1eXkIoyQY/Esr
      hMAtudXH/vTBH1jLuG2cenTnmCmrEbXjcKChzUyImZOMkXDiqw8cvpOp/2PV5Adg
      06O/nVsJ8dWO41P0jmP6P6fbtGbfYmbW0W5BjfIttep3Sp+dWOIrWcBAI+0tKIJF
      PnlUkiaY4IBIqDfv8NZ5YBberOgOzW6sRBc4L0na4UU+Krk2U886UAb3LujEV0ls
      YSEY1QSteDwsOoBrp+uvFRTp2InBuThs4pFsiv9kuXclVzDAGySj4dzp30d8tbQk
      CAUw7C29C79Fv1C5qfPrmAESrciIxpg0X40KPMbp1ZWVbd4=
      -----END CERTIFICATE-----
      ```

4. To apply advanced TLS options, click **Advanced TLS options** and decide which of the optional fields to specify, if any. For more information about these settings, see Advanced TLS options.

5. Configure the remaining **Create a host** settings as follows:

    - *(Optional)* From the **Shielding** menu, select a POP to enable the shielding feature. For more information, see our guide on shielding.

    - *(Optional)* From the **Health check** menu, select a health check for this origin server. For more information, see our guide on working with health checks.

    - *(Optional)* From the **Auto load balance** menu, select **Yes** to enable load balancing for this origin server. For more information, see our guide on load balancing.

    - If you enabled load balancing, enter a weight in the **Weight** field.

    - *(Optional)* In the **Override host** field, enter the hostname of your override Host header based on the origin you're using. The value in this field will take precedence over anything you've set using the global override host configuration. To see example Host headers for third-party services, refer to our developer documentation on overriding the Host header.

        > NOTE
        >
        > To override the incoming Host header on your origin using your Fastly service, refer to the Specifying the override host guide.

6. Click **Advanced options** and decide which of the optional fields to change, if any:

    - *(Optional)* In the **Maximum connections** field, enter the maximum number of connections for your backend. The default limit is 200 connections per cache node to protect your origins from being overloaded.

    - *(Optional)* In the **Connection timeout** field, enter how long, in milliseconds, to wait for a connection timeout. The default is 1000 milliseconds.

    - *(Optional)* In the **First byte timeout** field, enter how long, in milliseconds, to wait for a first byte timeout. The default is 15000 milliseconds.

    - *(Optional)* In the **Between bytes timeout** field, enter how long, in milliseconds, to wait between bytes. The default is 10000 milliseconds.

7. Click **Update**. The new override host appears under the **Show all details** field of the **Override host** section and a code block is added to the origin definition in your VCL that will look similar to the following:

```
Backend F_Host_1 {
    .host = "..."; # IP or hostname
    .host_header = "example.com";
    .always_use_host_header = true;
     ...
}
```

8. Click **Activate** to deploy your configuration changes.

Advanced TLS options

If you enabled TLS for the connection between Fastly and your host, you can configure optional settings by clicking the **Advanced TLS options** link.

Specifying acceptable TLS protocol versions

If your origin server supports modern TLS protocol versions, you can customize the TLS protocols Fastly will use to connect to it by setting a **Minimum TLS Version** and **Maximum TLS Version**. We recommend setting both to the most up-to-date TLS protocol, currently 1.3, if your origin supports it.

Use the `openssl` command to verify your origin supports a given TLS protocol version. For example:

```
$ openssl s_client -connect origin.example.com:443 -tls1_3
```

Replace `-tls1_3` with `tls1_2`, `tls1_1` and `tls1_0` to test other protocol versions. Fastly does not support SSLv2 or SSLv3.

> IMPORTANT
>
> In line with security best practices, Fastly recommends enabling servers with version 1.3 of the TLS protocol by default. For backend connections from our edge nodes to customer origins, Fastly supports TLS 1.3, 1.2, 1.1, and 1.0 depending on the versions of the protocol in use on the origin server. Fastly will continue to support TLS 1.0 based on the ServerHello message as described in RFC 5246 if the server selects TLS 1.0 as the highest supported version.

Specifying acceptable TLS cipher suites

Fastly supports configuring the OpenSSL cipher suites used when connecting to your origin server. This allows you to turn specific cipher suites on or off based on security properties and origin server support. The **Ciphersuites** setting accepts an OpenSSL formatted cipher list. We recommend using the strongest cipher suite your origin will support as detailed by the Mozilla SSL Configuration Generator.

Use the `openssl` command to verify your origin supports a given cipher suite. For example:

```
$ openssl s_client -connect origin.example.com:443 -tls1_2 -cipher ECDHE-RSA-AES128-GCM-A2
```

Replace `-cipher ECDHE-RSA-AES128-GCM-SHA256` with the cipher suite to test.

Specifying a TLS client certificate and key

To ensure TLS connections to your origin come from Fastly and aren't random, anonymous requests, set your origin to verify the client using a client certificate. Simply paste the certificate and private key in PEM form into the **TLS client certificate** text box.

> IMPORTANT
>
> The private key must not be encrypted with a passphrase.

Then configure your backend to require client certificates and verify them against the CA cert they were signed with. Here are some ways of doing that:

- Apache
- NGINX

- IIS

Fastly also supports securing your connection to origin with mutual TLS (mTLS).

Understanding the difference between certificate hostname and SNI hostname values

The following explains the difference between a certificate and SNI hostname value:

The certificate hostname (`ssl_cert_hostname`). This hostname validates the certificate at origin. It is always required. This value should match the certificate common name (CN) or an available subject alternate name (SAN). It displays as `ssl_cert_hostname` in VCL. This doesn't affect the SNI certification. You can set this value in **Certificate hostname** field.

The SNI hostname (`ssl_sni_hostname`). This hostname determines which certificate should be used for the TLS handshake. SNI is generally only required when your origin is using shared hosting, such as Amazon S3, or when you use multiple certificates at your origin. SNI allows the origin server to know which certificate to use for the connection. This value displays as `ssl_sni_hostname` in VCL. This doesn't affect the certificate validation.

The table below shows you what happens when you set the Certificate and SNI hostname values in the TLS settings:

If Certificate hostname contains...	and SNI hostname contains...	then the Certificate Validation value will be...	and the SNI value will be...
`www.example.com`	nothing	`www.example.com`	nothing
`www.example.com`	`www.example.org`	`www.example.com`	`www.example.org`

About the `ssl_hostname` value (deprecated). The `ssl_hostname` value has been deprecated and replaced with `ssl_cert_hostname` and `ssl_sni_hostname`. Use these two values instead.

IMPORTANT

If you use an IP address for your `.host` value (i.e., by not entering a value in your certificate hostname), this will generate an error where the certificate hostname specified in your service's origin TLS settings doesn't match either the Common Name (CN) or available Subject Alternate Names (SANs).

Using a wildcard certificate

If you're using a wildcard certificate, you can use any SNI hostname that matches the wildcard certificate. The SNI hostname must be a fully-qualified domain name (FQDN), per RFC 6066.

The table below shows a variety of possible combinations of certificate and SNI hostnames that could be used with a wildcard certificate for `*.example.com`:

Certificate hostname	SNI hostname
`www.example.com`	`www.example.com`
`live.example.com`	`live.example.com`
`*.example.com`	`www.example.com`

If you set the certificate hostname to `*.example.com`, Fastly will treat it as a literal.

What's next

Learn more about working with domains and working with health checks as you continue to refine versions of your service configurations.

* * *

Category: Essentials

Essentials is a collection of guides that introduce you to core concepts essential to understanding content delivery networks (CDNs) and Fastly.

What is an API?

/en/guides/what-is-api

An application programming interface (API) facilitates communication between technology products and systems. It is sometimes said that APIs “help machines talk to other machines” because by communicating through a structured set of rules, APIs let you programmatically interact with systems instead of using a web interface.

Along with accomplishing the tasks usually completed through a web interface, APIs enable you to integrate with existing applications and automate processes. There are thousands of APIs available on the web, many of which are free to use. You can reference an API directory like the Postman API Network or go to the service or product you want to integrate with to see if they have an API.

While there are multiple types of API architectures, this guide will focus on one of the most common: Representational state transfer (REST). Web services that conform to the REST API style are known as RESTful APIs. RESTful APIs use HTTP, and in this guide you'll see how a call to a RESTful API looks a lot like a webpage URL!

Before you begin

We recommend reading the What is caching? essentials guide to understand what we mean when we refer to requests, responses, clients, and servers in this guide.

APIs in action

You have probably encountered APIs in action without even realizing it! Imagine you’re on the website for your favorite local taco restaurant ordering lunch. At checkout, you need to enter your delivery address. You start typing your address in a search bar and it auto-completes the city, state, and zip code. Next, you need to enter payment information. You notice the payment fields are processed by a trusted mobile payment service. Finally, after you place your order, you can sign up for a loyalty program and earn rewards based on your order. The website lets you sign up using your preferred social media platform.

The address auto-completion, payment processing, and social media sign up are all examples of integrating with APIs. Instead of having to create each of these components, the website uses the Google Maps API to easily collect delivery addresses, the Square API to integrate with Square’s payment processing system, and the Facebook API to quickly make rewards accounts.

How do REST APIs work?

REST APIs use HTTP requests and responses to exchange information across the internet. This is beneficial because a web client making a request and the API server providing the response speak a common language. The message request and response themselves use a common HTTP web protocol.

There are many common use cases for REST APIs, typically along the lines of standard create, retrieve, update, and delete (CRUD) database functions. For example, REST APIs can perform create actions, like using the Facebook API to create a new rewards account. They can also retrieve data, like using the Google Maps API to retrieve addresses. For the remainder of this guide, we'll focus on data retrieval for simplicity's sake.

Anatomy of an API request

Let's get into the anatomy of an API request. An API request has five parts:

- The *base URL*, which is the prefix for the endpoint.
- The *endpoint*, which tells the request where to go.
- The *method*, which determines the type of request taking place.
- The *headers*, which provide information that helps the client and server talk to each other.
- The *body* which contains the information you want sent to the server.

Let's dig a little deeper.

Here is an example of an endpoint for the Google Maps Places API:

`https://maps.googleapis.com/maps/api/place/autocomplete/`

This endpoint has two components. The first is the base URL. This is the domain where the API is served. In this example, `https://maps.googleapis.com` is the base URL. The path of the endpoint determines the resource you're requesting. In this example, we're requesting a specific resource of the Google Maps Places API called Autocomplete. This makes `/maps/api/place/autocomplete` the path.

You might've noticed this looks a lot like a URL. That's because it is! Remember, this request is happening over HTTP. You could put this in your browser and get a very basic HTML page with the response formatted in different ways depending on the API, with JSON being one of the most common response formats.

However, to actually do something with the API, you need a few other components. The first is a method. Methods are predefined keywords that must be included in every request. The most common methods are related to the CRUD operations: POST (create), GET (read), PUT (update), and DELETE. The method tells the API what you want done, and each endpoint expects a certain method.

Another component used to call an API is a request header. A request header is an HTTP header. It provides additional information about the context of the request. For example, a request header might indicate the preferred language to receive the response. Most APIs also require authentication headers, which provide authentication information to the client like a personal key that shows the person using the API is legitimate.

Depending on the method, you may need to define additional data in the body of the request. For example, if you are creating something using a POST method, there might be data fields used on creation that you need to input.

For example, let's consider the user registering for an awards account. The user fills out certain fields on your website to create their account. In the backend, an API is invoked, and the details the user entered are used in the body of the request:

```
{
"first_name": "Kris",
"last_name": "Owner",
"email": "krisowner@email.com",
}
```

Calling an API

Now that you know what makes up an API request, how do you actually call one? To instantly test an API and see the response, you might send the request via curl or an application such as Postman.

For example, let's say you wanted to retrieve the details for a specific user account. The response may look something like the following:

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "comment": "",
  "created_at": "2020-04-27T19:40:49+00:00",
  "deleted_at": null,
  "customer_id": "x4xCwxxJxGCx123Rx5xTx",
  "first_name": "Kris",
  "last_name": "Owner",
  "email": "krisowner@email.com",
}
```

The response returned depends on the API and the request you made. Let's break this down. The first line contains the status, which in this case is 200 for success. In the next line, the Content-Type header indicates the format of the information you requested, which in this case is JSON. Finally, the body of the response contains the details about the user.

Here, we have demonstrated calling an API for a simple retrieval. You could probably get the same information by logging in via the web interface of your application. The more likely use case for calling an API is incorporating the request into your application and doing something with the data you receive in the response.

How to work with the Fastly API

The Fastly API is a RESTful API that provides access to all the features available through the Fastly web interface.

By using the API, you can work with the objects related to Fastly services and accounts in the way you choose, whether that’s integrating with your existing workflows or automating oft-repeated or cumbersome processes. For example, you might use Fastly’s real-time analytics API to integrate Fastly analytics into your custom analytics dashboard. Or, you may choose to set up an automated purging process. The possibilities are limited only by your imagination (and your programming skills!).

The Fastly API provides a variety of endpoints that we document in our API reference documentation. The endpoint documentation for each API call shows the method, path, authentication type, resource, and parameters that must be combined with the base URL to form a request. Fastly also provides several *client libraries* to choose from to make and process your API requests and responses. A client library makes it possible for you to use a specific programming language to make and process API requests and responses, thereby reducing the amount of code you need to write.

Best of all, the Fastly API is free to use! In order to start using our API, you will need a Fastly account (you can sign up and test up to $50 worth of traffic for free) and an API token. You can create an API token in the Fastly web interface. It's used to authenticate your identity so we know the API request is coming from a legitimate source. Check out our API reference for more information on getting started.

What's next

The API reference guide on Developer Hub has everything you need to get started using the Fastly API. For a complete index of all API endpoints provided by Fastly, refer to this page.

Have you signed up for a Fastly account? There's no obligation and you can test up to $50 of traffic for free.

* * *

What is caching?

/en/guides/what-is-caching

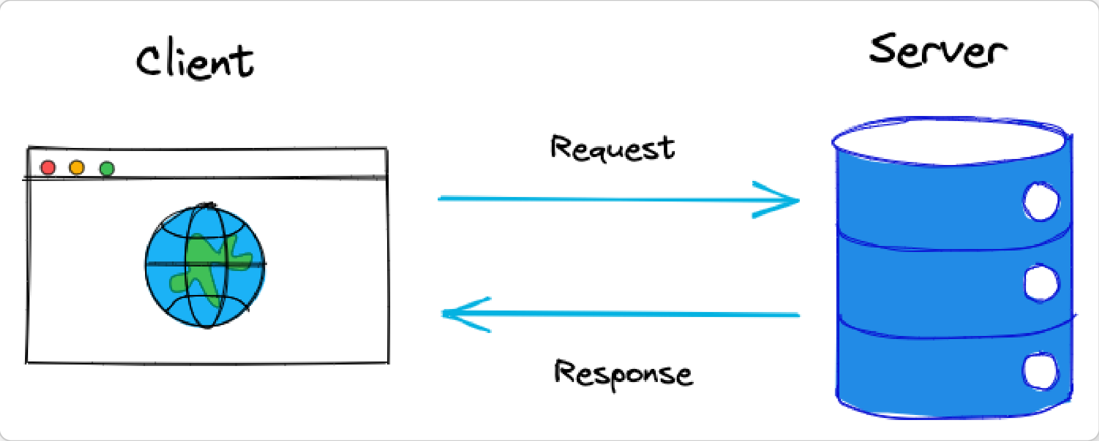
A cache is a location that temporarily stores data for faster retrieval by the things that need to access it. Caching refers to the process of storing this data. You may have heard the term cache used in non-technical contexts. For example, campers and hikers may cache a food supply along a trail. Just think of a cache as a place to store something for convenience.

In a perfect world, a user visiting your website would have local access to all data assets needed to render the website. However, those assets are located elsewhere, like in a data center or cloud service, and there is a cost to retrieving those assets. The user pays in latency, the time required to generate and load the assets. You as the site owner pay a monetary cost not only to host your content but to move that content out of storage when requested, incurring what's known as egress costs. And the more traffic your website receives, the more you have to pay to respond to those requests, some of which are the same.

Caching lets you store copies of your content to speed up delivery of that content while expending fewer resources.

How caching works

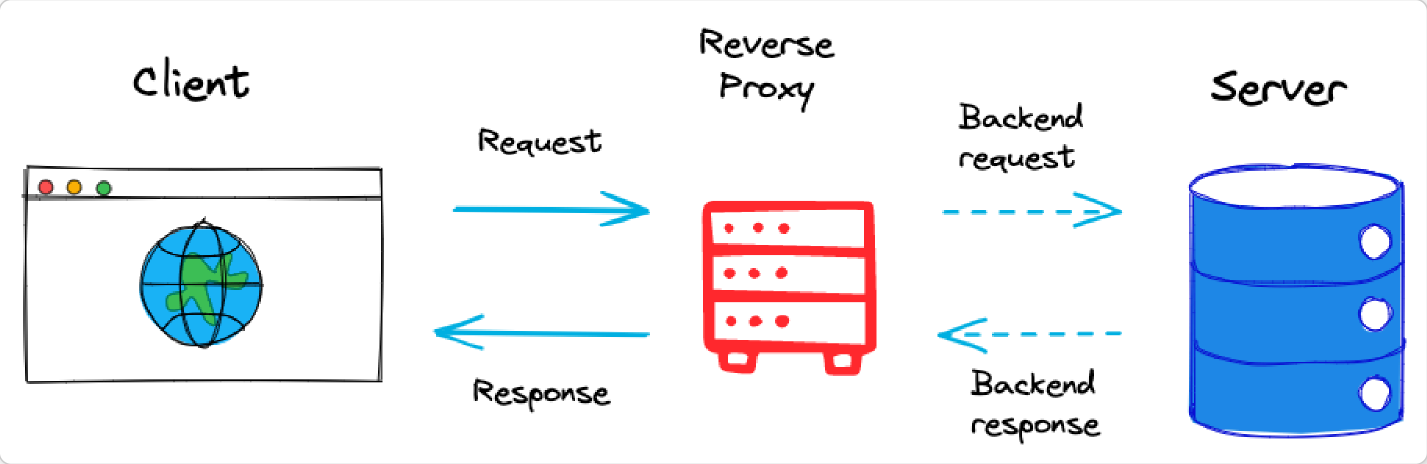
To understand how caching works, we need to understand how HTTP requests and responses work. When a user enters the URL to your website in their browser, a *request* is sent from their browser (the *client*) to the location where the content is located (your origin server). Your origin server processes the request and sends a *response* to the client.



While it may seem like this happens instantaneously, in reality it takes time for the origin server to process the request, generate the response, and send the response to the client. Additionally, many different requests are required to render your website, including requests for all the different data that make up your site (images, HTML web pages, and CSS files are just a few examples).

This is where a *local cache* comes into play. A user’s browser may cache some static assets, like the header image with your logo and the site's stylesheet, on the user’s personal computer to help improve performance the next time they access the site. However, this type of cache is only helpful to one user.

To cache assets for all users visiting your site, you might consider a *remote cache* like a reverse proxy. A *reverse proxy* is an application that sits between the client and your origin servers and receives and responds to requests on your behalf. You can install a reverse proxy on the same origin hosting your website or on a different server. The reverse proxy caches content from your origins so that subsequent requests can be served from cache.



HTTP headers are used to pass information between the client and origin server on the request and the response. You can set certain HTTP headers on the response to control which content is cached and for how long. When using a reverse proxy, the reverse proxy acts as an intermediary, stripping certain headers sent by the origin server and adding other headers to send to the client, all for the purpose of controlling how objects are cached. For more information about controlling how long to cache your resources, start with our documentation on cache freshness.

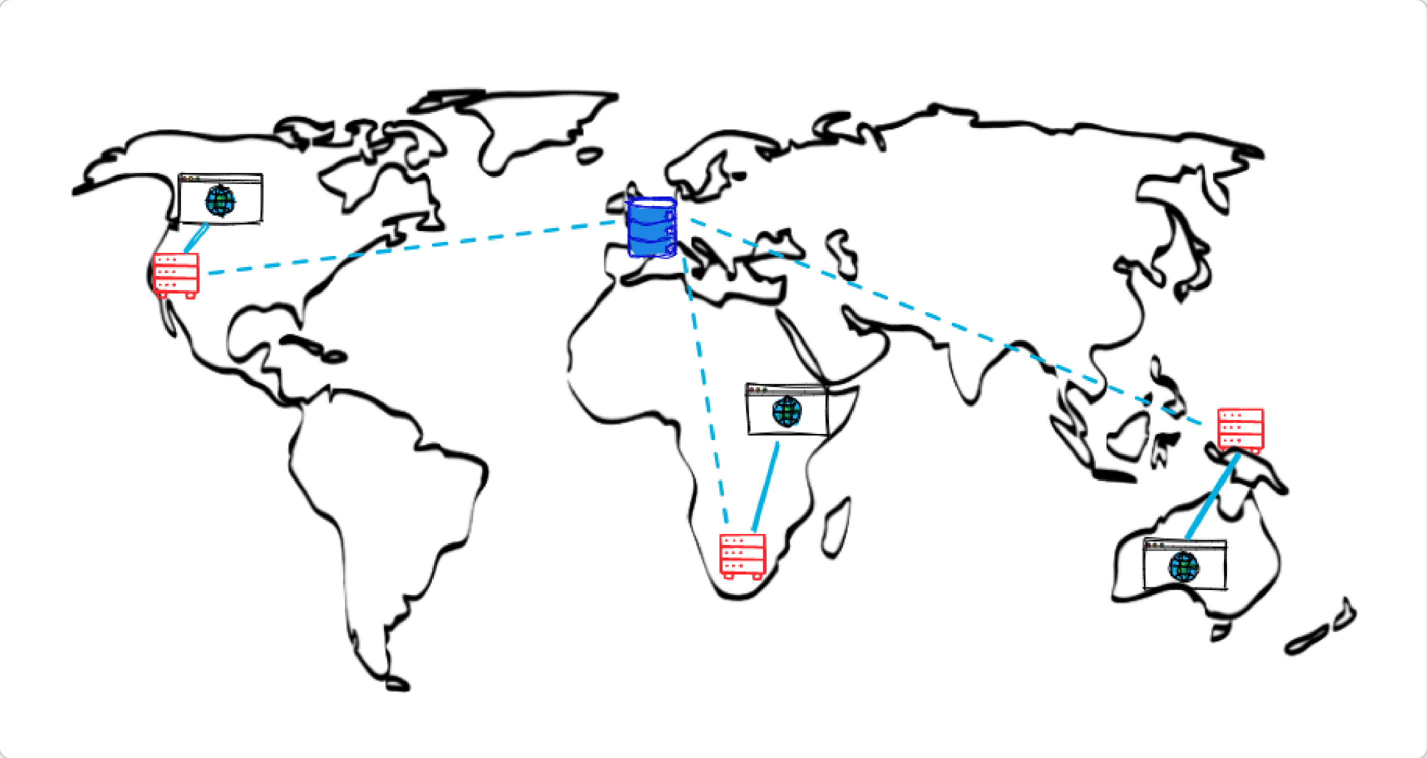
Benefits of caching with a reverse proxy

Remote caching via a reverse proxy lets you store a copy of assets so that subsequent requests from any user can be delivered immediately without having to wait for them to be generated. Caching an already-generated asset means that a request for that asset can be responded to immediately without your origin server needing to do any extra work. This creates a faster experience for your users, and it saves you money because you won't need to pay for that traffic to your origin. Your origin will still need to handle some requests, but not as many. Additionally, a reverse proxy can compress HTTP data before sending the response, which also makes for faster delivery.

Another way a reverse proxy optimizes the response time is by balancing the demand for content. If your website has a surge in popularity and starts getting thousands of concurrent requests, this can overload your origin server and crash your website. By serving content from cache, you can prevent all of that traffic hitting your origin at once. If you have more than one source origin, the reverse proxy can distribute the requests across those servers. In the event your website does go down, users can still access the cached content, so they won't have to experience any downtime.

Why Fastly

A CDN service like Fastly is a reverse proxy but on a greater scale. The time needed to generate and load content isn't the only factor that impacts how quickly a client receives a response to their request. The physical distance between a client and your origin also adds to the response time. A CDN consists of an entire network of cache servers that sit between clients and your origin servers. Each cache server in the network caches content from your origin and responds to requests from clients closest to them.



With Fastly's CDN network, you can efficiently cache and deliver your content because the strategic geographic distribution of our points of presence (POPs) puts our caches closer to your users.

Fastly uses the Varnish Cache reverse proxy as an underlying architecture, which is not only fast but highly customizable. In our guide to CDNs, we talk about the difficulties of caching event-driven content because of its unpredictable nature. However, with Fastly you can cache this type of content and programmatically purge when content changes.

No matter how you're caching content, it's important to monitor how well it's working. In general, you want to see more cache *hits* than cache *misses*. A cache hit means the requested content was found in the cache. A cache miss means the requested content was not found in cache and had to be retrieved from origin. While you can manually keep track of this information, Fastly has a real-time analytics dashboard where you can easily keep track of stats like cache hits, misses, and more.

One of the more useful stats Fastly automatically calculates is *cache hit ratio*. Generally speaking, cache hit ratio (CHR) is the ratio of requests delivered by the cache server (hits) to all requests (hits + misses). You want your cache hit ratio as high as possible because a high cache hit ratio means you've kept request traffic from hitting your origin unnecessarily. By monitoring the real-time stats, you can adjust your cache settings so that you have more hits than misses. You can also monitor several other stats to learn where you're caching traffic, how much of your site is being cached, errors being served by your site, and more.

What's next

Continue reading our essentials guides to learn how CDNs are used to cache at scale and why and how to purge cached content.

Once you're ready to start caching your content, check out our Introduction to Fastly's CDN tutorial for a real-life example for configuring caching that you can use with your own service. We also recommend reviewing our caching configuration best practices. For more information about Fastly's real-time and historical stats and how you can use them to monitor your services, refer to About Observability.

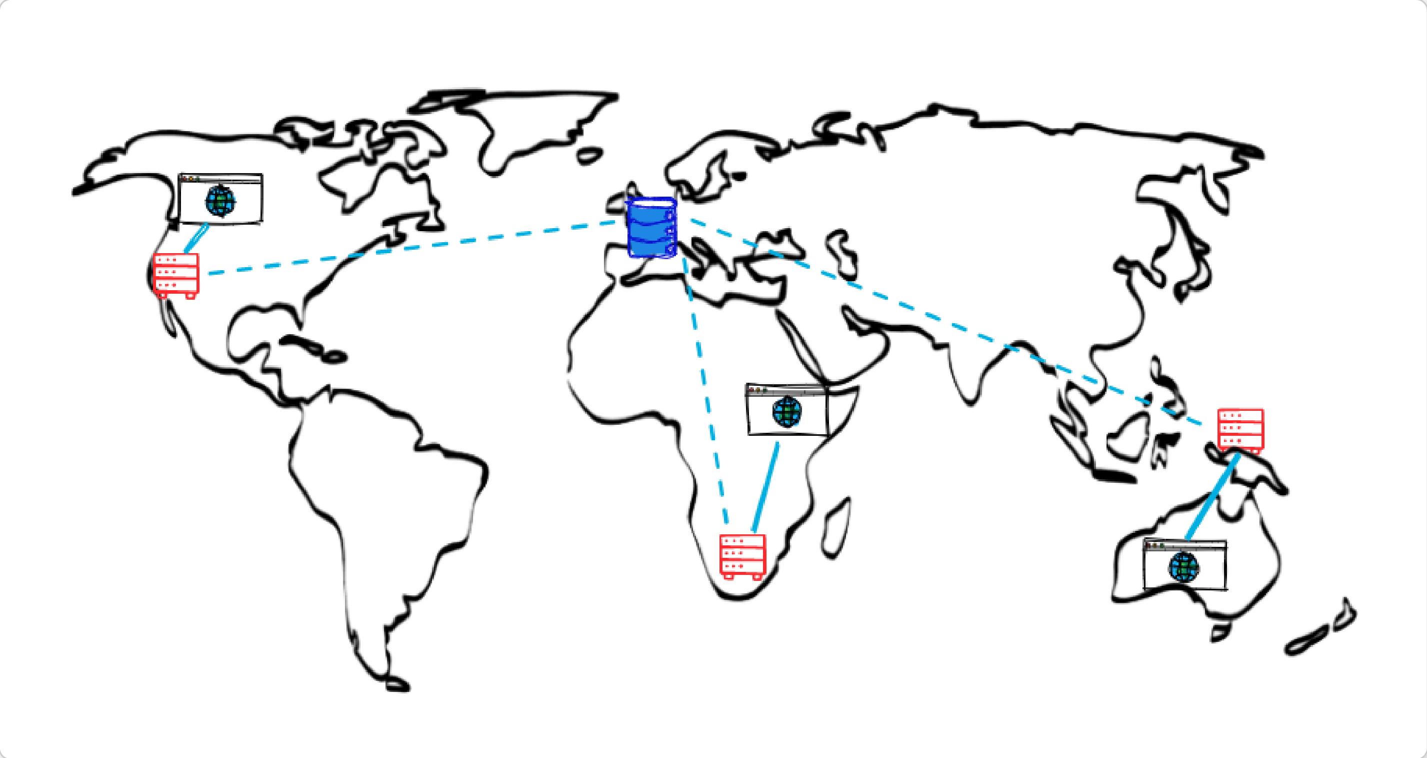
Have you signed up for a Fastly account? There's no obligation and you can test up to $50 of traffic for free.

* * *

What is a CDN?

/en/guides/what-is-cdn

A content delivery network, or CDN, is a network of servers distributed all across the world that work together to make delivering content to your users faster and more efficient.



Before you begin

We recommend reading the What is caching? essentials guide to understand some of the key concepts and terms we'll be using in this guide.

How CDNs work

If you have a website or application, you want to give your users the best experience possible. Part of that great experience is delivering the content your users request quickly. Where your users exist in relation to your content is an important variable that determines how quickly data is delivered and, as a result, how quickly your website or application loads for them.

In our What is caching? guide, we learned that when a user accesses your website, multiple simultaneous requests and responses are sent between the browser and wherever you're hosting your website's data. Depending on how you choose to host your content, the assets users request may exist on your servers in a data center (or hosting facility) or in applications running on cloud services on someone else's servers. It may even exist on a combination of those two. These source servers (frequently referred to as *origins*, *backends*, or *hosts* interchangeably) exist in a specific geographic location and so does your user. However, those two locations aren't always close to each other.

The greater the physical distance between your user and your origin servers, the more time the user will spend waiting for the content - this is called *latency*. The closer the destination (your user) is to the origin (your content), the lower the latency and the faster your website loads.

A CDN helps you lower latency by temporarily storing copies of your content in cache locations close to your user. CDN services have caches all over the globe, so when a user requests your content, the CDN can intelligently deliver the content from a cache located close to the user. This reduces the time to deliver content, thereby making your website load much faster.

TIP

Along with caching content, CDNs can help you update content quickly using a process known as *purging*. You might use purging to remove incorrect or older content from the cache and replace it with the most up-to-date information. You can learn more about purging in our guide to purging.

Benefits of using a CDN service

Besides helping you lower latency and deliver content faster, a CDN provides several other benefits: availability, scalability, and security.

When you use a CDN, you can still serve requests for content even if your origin servers are unavailable. This is because CDNs can serve cached content even if your service is interrupted. Additionally, serving more requests from the CDN instead of your origin means you can lower your *egress* costs, which is the cost hosting providers charge to move content out of storage.

A CDN can also help you lower infrastructure costs by automatically scaling to handle the vast majority of your traffic. You don't want your website to buckle when traffic volumes increase, but you also don't want to spend money on unused servers when traffic isn't at its peak. By using a CDN, you can provision fewer servers and let the CDN automatically scale for high demand.

Not all demand is good, however. A DDoS attack is a common method used by cybercriminals to send a flood of unwanted traffic to your site in an attempt to render it unusable. By using a CDN, you can protect your origin from being hit with that traffic. The CDN can handle the load by helping you block the attack and continuing to route valid traffic throughout the network, ensuring your site stays online.

Unpredictable traffic spikes, security events, and a long list of other factors can make your content slow to render or, worse yet, entirely unavailable. By relying on a CDN instead of a single origin, you can mitigate those risks and save money in the process.

Why Fastly

Fastly's CDN service functions much like a traditional CDN but with added benefits.

Most CDN services are optimized to serve small, static resources very quickly. Things like images, CSS files, and JavaScript files don't have logic that needs to run on a server and are often stored in a file system. For example, your website might have certain branding elements like a header image with a logo and a universal stylesheet. This type of content, which is referred to as *static content*, is easy to cache because it doesn't change very often.

Event-driven content, on the other hand, is harder to cache because it changes at unpredictable intervals. Event-driven content includes things like sports scores, weather forecasts, breaking news, and current store item inventory. Having to serve event-driven content from your origin is taxing on your server and costly. However, with Fastly's architecture and purging capabilities, you can cache event-driven content for a period of time and, when it changes, programmatically purge it. (Typically this can be done as quickly as 150 milliseconds on average!)

Being able to cache both static and event-driven content allows you to deliver a faster user experience and reduce the demand on your origin, which gives you the dual benefits of stability and cost savings.

NOTE

Fastly can also cache video content. Video content includes live video streams and video on demand (VOD) content libraries.

Part of what enables Fastly to cache static and event-driven content is where we deliver our service from. Fastly delivers its CDN service from key access points to the internet called points of presence (POPs). Each POP has a cluster of Fastly cache servers that receive and process requests for your content. Compared to the typical CDN service, Fastly chooses to employ fewer, more powerful POPs in strategic locations enabling you to serve more from cache than origin. Another unique option available to help reduce load on your origin is shielding. When you enable shielding, the shield POP you select will pull content from your origin and distribute it to other cache servers. Requests that can't be served from a POP's cache are directed to the shield POP instead of your origin.

Fastly, like all CDNs, relies on a combination of open source and proprietary software to make our technology work. Fastly uses Varnish as an underlying technology because of its fast speed and because of the flexibility offered by its configuration language, Varnish Configuration Language (VCL). We extend the flexibility of VCL to you by allowing you to customize VCL to generate and compile changes to your Fastly services. These changes can be loaded, activated, and distributed to all Fastly caches worldwide without requiring maintenance windows or service downtime.

What's next

We recommend reading our guide on purging before getting started with Fastly. Getting started with Fastly is as simple as creating an account, connecting your origin server, and adding your domain name. You can do this completely through the web interface or you can use the Fastly API.

Have you signed up for a Fastly account? There's no obligation and you can test up to $50 of traffic for free.

* * *

What is purging?

/en/guides/what-is-purging

Once you have figured out how to cache your content, efficiently invalidating it is another matter. As Phil Karlton aptly described, "There are only two hard things in computer science: cache invalidation and naming things." Cache invalidation, otherwise known as purging or cache purge, is the process of eliminating content objects from cache ahead of when it would naturally expire or be evicted. Once an object is purged, a request for that object will be retrieved from the source rather than the cache.

There are many reasons why you, as a website owner, might want to purge cache. For example, suppose you are an online retailer. You want to make sure the price or quantity of a product is always up to date. This might be a reason to implement a purging policy. Or, let's say you have a news website and the headline or content of an article is no longer valid. You want to purge so that the new title and content will be retrieved from your origin server and stored in cache.

Before you begin

We recommend reading our What is caching? and What is a CDN? essentials guides to understand some of the key concepts and terms we'll be using in this guide.

How purging works

In our What is caching? guide, we learned that HTTP headers transmit information about requests and responses, including how long to keep the requested object in cache. While an object is stored in cache, it has a freshness lifetime, also known as a TTL or *time to live*, and can be served without needing to be revalidated by the origin server. Once the cached object reaches its TTL value, it *expires* and the cache is no longer considered *fresh*. This object may continue to be served *stale* before it's formally evicted and the cache revalidated.

Purging invalidates an object ahead of when it would naturally expire. You may want to purge something because it's incorrect, out of date, or because you have a breaking update. Whatever the case, you are purging to replace what's currently cached on your site with newer, fresh version on your origin.

Let's consider one of the examples we gave earlier. Let's say you publish a news article but notice there's a mistake. You correct the article and push it to your origin servers using your deploy process. However, the incorrect article still remains in cache and can be served to customers until you send a purge request, which you can also do with an HTTP header. Purging informs the cache to stop serving the cached object to client requests and instead retrieve the newer object from origin. This newer object replaces what's in cache and can be served in response to subsequent requests.

Note that this method of purging only works for remote caches you have implemented, like a reverse proxy or CDN. You cannot send purge requests to a user's local cache, like their computer browser, to force it to purge. If you're using a CDN, we recommend setting HTTP headers to prevent objects from being cached in the browser in the first place.

Purging with Fastly

When purging objects, you want to make sure you're purging the right amount of cache at the right time. Purging your entire cache when only a few files are stale is inefficient and expensive to revalidate. With Fastly, you can precisely purge only the objects you want updated. Each of our different purging methods can be initiated via the Fastly web interface or API and are executed on a single service (not across multiple services).

Fastly also gives you the option to execute certain purges as a soft purge or a hard purge. A *soft purge* invalidates an object and marks it as stale. The next time a user requests the object, they'll be served the stale object and then the updated object will be retrieved from origin. Stale objects can also be served to client requests even when your origin servers are down. A *hard purge* permanently invalidates cached objects and makes them unusable for future requests. It forces Fastly to retrieve that object again from your origin servers before it can be re-cached in Fastly POPs.

For the rare situations when you want to purge your entire cache, initiate the Purge all option. By design, Purge all is a hard purge because it invalidates the entire cache in one action.

For more targeted purging, Fastly lets you purge by URL or surrogate key. Purging via URL or by surrogate key is gentler on your origin because it lets you invalidate one object or a targeted set of objects instead of the entire cache. This also saves you money because you're only purging what you need to. When you purge via URL, you specify a single URL path to purge. When you purge by surrogate key, you assign a unique identifier to a set of objects and specify to only purge objects with that identifier. This is helpful for purging groups of related objects. Additionally, both of these options can be run as soft or hard purges.

Why Fastly

Purging with Fastly has benefits for both you and your end users.

The global average time it takes Fastly to purge your content is 150 milliseconds (it's frequently faster!). This lets you cache objects for longer and purge on demand when needed. Plus, you can use an API client or integration to automate the purging process to instantly purge whenever your content changes, giving your users access to the latest and greatest version.

Fastly servers have a *distributed purging system*, which means our cache servers are responsible for distributing purges themselves. This means you can send a purge to the nearest server, and from there it's sent directly to all other cache servers, removing the single point of failure that comes with a centralized system. If there's an outage in a certain area, other cache servers can respond to purge requests.

By caching at the edge and purging in real-time, you can serve more requests from the cache and minimize trips to origin, which reduces the load on your servers. This means you can spend less on servers and other related infrastructure equipment and instead rely on Fastly to store and deliver content.

What's next

You can access Fastly's purging features by using the Fastly web interface or the Fastly API. Our Introduction to Fastly's CDN tutorial has real-life examples for purging your entire cache and purging with keys.

Have you signed up for a Fastly account? There's no obligation and you can test up to $50 of traffic for free.

* * *

Category: Web interface

These articles describe key features of the Fastly web interface controls.

About Account

/en/guides/about-account

Use the Account pages in the web interface to review and manage (depending on the roles and permissions you've been assigned) various aspects of your company's account, as well as your personal account access and its appearance.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Account controls

From the **Account** controls, you can access:

- the **Company settings** page where you can control details that identify your company (e.g., the owner and various company contacts), as well as the location to enable a login IP allowlist, enable company-wide account options (e.g., two-factor authentication), and cancel your account.
- a **User management** page where you can control user invitations and configure their roles.
- an **Account tokens** page where you can view details about active API tokens created by other users within your account for various services.
- an **Audit log** page where you can monitor events related to your account, users, and services.
- the **Single sign-on** page where you can manage user authentication by enabling single sign-on (SSO).

- a **Billing** menu where you can review metrics related to your use of products and features enabled on your account, as well as control how you are charged for them.
- a **Personal Profile** menu where you can change personal details related to your account.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About CDN

/en/guides/about-cdn

The CDN controls are the key area of the web interface where you access your CDN services.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the CDN services page

The CDN services page provides a list of clickable links to all CDN services associated with your account. To access this page, go to **CDN** > **CDN Services**.

From the CDN services page, you can view high-level details about each service including the active version number, the current requests per second, and a chart of the last hour of requests. Use the search box to search for a service by ID, name, or domain.

Clicking a link to a CDN service takes you to the Service summary tab, where you can review other general service details, and gives you access to the Service configuration tab, which contains the main controls for editing your service configurations.

About the Service summary tab

When you select a service from the CDN services page, the **Service summary** tab appears. From here, you can review general service details including the event log, duplicate (clone) service versions so you can edit them, and purge content.

The Service summary tab also allows you to compare differences between two configuration versions.

About the Service configuration tab

The **Service configuration** tab contains the main controls for editing your service configurations. Specifically you can:

- manage the domains used to route requests to a CDN service

- manage the hosts used as backends for a site and how they should be accessed
- manage the health checks that monitor backend hosts
- enable certain Fastly products for the service, if you are assigned the role of superuser or engineer. Note that this will result in changes to your monthly bill.
- manage various request and cache settings, headers, and responses that control how Fastly caches and serves content for a CDN service
- specify how logging should be performed and where server logs should be sent
- create custom Varnish configuration language (VCL) files
- specify how conditions are mapped and used for a service at various times (e.g., during request processing, when Fastly receives a backend response, or just before an object is potentially cached)

With the appropriate permissions, you can activate configuration changes immediately and roll back those changes just as quickly should they not have the intended effect.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About Compute

🔗 /en/guides/about-compute

The Compute controls are the key area of the web interface where you access your Compute services. The Compute controls serve as a web-based alternative to using the Fastly CLI.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Compute services page

The Compute page provides a list of clickable links to all Compute services associated with your account. To access this page, go to **Compute** > **Compute Services**.

From the Compute services page, you can view high-level details about each service including the active version number, the current requests per second, and a chart of the last hour of requests. Use the search box to search for a service by ID, name, or domain.

Clicking a link to a Compute service takes you to the Service summary tab, where you can review other general service details, and gives you access to the Service configuration tab, which contains the main controls for editing your service configurations.

About the Service summary tab

When you select a service from the Compute services page, the **Service summary** tab appears. From here, you can review general service details including the event log, duplicate (clone) service versions so you can edit them, and purge content.

About the Service configuration tab

The **Service configuration** tab contains the main controls for editing your service configurations. The page allows you to define exactly how each instance of your Compute services should behave and interact with its data sources. Specifically you can:

- manage the domains used to route requests to a Compute service
- manage the hosts used as backends for a site and how they should be accessed
- manage the health checks that monitor backend hosts
- enable certain Fastly products for the service, if you are assigned the role of superuser or engineer. Note that this will result in changes to your monthly bill.
- specify how logging should be performed and where server logs should be sent
- upload your Compute compatible Wasm packages once you've built them with the Fastly CLI

About the Learn tab

The **Learn** tab contains specific resources for using Fastly's Compute platform including:

- details that help you select and learn about languages that can be used on Fastly's Compute platform
- steps for using the Fastly CLI and a link to the Fastly CLI reference documentation
- information about tools like Terraform to help you automate your workflows
- a way to experiment with code in the Fastly Fiddle so you can create ephemeral Fastly services without logging into a Fastly account and learn how your requests and responses are handled by Fastly

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About Observability

/en/guides/about-observability

You can use the Observability pages in the web interface to monitor your Fastly CDN and Compute services.

Before you begin

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

About the Observability pages

When you click **Observability** in the navigation sidebar, the Account Summary page appears by default. The Account Summary page provides a high-level overview of all your services.

To access additional Observability pages, go to **Observability** in the navigation sidebar and select one of the following options:

- **Services menu**: provides access to the Service Overview page, which displays top metrics for a Fastly service, and the All Services page, which displays aggregated metrics for all of your services.
- **Domains page:** provides real-time and historic visibility into detailed response data for traffic from your domains and subdomains to Fastly.
- **Origins page:** provides real-time and historic visibility into detailed response data for traffic from your origin servers to Fastly.
- **Logs menu:** provides access to the Log tailing page, where you can view custom log messages from your Compute application.
- **Alerts menu:** provides access to the Overview, Definitions, and Integrations pages, where you can view current and historical alert feeds and configure alert definitions and integrations.
- **Custom Dashboards page:** where you can view and manage custom dashboards that surface relevant metrics for a Fastly service.

The exact graphs displayed on the Observability pages depend on your access permissions and the type of service you create, as well as other products and features you may have purchased.

The data on the Observability pages may also appear grayed out or blank to some users, with no information displayed in the controls, when a service hasn't yet received enough requests for Fastly to display meaningful information about it.

About the controls on the Observability pages

On the Account Summary, Service Overview, All Services, Origins, Domains, and Custom Dashboards pages, a series of controls appear to the right of and below the dashboard name. These controls allow you to refine the displayed metrics on the dashboard. Specifically:

- the **Alerts** link, which opens the Alerts firing window. From this window, you can view firing alerts.
- the **Dashboard Name** menu (which only appears on dashboards for specific services, not the Account Summary dashboard) allows you to select and view a different dashboard.
- the **Options** menu allows you to copy a link to the dashboard, export the data from the dashboard to a CSV file, and display event markers on the charts. Event markers are vertical lines that signify when a new version of the service was activated.
- the **Service configuration** link opens the service configuration page for the service selected in the dashboard.
- the **Create dashboard** button allows you to create a custom dashboard.

- the **Service** menu allows you to specify the service that the metrics are drawn from. From the menu, you can search for a service by ID or name.
- the **Region** menu allows you to limit the displayed metrics on any dashboard to a specific region around the world.
- the **Data Resolution** menu allows you to select how the data is represented for the defined time range.
- the **Time Range** menu allows you to change the timeframe over which metrics will be displayed and to change the timezone from Coordinated Universal Time (UTC) to the timezone identified by your browser. By default, the system displays 1 days worth of static data. The selected timeframe and timezone persist as you navigate between dashboards.
- the **Live** button (which only appears on dashboards for specific services, not the Account Summary dashboard) allows you to change displayed metrics to live data that changes in real time. When viewing the live data stream, the time axis will scroll even if there is no data. You can click **Pause** to pause the live data stream.

You can also learn more information about a chart by:

- hovering over any part of a graph. This displays a timestamp indicator that updates as you move the cursor along with a service version indicator.
- using the summary menu at the bottom of a chart to view the total, maximum, minimum, mean, latest, and 95th percentile values of a graph. By default, percentile charts display the mean value while other charts display the total. The summary menu for percentile charts does not include the total metric.

In addition, on dashboards that aren't custom, you can click expand in the upper right corner of a chart to access the Metric Details page, which contains detailed information about the metric being tracked. Specifically, the Metric Details page displays:

- the expanded chart.
- a summary table that automatically updates the total, maximum, minimum, mean, and 95th percentile values for the metric based on the timestamp that you're hovering over on the expanded chart.
- a table that displays the total for the metric over time.

About the Account Summary page

The **Account Summary** page provides aggregate metrics for your account for the last two weeks:

- **Requests:** the total number of CDN and Compute requests aggregated across your account.
- **Bandwidth:** the total amount of bandwidth served across all services in your account.
- **Error ratio:** the average ratio of error responses (4xx and 5xx) to the total number of requests across your account.

Links to your latest alerts and custom dashboards are listed at the bottom of the page. Alerts and custom dashboards are disabled by default.

About data latency

When working with dashboards and the metrics they display, keep the following in mind about data:

- We do not report data for time periods that have not yet ended.

- Data with `day` resolution is bucketed based on UTC days and each day's data becomes available around 2am the following day.
- Data with `hour` resolution buckets becomes available approximately 15 minutes after the end of each hour.
- Data with `minute` resolution buckets usually becomes available two minutes after the end of the minute, but can take up to 15 minutes to appear.

If your use case requires data closer to real-time, consider using the generally available real-time API instead.

> IMPORTANT
>
> You cannot retrieve minutely historical statistics data older than 35 days from the current date. You cannot retrieve hourly historical statistics data older than 375 days from the current date. Contact support to discuss your minutely or hourly data needs.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About Resources

/en/guides/about-resources

Resources are versionless data stores that can be shared across Compute services. To access pages for config store, KV store, and secret store go to **Resources** in the navigation sidebar.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Config stores page

From the **Config stores** page, you can view individual cards for each config store that detail the key-value pairs and linked services. Search for a specific store by typing the store name in the **Search config stores** field.

Click **Key-value pairs** to view the following information about the keys within the store:

- **Key:** the name of the key.
- **Value:** the value associated with the key.
- **Date Added:** the date the key-value pair was created.

Click **Services** to view the following information about the services linked to the store:

- **Service:** the name of the service linked to the store.

- **Type:** the type of service, either Deliver or Compute.
- **Linked versions:** the version number of the last active and last draft version of the service linked to the store.

About the KV stores page

From the **KV stores** page, you can view individual cards for each KV store that detail the key-value pairs and linked services. Search for a specific store by typing the store name in the **Search KV stores** field.

Expand **Key-value pairs** to view the keys within the store. By default, the web interface displays only the first 1,000 keys in the store. To view more keys, click **Load more keys**.

Expand **Services** to view the following information about the services linked to the store:

- **Service:** the name of the service linked to the store.
- **Type:** the type of service, either Deliver or Compute.
- **Linked versions:** the version number of the last active and last draft version of the service linked to the store.

About the Secret stores page

From the **Secret stores** page, you can view individual cards for each secret store that detail the key-value pairs and linked services. Search for a specific store by typing the store name in the **Search secret stores** field.

Click **Key-value pairs** to view the following information about the keys within the store:

- **Key:** the name of the key.
- **Value:** the value associated with the key.
- **Date Added:** the date the key-value pair was created.

Click Services to view the following information about the services linked to the store:

- **Service:** the name of the service linked to the store.
- **Type:** the type of service, either Deliver or Compute.
- **Linked versions:** the version number of the last active and last draft version of the service linked to the store.

What’s next

Explore our edge data storage options to find out which best fits your needs.

---

* * *

About Security

/en/guides/about-security

The Security controls give you access to the different security products Fastly has to offer. If you have the appropriate access permissions, these controls appear when you go to **Security** in the navigation sidebar.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Security controls

From the Security controls, you can access the:

- **Overview** page, which provides introductions to various security offerings that can help you protect your services and links you to more information about them. If you have a Signal Sciences account, you can log in to it directly from this page.
- **TLS management** menu, where you can find controls for adding HTTPS to your domains, adding additional privacy and data security to your services. You can either upload your own TLS certificates or have Fastly manage this for you.
- **Edge rate limiting** page, where you can implement a rate limiting policy to control the rate of requests sent to your origin servers.
- **DDoS mitigation** page, where you can learn more about our DDoS mitigation options. Fastly's high-bandwidth globally distributed network was built with "always-on" DDoS mitigation designed to absorb DDoS attacks. You can also purchase a DDoS Protection and Mitigation Service that provides additional Fastly resources to assist you with mitigating the service and financial impacts of DDoS and related attacks.
- **Bot management** page, where you can review potential abusive bot activity on your network.
- **Next-Gen WAF** page, where you can log in to your Signal Sciences account if you have one or contact us for information on the Fastly Next-Gen WAF (powered by Signal Sciences). You can use a WAF to protect your applications from malicious attacks designed to compromise web servers.

Security products note

No security product, such as a WAF or DDoS mitigation product, including those security services offered by Fastly, will detect or prevent all possible attacks or threats. As a subscriber, you should maintain appropriate security controls on all web applications and origins. The use of Fastly's security products do not relieve you of this obligation. As a subscriber, you should test and validate the effectiveness of Fastly's security services to the extent possible prior to deploying these services in production, continuously monitor their performance, and adjust these services as appropriate to address changes in your web applications, origin services, and configurations of the other aspects of your Fastly services.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Alerts menu

/en/guides/about-the-alerts-menu

To access the pages associated with the Alerts feature, go to **Observability** > **Alerts**. With the Alerts feature, you can receive automated notifications via various integrations when a metric you define either goes above or remains below a threshold.

The Alerts menu contains links to the following pages:

- **Overview:** where you can view a historical feed of alerts that are firing and that have been resolved.
- **Definitions:** where you can view and manage alert definitions. An alert definition is a configuration rule that defines when an alert should be triggered and how notifications should be handled.
- **Integrations:** where you can view and manage alert integrations. Alert integrations are external channels that receive notifications when alerts are triggered.

Before you begin

The Alerts feature is disabled by default and can be purchased by contacting sales@fastly.com. Once enabled, any user with the appropriate roles and permissions will be able to manage alerts for Fastly services and historic stats metrics.

Be sure you know how to access the web interface controls before learning about the information you'll encounter here.

About the Overview page

The **Overview** page contains a historical feed of alerts that are firing and that have been resolved. From this page, you can:

- view alerts that are firing by clicking the Alerts bell.
- filter the alert instances by service, alert definition name, and the time range when the alert was triggered or resolved.
- monitor alerts that are firing. An alert is firing when the condition set is no longer met during an evaluation period. Read about how alerts are triggered and resolved.
- view alerts with a Resolved status. An alert that was firing is considered resolved when the defined threshold is no longer being exceeded.
- click on a firing or resolved alert instance to access the details page for the alert. The alert details page displays a graph of the observed metric. On the graph, a horizontal black line marks the threshold, vertical red bars indicate when the metric went above or below the threshold, and vertical black lines indicate when a new service version was activated. The page also lists all firing and resolved instances of the alert. From the details page, you can edit the alert definition.

About the Definitions page

The **Definitions** page lists your alert definitions. From this page, you can:

- view alerts that are firing by clicking the Alerts bell.
- view a list of your alert definitions. You can use the **Service** menu to filter the list.
- create alert definitions.
- edit and delete existing alert definitions.

About the Integrations page

The **Integrations** page lists your alert integrations. From this page, you can:

- view alerts that are firing by clicking the Alerts bell.
- filter the alert integrations by type (e.g., Slack).
- create alert integrations.
- delete alert integrations.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Billing menu

🔗 /en/guides/about-the-billing-menu

To access billing-related detail for your account, go to **Account** > **Billing**. This menu provides options that allow you to review and manage things related to the charges associated with products and features enabled for your account.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Billing menu

From the Billing menu, you can access:

- the **billing Overview** page where you can gain insight on past bills and usage related to bill-related metrics for your account broken down by product and region over key, monthly timeframes.
- the **Plan usage** page where you can view your account plans and usage details.
- the **Trials** page where you can view details about product or feature trials.

- the **Invoices** page where you can view a complete history of the monthly bills for your Fastly account and their payment statuses, which you can use to review your specific account charges that are active or can be started.
- the **Billing information** page where you can upgrade to a paid account from a trial account. Once on a paid account, you can view and edit your credit card information and update your billing address for your account, if you have the appropriate permissions to do so.

> NOTE
>
> Billing plan information is only available for Delivery customers.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the billing overview page

/en/guides/about-the-billing-overview-page

The billing overview page allows you to view bill-related metrics for your account broken down by product and region over key, monthly timeframes. These metrics provide insights into what you're being charged for by Fastly and how that amount applies to the various products you've purchased and the regions in which traffic to your services operates. To access the billing overview page, select **Billing** from the user quick links menu.

Before you begin

Be sure you know how to access the web interface controls before learning about the information you'll encounter here.

About the billing overview page

When your user role allows you access to this page, a series of graphs appear:

- the **Total bill** graph, which displays the total US dollar amount that Fastly has billed your account over a specific time period
- the **Full-Site Delivery bill** graph, which displays the total US dollar amount your account has been billed for the Fastly Full-Site Delivery portion of your account's bill over a specific time period
- the **Compute bill** graph, which displays the total US dollar amount your account has been billed for Compute platform operations over a specific time period
- the **Bandwidth** graph, which displays the total bandwidth in gigabytes that has been processed for your account over a specific time period
- the **Requests** graph, which displays the total number of requests processed for your account over a specific time period

- the **Other services & products** graph, which displays the total US dollar amount that Fastly has billed your account for all Fastly products and services, not including Full-Site Delivery and Compute, over a specific time period
- the **Regional bills** graph, which displays how much you were billed for Fastly products and services within each geographical region of the world, excluding non-regional offerings, over a specific time period

About the billing graph data

When reviewing the metrics displayed on the billing graphs, keep the following things in mind:

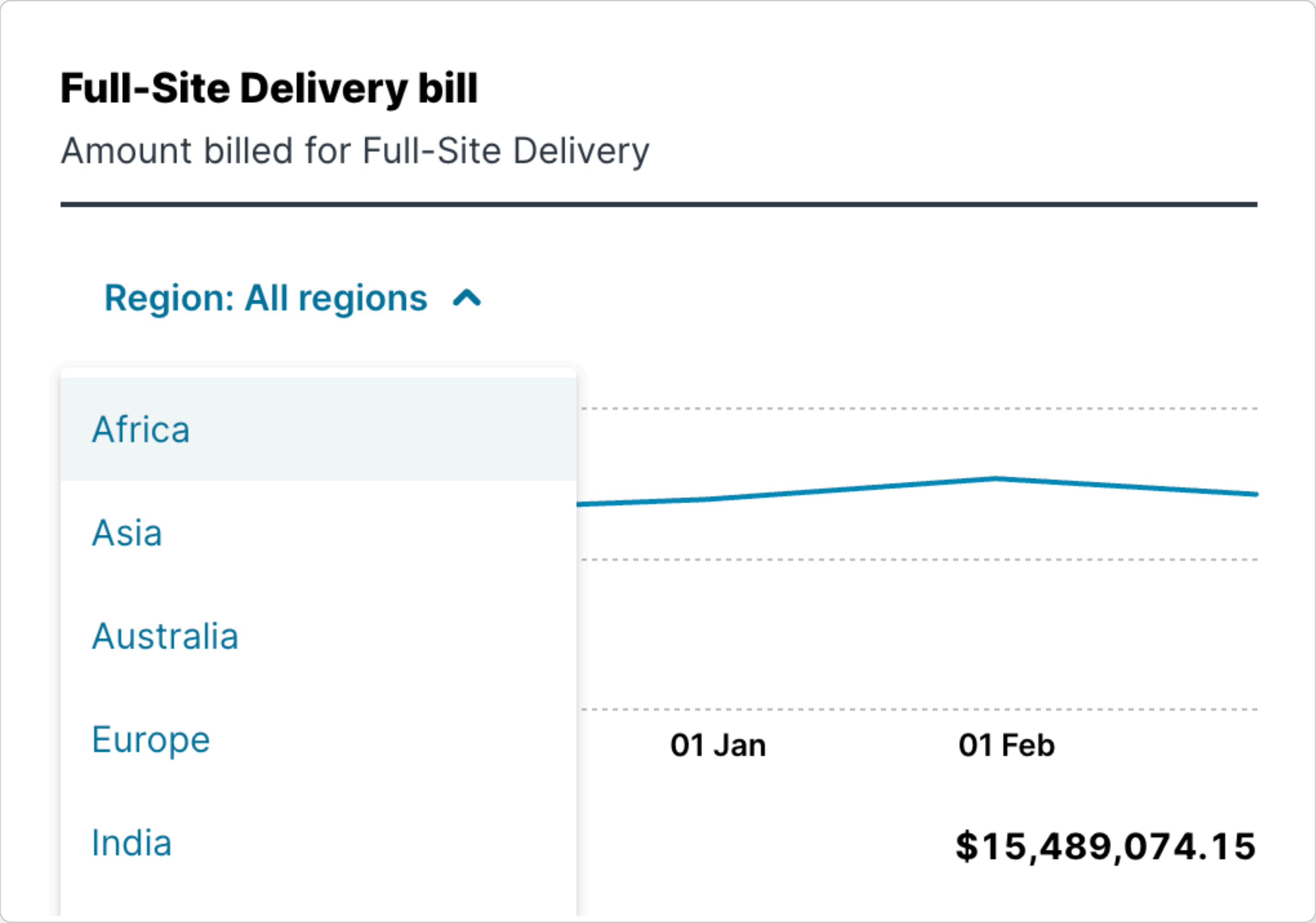
- **Data doesn't appear in real time.** Unlike the metrics displayed in the Observability page graphs, data for billing metrics does not appear in real-time nor for time periods shorter than one month.
- **Dollar amounts shown are for US dollars.** For any graph displaying monetary amounts, these amounts appear in US dollars only.

Working with the billing graphs

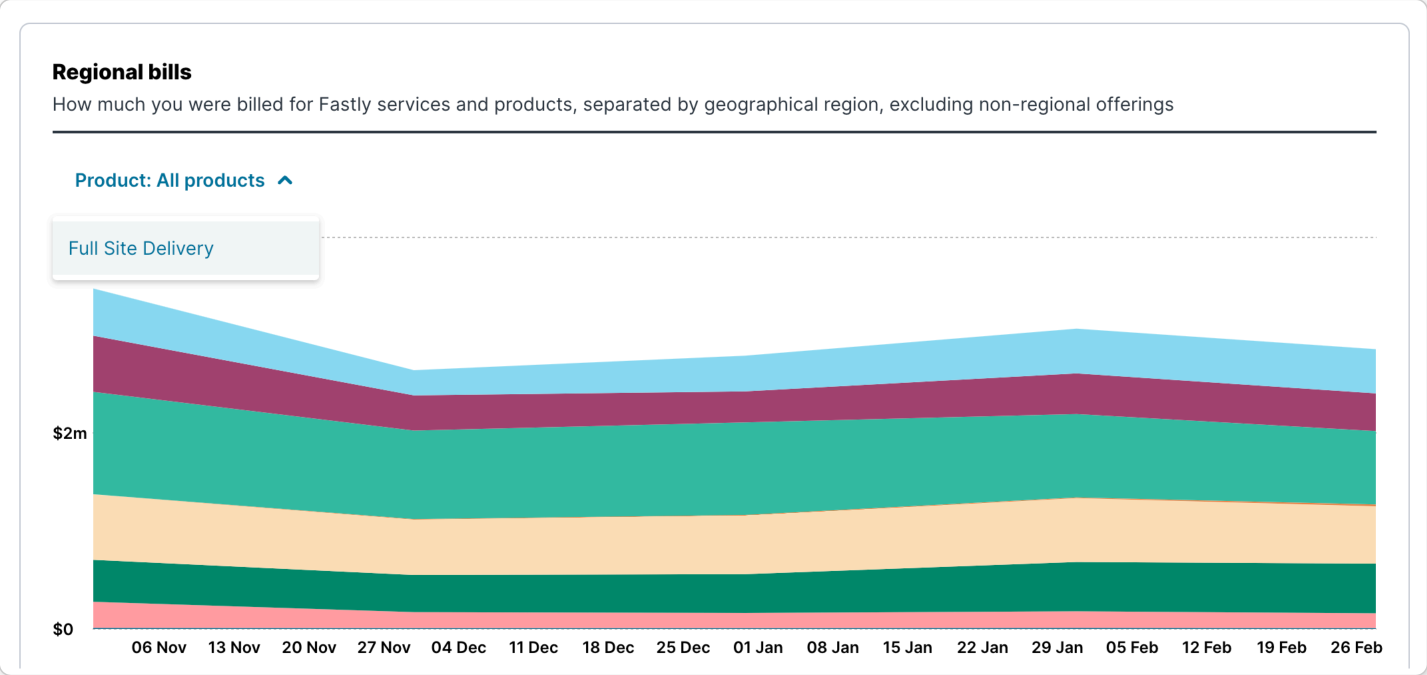
You can interact with and control the billing graphs as follows.

Limiting data to specific geographic regions or products

By default any graph that displays metrics for regional offerings displays data for all regions. You can change these to display data for specific regions by selecting a specific region from the **Region** menu near the top of each graph.

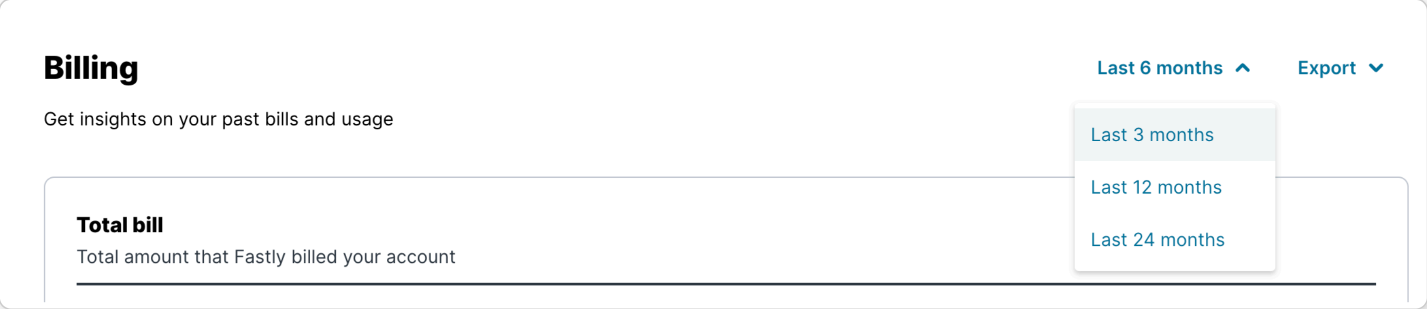


Likewise, the **Regional bills** graph displays metrics for all products you currently pay for. You can change this to display data for a specific offering by selecting it from the **Product** menu near the top of the graph.


Regional bills
How much you were billed for Fastly services and products, separated by geographical region, excluding non-regional offerings
Product: All products
Full Site Delivery
$2m
$0
06 Nov
13 Nov
20 Nov
27 Nov
04 Dec
11 Dec
18 Dec
25 Dec
01 Jan
08 Jan
15 Jan
22 Jan
29 Jan
05 Feb
12 Feb
19 Feb
26 Feb

Changing displayed time periods

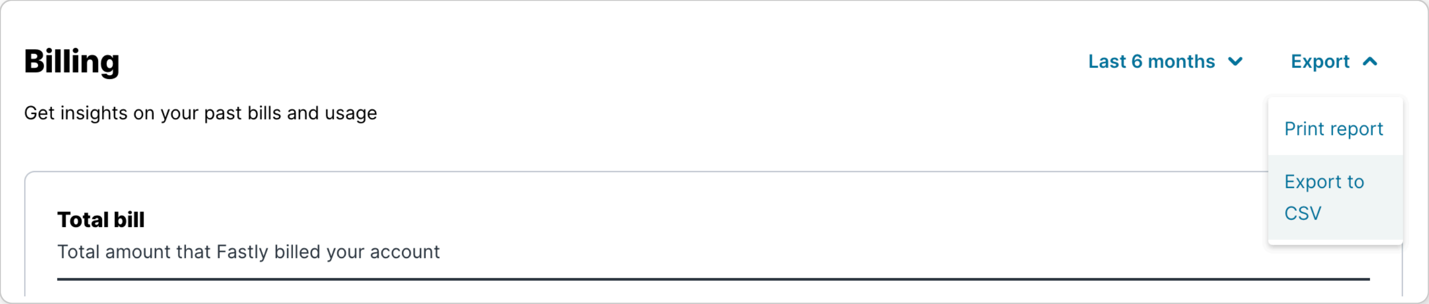
By default all graphs display billing data for the past six months. You can change this by selecting a new time frame from the month control at the top of the page to a maximum of 24 months.


Billing
Get insights on your past bills and usage
Last 6 months
Export
Last 3 months
Last 12 months
Last 24 months
Total bill
Total amount that Fastly billed your account

Printing and exporting billing data

You can print a report of your billing data by selecting **Print** from the **Export** menu at the top of the page. This will launch the printing controls for your browser and allow you to send the report to a printer or create a PDF version of the data for the time period you've selected.

You can create an exportable version of your billing data by selecting **Export to CSV** from the **Export** menu at the top of the page. This will automatically download a CSV-formatted text file to a local directory.



Billing
Get insights on your past bills and usage
Last 6 months
Export
Print report
Export to CSV
Total bill
Total amount that Fastly billed your account

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Domains page

/en/guides/about-the-domains-page

The Domains page gives you access to the Domain Inspector dashboard, which provides real-time and historic visibility into response data for traffic from your domains and subdomains to a Fastly Deliver service. To access the Domains page, go to **Observability** > **Domains**.

Before you begin

Domain Inspector is disabled by default. Anyone assigned the role of superuser can enable a 30-day trial directly in the web interface. After the trial expires, Domain Inspector can be purchased for an account by contacting sales@fastly.com. Once purchased, it can be enabled in the web interface by anyone assigned the role of superuser or engineer, which will result in changes to your monthly bill. When Domain Inspector is enabled, any user will be able to view the real-time and historic stats specific to Domain Inspector.

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

Enabling and disabling Domain Inspector

Domain Inspector can be enabled and disabled directly in the web interface by anyone assigned the role of superuser.

Enabling Domain Inspector

To enable Domain Inspector for a service:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Domains**.
3. Click the **Monitor domain responses** switch to the **On** position to enable Domain Inspector for the service.

Once Domain Inspector is enabled, you can start viewing real-time metrics about your domains immediately. Historic metrics usually become available two minutes after the end of each minute, but can take up to 15 minutes to appear.

Disabling Domain Inspector

To disable Domain Inspector for a service:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Domains**.
3. Click the **Monitor domain responses** switch to the **Off** position to disable Domain Inspector for the service.

About the Domain Inspector dashboard

Once Domain Inspector is enabled for your account, the **Domain Inspector** dashboard displays the following metrics when you select **All domains** from the **Domain** menu:

- **Edge Requests by Domain:** the number of requests processed at the edge by each domain.
- **Response Body Bytes by Domain:** the number of response body bytes delivered from the edge by each domain.
- **Response Header Bytes by Domain:** the number of response header bytes delivered from the edge by each domain.
- **2xx Status Codes by Domain:** the number of 2xx type (success) HTTP response codes delivered by each domain.
- **3xx Status Codes by Domain:** the number of 3xx type (redirection) HTTP response status codes delivered by each domain.
- **4xx Status Codes by Domain:** the number of 4xx type (client error) HTTP response status codes delivered by each domain.
- **5xx Status Codes by Domain:** the number of 5xx type (server error) HTTP response status codes delivered by each domain.

When you select any single domain from the **Domain** menu, the **Domain Inspector** dashboard displays the following metrics:

- **Edge Hit Ratio:** the percentage of cache hits to all cacheable content over time.
- **Edge Hit Ratio Percentage:** the percentage of cache hits to all cacheable content.
- **Edge Response Status Codes:** the number of response status codes delivered from the edge.
- **Origin Response Status Codes:** the number of status response codes from the domain's origin.
- **Edge Status 5xx Details:** the distribution of 5xx (server) status codes delivered from the edge.
- **Origin Status 5xx Details:** the distribution of 5xx (server) status codes delivered from the domain's origin.
- **Edge Status 4xx Details:** the distribution of 4xx (server) status codes delivered from the edge.
- **Origin Status 4xx Details:** the distribution of 4xx (server) status codes delivered from the domain's origin.
- **Edge Status 3xx Details:** the distribution of 3xx (server) status codes delivered from the edge.

- **Origin Status 3xx Details:** the distribution of 3xx (server) status codes delivered from the domain's origin.
- **Edge Status 2xx Details:** the distribution of 2xx (server) status codes delivered from the edge.
- **Origin Status 2xx Details:** the distribution of 2xx (server) status codes delivered from the domain's origin.
- **Edge Requests:** the number of requests received at the edge.
- **Origin Requests:** the number of origin requests fetched for a domain.
- **Bandwidth:** total number of bytes delivered for a domain.
- **Origin Offload Percentage:** the percentage of content delivered from the edge compared to what was delivered from your origin for a domain.
- **Origin Offload:** the average amount of content, in bytes served, from the edge.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

---

* * *

About the Home page

/en/guides/about-the-home-page

The Home page displays summary information about all services associated with your account. Think of it as a dashboard where you can get quick insights and basic information about your services along with some aggregate, high-level metrics about them. You can use the dashboard to view who's a user on your team and edit their roles or even invite more users to your account. The dashboard also provides links to helpful resources you may need as you navigate your way around the Fastly app.

Before you begin

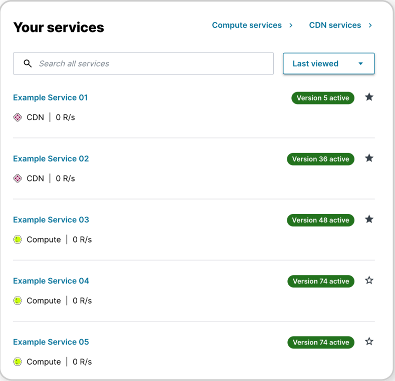
Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Home page

The Home page appears automatically when users with the appropriate access permissions log in to the Fastly web interface. You can access it by clicking **Home** in the navigation sidebar.

About the Your services card

The **Your services** card displays the services associated with your account and defaulting to those you've designated as your favorites. Each service listed includes a clickable link to the configuration details for that service and a summary of key information about it, including the type of service (CDN or Compute), the current requests per second for it, and the most recently activated service version.

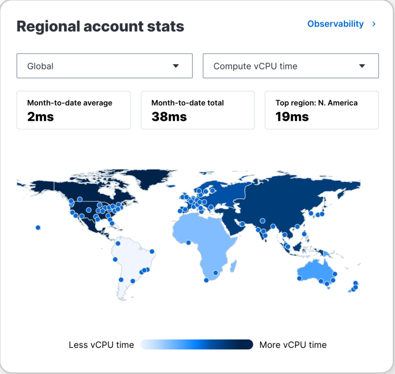

Your services
Compute services ›
CDN services ›
Search all services
Last viewed
Example Service 01
Version 5 active
CDN | 0 R/s
Example Service 02
Version 36 active
CDN | 0 R/s
Example Service 03
Version 48 active
Compute | 0 R/s
Example Service 04
Version 74 active
Compute | 0 R/s
Example Service 05
Version 74 active
Compute | 0 R/s

You can change the displayed list of services on the card by selecting a different sort order from the menu to the right of the search field. Other filtering options besides the default **Favorites** include:

- most recently viewed
- most recently activated
- highest number of requests per second
- most recently created

Clicking the **Compute services** and **CDN services** links that appear to the right of the card's title will take you to the **Compute** and **Deliver** pages respectively where you can view all of the services of that type associated with your account and access the main controls for editing their configuration.

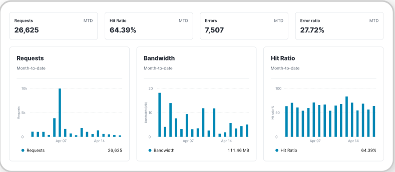
About the Regional account stats card


Regional account stats
Observability ›
Global
Compute vCPU time
Month-to-date average
2ms
Month-to-date total
38ms
Top region: N. America
19ms
Less vCPU time
More vCPU time

The **Regional account stats** card displays summarized, basic metrics and a map of parts of the world to help you visualize their source. Clicking **Observability** to the right of the card title takes you to the **Observability page** where you can access additional stats details for all your services.

By default, the map and summary stats displayed represent metrics globally around the world. You can select a smaller area of the world by selecting that area from the left most menu immediately below the card title. Likewise, you can select a variety of other stats to display by selecting one from the right most menu immediately below the card title.

About the aggregate account-level stats cards


Requests MTD
26,625
Hit Ratio MTD
64.39%
Errors MTD
7,507
Error ratio MTD
27.72%
Requests
Month-to-date
10k
5k
0
Requests
Apr 07
Apr 14
Requests 26,625
Bandwidth
Month-to-date
20
10
0
Bandwidth (MB)
Apr 07
Apr 14
Bandwidth 111.46 MB
Hit Ratio
Month-to-date
100
50
0
Hit ratio %
Apr 07
Apr 14
Hit Ratio 64.39%

The aggregate account level stats cards provide a high-level overview of metrics for all your services for the current month. Specifically, it displays four small cards summarizing current, month-to-date details about:

- **Requests:** the number of requests Fastly receives for your services
- **Hit ratio:** the percentage of content being accessed and currently cached by Fastly
- **Errors:** the number of error requests that occurred over time
- **Error ratio:** the percentage of cache errors for your services over time

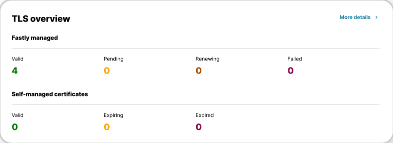
It also displays three large cards with current, month-to-date graphs broken down by day and detailing:

- **Requests:** a graph displaying the total number of requests received for your site by Fastly
- **Bandwidth:** a graph displaying the bandwidth served from Fastly's servers to your website's visitors
- **Hit ratio:** a graph displaying the percentage of content being accessed that is currently cached by Fastly

To learn more about how these stats are calculated, refer to our Historical Stats API and latency documentation.

About the TLS overview card

The **TLS overview** card shows you the current state of any Fastly-managed and self-managed TLS certificates associated with your account. Click **More details** to the right of the card title to access the TLS certificate dashboard on the Secure page. From the Secure page, you can manage and track additional details about certificate statuses and actions.



Fastly managed

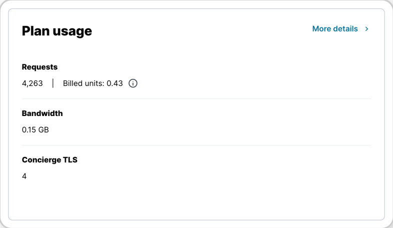
Valid 4
Pending 0
Renewing 0
Failed 0

Self-managed certificates

Valid 0
Expiring 0
Expired 0

About the Plan usage card

The **Plan usage** card shows you the current month's details about basic usage metrics associated with the products and services you've purchased for your account. Click **More details** to the right of the card title to access your account's plan usage. There you'll be able to view your account usage metrics broken down by product and region over key, monthly timeframes.



Requests
4,263 | Billed units: 0.43

Bandwidth
0.15 GB

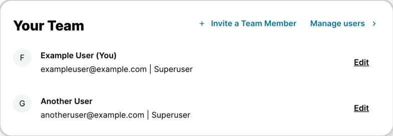
Concierge TLS
4

About the Explore Fastly card

The **Explore Fastly** card provides a curated list of documentation links for key features and products of Fastly that can help you do things like speed up defeat threats and protect your website, speed up your site and optimize images, and build an observability system with streaming logs. Click the links to the right of the card title to access Fastly's complete documentation content.

About the Your Team card

The **Your team** card displays a paginated list of users associated with your account. Depending on your assigned role, each user displayed may also include an **Edit** link to the right of their name that directs superusers to the user management controls.



Example User (You)
exampleuser@example.com | Superuser

Another User
anotheruser@example.com | Superuser

Click the links to the right of the card title to invite a team member or to manage all users and their accounts.

About the Helpful resources card

The **Helpful resources** card provides links to commonly accessed help and information resources that Fastly provides, including:

- a direct line of contact for support if you're stuck and need help
- a way to get in touch with our Sales experts if you need to discuss product and feature options for a tailored experience
- direct links to our main documentation site housing how-to guides and product documentation, the Fastly Community site, and the Fastly Academy site

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Logs menu

🔗 /en/guides/about-the-logs-menu

The Logs menu contains a link to the Log tailing page, which gives you access to the Compute log tailing feature. This feature allows you to view custom log messages from your Compute application. Specifically, the Log Tailing feature streams your service's log output to the Fastly web interface so you can respond quickly when debugging your application without setting up a third-party logging tool.

Before you begin

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

About the Log tailing page

The Log tailing page live streams the log outputs for a compute service. Streamed outputs can include any output sent to the standard output stream (`stdout`) and the standard error stream (`stderr`), as well as any runtime errors encountered by the application. For each log, the Log tailing page displays the following information:

- **Date (UTC):** a UTC timestamp of when the request occurred.
- **Request ID:** a unique identifier applied to the request.
- **Event:** the event type, either `stdout` or `stderr`.
- **Message:** the message included in the request.

A series of controls on the Log tailing page allow you to interact with and refine the logs. Specifically:

- the **Service** menu allows you to specify the Compute service that the logs are from. From the menu, you can search for a service by ID or name.
- the **Events** menu allows you to limit the displayed logs to a specific stream (e.g., `stdout` and `stderr`).
- the **Pause** button allows you to pause the live stream of Compute logs. Once paused, you can resume live stream by clicking **Live**.

Accessing the Log tailing page

To access the Log tailing page:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Logs** > **Log tailing**.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Origins page

/en/guides/about-the-origins-page

The Origins page gives you access to the Origin Inspector dashboard, which provides real-time and historic visibility into responses delivered from your origin servers to Fastly. To access the Origins page, go to **Observability** > **Origins**.

Before you begin

Origin Inspector is disabled by default. Anyone assigned the role of superuser can enable a 30-day trial directly in the web interface. After the trial expires, Origin Inspector can be purchased for an account by contacting sales@fastly.com. Once purchased, it can be enabled in the web interface by anyone assigned the role of superuser or engineer, which will result in changes to your monthly bill. When Origin Inspector is enabled, any user will be able to view the real-time and historic stats specific to Origin Inspector.

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

Enabling and disabling Origin Inspector

Origin Inspector can be enabled and disabled in the web interface by anyone assigned the role of superuser.

> NOTE
>
> To enable or disable Origin Inspector via the API, check out our developer documentation.

Enabling Origin Inspector

To enable Origin Inspector for a service:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Origins**.
3. Click the **Origin Inspector** switch to the `ON` position to enable Origin Inspector for the service.

Once Origin Inspector is enabled, you can start viewing real-time metrics about your origins immediately. Historic metrics usually become available two minutes after the end of each minute, but can take up to 15 minutes to appear.

Disabling Origin Inspector

To disable Origin Inspector for a service:

1. Log in to the Fastly web interface.
2. Go to **Observability** > **Origins**.
3. Click the **Monitor origin responses** switch to the `OFF` position to disable Origin Inspector for the service.

About the Origin Inspector dashboard

Once Origin Inspector is enabled for your account, the **Origin Inspector** dashboard displays the following metrics when you select **All origins** from the **Origin** menu:

- **Responses by Origin:** the number of responses processed by each origin.
- **Response body bytes by Origin:** the number of response body bytes returned by each origin.
- **Response header bytes by Origin:** the number of response header bytes returned by each origin.
- **Status 2xx by Origin:** the number of 2xx type (success) HTTP response status codes returned by each origin.
- **Status 3xx by Origin:** the number of 3xx type (redirection) HTTP response status codes returned by each origin.
- **Status 4xx by Origin:** the number of 4xx type (client error) HTTP response status codes returned by each origin.
- **Status 5xx by Origin:** the number of 5xx type (server error) HTTP response status codes returned by each origin.
- **Origin Latency:** a histogram showing the distribution of origin latency times. This tells you how quickly your origin is responding to Fastly. This metric only appears on the Origin Inspector dashboard when the Origin menu is set to a specific origin.

When you select any single domain from the **Origin** menu, the **Origin Inspector** dashboard displays the following metrics:

- **Total Responses:** the number of responses from an origin.
- **Origin Bandwidth:** the total bandwidth from an origin.
- **Status Response Codes:** the distribution of the number of status response codes for an origin.
- **Origin Latency:** the distribution of origin latency times, indicating how quickly your origin processes requests when responding to Fastly.
- **Status 5xx Details:** the distribution of different 5xx (server error) status response codes for the origin.
- **Status 4xx Details:** the distribution of different 4xx (client error) status response codes for the origin.
- **Status 3xx Details:** the distribution of different 3xx (redirection) status response codes for the origin.
- **Status 2xx Details:** the distribution of different 2xx (success) status response codes for the origin.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Personal profile menu

/en/guides/about-the-personal-profile-menu

To access personal details for your account, go to **Account** > **Personal profile**. This menu provides options to manage the name and email address associated with your account, your account security, the API tokens you create personally, and the appearance of the control panel.

Before you begin

Be sure you know how to access the web interface controls before learning about each of the pages you'll encounter there.

About the Personal Profile menu

From the **Personal profile** menu, you can access:

- the **Your profile** page, where you can update your name and your email address
- the **Change password** page, where you can update your current login password
- the **Two-factor authentication** page, where you can manage the multi-factor authentication controls for your personal login
- the **API tokens** page, where you can create and delete your personal API tokens you need to control access to various services and resources within your Fastly account
- the **Appearance** page, where you can change enable dark mode.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Plan usage page

/en/guides/about-the-plan-usage-page

The Plan usage page allows you to view your account usage metrics broken down by product and region over key, monthly timeframes, up to the last 24 months. These metrics provide insights into the monthly usage of things your account's billing is based on (for example, Image Optimizer requests or Concierge TLS). Use the information here to help you estimate your bill and plan for the future. To access the Plan usage page, go to **Account** > **Billing** > **Plan usage**.

Before you begin

Be sure you know how to access the web interface controls before learning about the information you'll encounter here.

About the Plan usage page

When your user role allows you access to this page, you'll find:

- a **Summary** area where you can view global, monthly usage-based metrics for your accounts.

- a **Usage** area where you can view a graph of your monthly trends for each of the usage-based metrics related to your accounts. Select a different metric using the menu above the graph. Hover the cursor over a specific date in the graph to display the specific metrics for that day.
- a **Service usage** area where you can view simple, monthly usage metrics on a per-service basis. Use the search box to search by service name or ID.
- a **Regional usage** area where you can view usage metrics within each geographical region of the world for past months and the current month to date. Expand or collapse a region by clicking on its name or use the controls to the right of the area's title to expand or collapse and hide all regions at once.

About the plan usage metrics data

When reviewing the metrics displayed on the Plan usage page, keep the following things in mind:

- **Per-service usage metrics appear only for the account you're logged in to.** To view per-service metrics for a child account, log in to that account.
- **Data doesn't appear in real time.** Unlike the metrics displayed in the Observability page graphs, data for plan usage metrics does not appear in real-time. Usage-based metrics are updated daily. The date and time of the last update appears at the top of the page.
- **Plan usage data is exportable.** You can create an exportable version of your plan usage data for each month by clicking the data export link at the top of the page.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the search controls

/en/guides/about-the-search-controls

Can't find exactly what you're looking for using the navigation controls? The universal search controls allow you to search globally across your account to find exactly what you're looking for. Enter search terms like specific service names or user names to get suggestions or limit your search to specific data types by using filters. You'll only see results appropriate to the roles and permissions you've been granted for your account.

Before you begin

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

Limitations and considerations

Keep the following things in mind:

- The universal search controls only appear in the Fastly control panel and aren't available in the Next-Gen WAF control panel.
- Universal search results are limited to Fastly users, services (both Compute and Deliver), and TLS certificates.

Using the search controls

Use the search controls to find information globally across your account. To narrow your search results, you can use filters as part of your initial search.

> ✓ TIP
>
> Are the search results not showing you what you expect? Check your roles and permissions to make sure you have the appropriate access to it.

Searching for data

To search for something, display all the results, and then use those results to navigate to what you're looking for, follow these steps:

1. Log in to the Fastly web interface.
2. In the **Search** field at the top of the page, enter your search term.
3. Press **Return**. Search results appear, grouped by type. By default, all results appear unless you've chosen to refine the results by limiting them to a specific subset using filters.
4. From the search results that appear below the field, select the appropriate result either by clicking on it directly or by using the up and down arrow keys and then pressing **Enter** when the specific result you need is highlighted.

> ✓ TIP
>
> Recent searches you've conducted appear below the search results.

Fine-tuning search results with filters

You can limit the results you'll receive during searches by prefacing search terms with words that have special meaning. These special words, called filters, restrict your data to a specific kind of result.

Filter	Data displayed
`services:`	Only the services your roles and permissions allow you access to
`users:`	Only the users associated with your account. This filter is only available if you've been assigned the role of superuser.
`tlscertificates:`	Only the TLS certificates associated with your account

To use filters, enter the filter name (followed immediately by a colon) and the search terms of the data you're looking for directly in the search field. For example, to search for your `www.example.com` service, you would enter `services: www.example.com` in the search field.

> ✓ TIP

Clicking the search filter buttons below the search field will enter the filter names into the search field for you automatically.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Services menu

/en/guides/about-the-services-menu

The Services menu contains links to two pages that display high-level metrics for your services:

- **Service Overview:** page that contains the Service Overview dashboard, which displays different information depending on if you are viewing a Deliver or Compute service.
- **All Services:** page that contains the All Services dashboard, which displays aggregated metrics from all of your services.

To access the Services menu, go to **Observability** > **Services**.

Before you begin

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

About the Service Overview page

The **Service Overview** page gives you access to the Service Overview dashboard, which displays different information depending on if you are viewing a Deliver or Compute service.

Service Overview dashboard for a Deliver service

When a Deliver service is selected from the **Service** menu, the Service Overview dashboard displays the following metrics:

- **Global POP Traffic:** the map of Fastly's points of presence (POPs) that receive requests for your service. The map only appears when you are viewing live traffic.
- **Hit Ratio:** the percentage of cache hits to all cacheable content over time.
- **Requests:** the number of requests by cache response type.
- **Bandwidth:** the total bytes delivered.
- **All Status Codes:** the number of responses sent by status group.
- **Error ratio:** the number of cache errors.

- **Status 3XX Details:** the breakdown between the number of HTTP status 301s, 302s, 304s, and other 3XX requests.
- **Status 4XX Details:** the breakdown between the number of HTTP status 400s, 401s, 403s, 404s, 416s, 427s, and other 4XX requests.
- **Status 5XX Details:** the breakdown between the number of HTTP status 500s, 501s, 502s, 503s, 504s, 505s, and other 5XX requests.
- **Misses:** the number of cache misses.
- **Object Sizes:** the number of objects served by payload size.
- **Hit Time:** the average amount of time spent processing cache hits.
- **Miss Time:** the average amount of time spent processing cache misses.
- **Header & Body Bytes Transferred:** the relative values of bytes transferred when serving the body and header portions of HTTP requests.
- **Log Lines:** the number of log lines sent.
- **Log Bandwidth:** the total log bytes sent.
- **Origin Latency:** the distribution of origin latency times that approximate how quickly your origin is responding to Fastly. This histogram only appears when you are viewing live traffic.
- **Origin Latency Percentiles:** the origin latency percentiles that approximate how quickly your origin is responding to Fastly. Each percentile is indicated by `pn`, where `n` is the percentile number. `pn` means `n`% of all requests had an origin latency value that is less than or equal to the value on the y-axis. For example, if the value of `p95` is `8s`, 95% of all requests had an origin latency that was less than or equal to 8 seconds. This graph only appears when you are viewing live traffic.
- **Cache Coverage:** the percentage of your site that you are caching with Fastly.
- **HTTP Versions:** the number of requests using HTTP/1.1, HTTP/2, and HTTP/3(QUIC).
- **Error Ratio by Region:** the percentage of cache errors per region.

The dashboard also displays collections of metrics that you can expand and collapse by clicking the collection name. The collections include:

- **Image Optimizer:** metrics for your services that relate to image optimization.
- **WebSockets:** metrics for your service that relate to WebSockets.
- **Fanout:** metrics for your service that relate to Fanout.

Service Overview dashboard for a Compute service

When a Compute service is selected from the **Service** menu, the Service Overview dashboard displays the following metrics:

- **Requests:** the total number of requests that were received for your service by Fastly.
- **Response status codes:** the total number of response status codes by category delivered by Compute.
- **Total resource limits exceeded:** the number of times a guest exceeded its resource limit, includes heap, stack, globals, and code execution timeout.

- **Memory Resource Limits Exceeded:** the number of times a guest exceeded its heap, stack, and global limits.
- **Backend Requests Errors:** the number of backend request errors, including timeouts.
- **Edge Code Errors:** the number of times a service experienced an edge code error.
- **Guest Runtime Errors:** the number of times a service experienced a guest runtime error.
- **Bytes Received by Compute:** the total bytes received by Compute.
- **Bytes Transferred to Client:** the total bytes sent from Compute to end user.
- **Bytes Transferred to Backend:** the total bytes sent to backends (origins) by Compute.
- **Bytes Transferred from Backend:** the total bytes received from backends (origins) by Compute.
- **Total Wall Clock Time:** the total, actual amount of time used to process your requests, including active CPU time.
- **CPU Time:** the amount of active CPU time used to process your requests.
- **RAM Usage:** the amount of RAM used for your service by Fastly (in bytes).
- **Logs:** the number of logs sent to your endpoints from Fastly.
- **Log Bandwidth:** the total bandwidth size of the logs sent to your endpoints from Fastly.

About the All Services page

The **All Services** page gives you access to the All Services dashboard, which displays aggregated metrics from all of your services. Specifically, it displays the following overview metrics:

- **Requests:** a graph displaying the number of requests served across your account.
- **Bandwidth:** a graph displaying the number of bytes served from Fastly's servers to your website's visitors.
- **Error Ratio:** a graph displaying the number of cache errors across your account.

The dashboard also displays collections of metrics that you can expand and collapse by clicking the collection name. The collections include:

- **Deliver:** aggregated metrics from all of your Deliver services. The metrics in this collection mirror the metrics on the Service Overview dashboard for a Deliver service.
- **Compute:** aggregated metrics from all of your Compute services. The metrics in this collection mirror the metrics on the Service Overview dashboard for a Computer service.
- **Image Optimizer:** aggregated metrics from all of your services that relate to image optimization.
- **WebSockets:** aggregated metrics from all of your services that relate to WebSockets.
- **Fanout:** aggregated metrics from all of your services that relate to Fanout.

About the Image Optimizer metrics collection

The **Image Optimizer** metrics collection appears on the Service Overview dashboard for a Deliver service and on the All Services dashboard. The collection includes the following metrics:

- **Image Optimizer total responses:** the total number of responses that came from the Fastly Image Optimizer service.
- **Image Optimizer shield responses:** the number of responses that came from the Fastly Image Optimizer service via a shield.
- **Image Optimizer total bytes delivered:** the total bytes delivered from the Fastly Image Optimizer service, including shield traffic.
- **Image Optimizer shield bytes delivered:** the total bytes delivered via a shield from the Fastly Image Optimizer service.
- **Image Optimizer video responses:** the number of video responses that came from the Fastly Image Optimizer service.
- **Image Optimizer video frames:** the number of video frames that came from the Fastly Image Optimizer service. A video frame is an individual image within a sequence of video.
- **Image Optimizer video bytes delivered:** the total bytes of video delivered from the Fastly Image Optimizer service.
- **Image Optimizer video shield responses:** the number of video responses delivered via a shield that came from the Fastly Image Optimizer service.
- **Image Optimizer video shield bytes delivered:** the total bytes of video delivered via a shield from the Fastly Image Optimizer service.
- **Image Optimizer video shield frames:** the number of video frames delivered via a shield that came from the Fastly Image Optimizer service. A video frame is an individual image within a sequence of video.

About the WebSockets metrics collection

The **WebSockets** metrics collection appears on the Service Overview dashboard for a Deliver service and on the All Services dashboard. The collection includes the following metrics:

- **WebSockets connection time:** the total duration of passthrough WebSocket connections with end users.
- **WebSockets bytes transferred to client:** the total bandwidth transferred by Fastly for your service when using requests.
- **WebSockets bytes transferred from client:** the total bandwidth transferred by Fastly for your service when receiving requests.
- **WebSockets bytes transferred to origin:** the total bandwidth transferred by Fastly to your backend when receiving requests.
- **WebSockets bytes transferred from origin:** the total bandwidth transferred to Fastly from your backend when serving requests.

About the Fanout metrics collection

The **Fanout** metrics collection appears on the Service Overview dashboard for a Deliver service and on the All Services dashboard. The collection includes the following metrics:

- **Fanout connection time:** the total duration of Fanout connections with end users.

- **Fanout published messages:** the total number of messages received from the publish API and sent to end users.
- **Fanout bytes transferred to client:** the total bandwidth transferred by Fastly for your service when serving requests.
- **Fanout bytes transferred from client:** the total bandwidth transferred by Fastly for your service when receiving requests.
- **Fanout bytes transferred to backend:** the total bandwidth transferred by Fastly to your backend when receiving requests.
- **Fanout bytes transferred from backend:** the total bandwidth transferred to Fastly from your backend when serving requests.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the Trials page

/en/guides/about-the-trials-page

The Trials page allows you to view details about product or feature trials that are active or can be started. The Trials page also provides usage metrics for trials and the ability for superusers to upgrade to full access of the product or feature. To access the Trials page, go to **Account** > **Billing** > **Trials**.

Before you begin

- Only users assigned the billing or superuser role can access this page.
- Be sure to review the limitations on trials before starting any product or feature trial.

About the Trials page

When your user role allows you access to this page, a list of cards for products and features with trials appears. These cards represent trials across your account, so you can check trials that were enabled by other users with access.

Use the search bar to filter the page to view only trial details for a specific product.

Starting a trial

Superusers can start a trial from the Trials page. To start a trial, click **Start trial** and then confirm you want to start the trial. Trial cards for products or features that have been activated, regardless of whether that trial is now expired, contain the following details:

- the date on which the trial expires
- the date on which the trial started

- the user who started the trial

> NOTE
>
> The user name appears as "Fastly" if the trial was started by an internal Fastly admin and "Deleted user" if the trial was started by a user which has since been deleted.

- usage details for the trial
- a link to relevant documentation for the product or feature

A trial can only be started once per account. Once the trial is over, you can upgrade your account to full access of the product or feature by clicking **Upgrade**.

Monitoring trial usage

Once a trial has been started, click **Usage details** to view usage metrics about your trial period. You can view these usage metrics even after the trial has expired.

* * *

About the user quick links menu

/en/guides/about-the-user-quick-links-menu

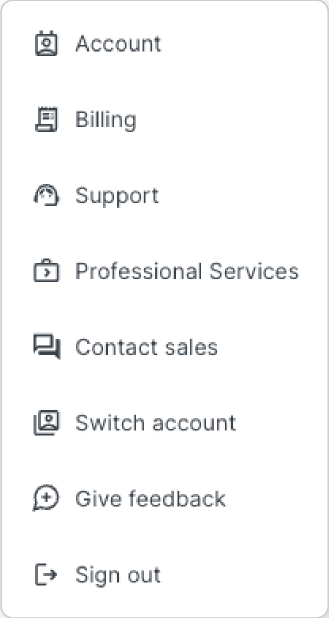
The user quick links menu provide you with direct links to account, billing, and contact information you're likely to want access to most often.

Before you begin

Be sure you know how to access the web interface controls before learning about the details you'll encounter here.

About the user quick links menu

The user quick links menu appears at the far right of the top navigation controls when you log in to the Fastly control panel:



Use this menu to:

- access company and personal profile information about your account
- access your account's billing-related details (if you've been assigned a role with billing access)
- request help from Fastly Support
- contact Sales
- log out of the web interface

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

About the web interface controls

/en/guides/about-the-web-interface-controls

Fastly provides web interface access to all of its features and functions, which are also accessible using Fastly's application programming interface (API).

Before you begin

Before you begin learning about the Fastly web interface controls, it's useful to know the basics of content delivery and how caching and CDNs work.

Accessing Fastly's web interface controls

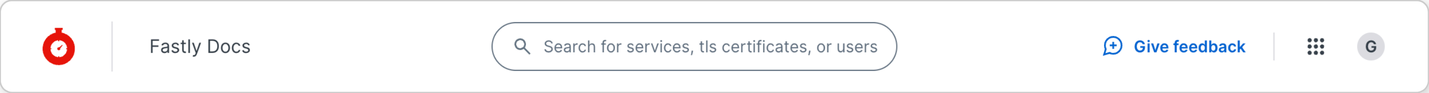
Access to Fastly's web interface controls requires you to sign up for a Fastly account. Creating an account is free. Once you've created an account, you can navigate to the controls via the Fastly login page at https://manage.fastly.com, either directly using any standard web browser or by clicking a link at the top right of almost all pages at the Fastly website.

About the web interface controls

If it's your first time logging in, the basic controls may appear a bit empty. More details will begin to appear as you start working with services.

Top navigation bar

The top navigation bar appears across the top of the Fastly web interface. It contains primary navigation elements that include the name of the company for the account you're logged into, the search field, a link you can use to give Fastly feedback, and a switcher menu that provides direct access to both the Fastly and Signal Sciences applications from a single location.



The user quick links menu appears to the right of the switcher menu. You can use the user quick links menu to access account-specific settings, billing-related details, and support links.

Navigation sidebar

The navigation sidebar provides access to:

- the **Home** page, which appears by default when you log in, where you view summarized details about all your services
- the **Observability** controls that you can use to monitor real-time and historic caching-related details about a specific service
- the **Deliver** page where you define the settings that specifically control how cached content should behave and be delivered
- the **Compute** page where you define how each instance of your Wasm services should behave and interact with its data sources
- the **Secure** controls that you can use to access the different security products Fastly has to offer
- the **Resources** controls that you can use to access data stores shared by your services
- the **Account** controls that you can use to access account-specific settings, personal profile information, and billing-related details

TIP

Check with a superuser on your account if you don't have access to the links you expect to. The roles and permissions assigned by whoever manages your company's account can change which of the navigation links you have access to.

Enabling dark mode for the web interface

You can change the theme of the Fastly web interface. Follow these instructions to enable dark mode for your personal profile:

1. Log in to the Fastly web interface and select **Account** from the navigation sidebar.
2. Go to **Account** > **Personal profile** > **Appearance**.
3. To enable dark mode, click the **Dark mode** switch.

You can click the switch again to change the theme back to light mode.

Browser requirements and recommendations

To access the Fastly web interface, your web browser must be configured to support cookies, JavaScript, and site data (also referred to as local storage or web storage).

We support a minimum display width of 768 pixels on the latest version of the following browsers:

- Google Chrome
- Firefox
- Safari

If you aren't using one of these browsers, then some visual styling may not be correct when using the Fastly web interface.

We strongly recommend updating your browser before beginning any debugging of Fastly services and before reporting problems to Fastly Customer Support. You can find the latest, downloadable versions of all major browsers online. The list at Browse Happy may help you.

What's next

Dig deeper into details about all areas of the web interface controls before you move on to using them to work with services.

* * *

Integrations guides

Category: Streaming logs

These articles describe how we support real-time log streaming of data that passes through Fastly.

About Fastly's real-time log streaming features

/en/guides/about-fastlys-realtime-log-streaming-features

To help you tune the performance of your Fastly services, we support real-time log streaming of data that passes through Fastly. We support a number of protocols that allow you to stream logs to a variety of locations, including third-party services, for storage and analysis.

Supported protocols and logging providers

Fastly supports a variety of syslog-compatible logging providers, such as Sumo Logic and Papertrail. In addition, we provide a syslog endpoint specifically for sending log files to other syslog-based software (for example, to Logstash, part of the ELK stack, which supports input via syslog).

We also support other methods of sending logs besides the syslog protocol. We allow pushing of log files to Amazon S3 buckets as well as any S3-compatible providers (such as DreamHost's DreamObjects). And we support FTP uploading.

As part of our third-party integrations, Fastly offers a number of endpoints to which you can stream logs. If the logging endpoint you're looking for isn't here, contact support for suggestions on another endpoint that might provide the same functionality.

Supported log streaming features

Fastly's real-time log streaming supports the following specific features:

- **TLS support.** Fastly allows logging configuration information to be sent over TLS (Transport Layer Security) for certain endpoints. This means that logging information can be encrypted while in transit, which allows you to send potentially sensitive information to log files without exposing data.
- **Encryption.** Fastly allows you to encrypt log files for certain endpoints before they are written to disk. We encrypt files using OpenPGP (Pretty Good Privacy). For our Amazon S3 endpoint in particular, we also support server-side encryption.
- **Customized log formats.** Fastly allows you to change the format of your logs by providing variables compatible with the Apache Common Log Format (NCSA Common log format).
- **Log file locations.** Fastly provides two different ways for you to change where your log files are written for certain endpoints. You can change a log file's timestamp format (for example, if you wanted to remove characters from the log file name) and you can control the specific path to which those files are written.
- **Multiple endpoints.** Fastly allows you to send logs to multiple endpoints.
- **Allowlisting.** Fastly's publicly available list of IP ranges allows you to enable Fastly-only access to your logging servers through your firewall.

How real-time log streaming works

Varnish sends all streaming log records to a log aggregator, which streams them in near-real-time to the logging endpoint you configure.

* * *

Changing log compression formats

/en/guides/changing-log-compression-options

Fastly's Real-Time Log Streaming feature allows you to specify compression format and options for file-based logging endpoints. These include the Azure Blob, FTP, Google Cloud Storage, Kafka, OpenStack, Amazon S3, SFTP, Digital Ocean, and Cloud Files logging endpoints.

Available log compression formats

Although the default is to use no compression, we allow you to choose one of several compression mechanisms:

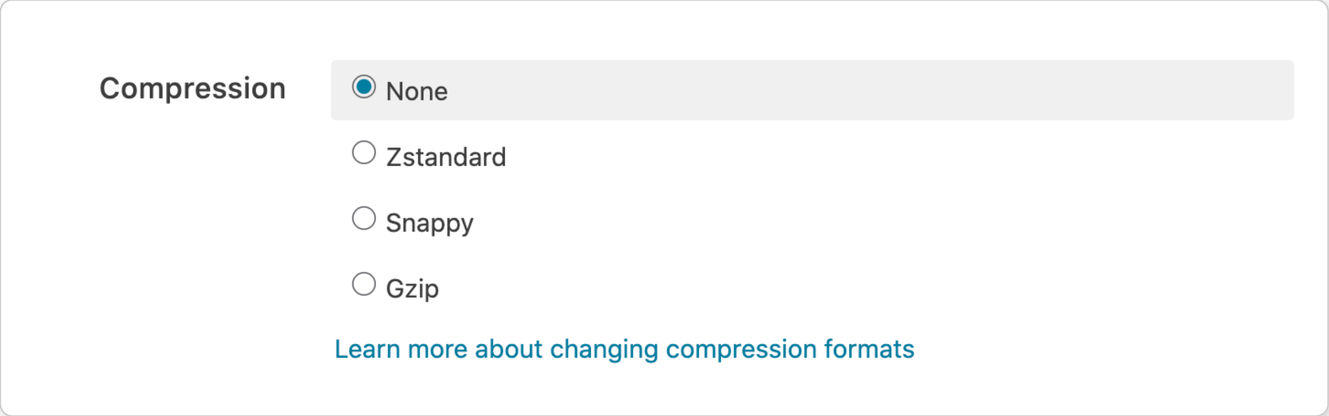
- **Zstandard**, a compression algorithm defined by RFC 8478
- **Snappy**, a compression and decompression library used by many Google products as referenced in the Snappy compressed format description
- **Gzip**, a compression utility as defined in RFC 1952 and RFC 1951

Using a different compression format

> IMPORTANT
>
> If you're using Gzip compression, the web interface defaults to a Gzip compression level of 3 and can only be changed using the Logging API. If the Gzip compression level has been set to a value other than `3` via an API call, then that level is displayed as a read-only value.

Follow these instructions to update a file-based logging endpoint's compression format using the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**.
5. Click the name of a file-based logging endpoint you want to edit.
6. Click **Advanced options** near the bottom of the page.
7. In the **Compression** section, select a compression format for the logging endpoint.



Compression

● None

○ Zstandard

○ Snappy

○ Gzip

8. Click **Update**.
9. Click **Activate** to deploy your configuration changes.

You can use the Logging API for any file-based logging endpoint to update the compression format.

The API provides two fields for specifying log file compression options: `compression_codec` and `gzip_level`.

- `compression_codec` is a string that specifies the codec used to compress your logs. One of the following:
  - `zstd` Use Zstandard compression.
  - `snappy` Use Snappy compression.
  - `gzip` Use Gzip. Defaults to gzip compression level 3.
- `gzip_level` is an integer in the range of 0, signifying no compression, to 9, signifying maximum compression.

> NOTE
>
> Use of `compression_codec` is recommended over use of `gzip_level` unless you need to set the gzip compression level to a value other than 3.

Keep in mind the following with respect to the request payload:

- Only one of `compression_codec` or `gzip_level` should be included in a request payload. Specifying both will result in an error.
- If `compression_codec` is set to "gzip", `gzip_level` will default to 3.
- If `compression_codec` is not specified in the request payload and a non-zero value for `gzip_level` is set, then `compression_codec` will default to "gzip".
- If neither `compression_codec` nor `gzip_level` is specified in the request, no compression is applied to log files and `compression_codec` will default to `null`.

For example, to update the compression format for a service's SFTP logging endpoint to use the Snappy compression codec, the curl command would look something like this:

```
$ curl -X PUT -H "Fastly-Key: YOUR_FASTLY_TOKEN"  -H "Accept: application/json" -
```

To apply gzip compression to an Azure Blob Storage logging endpoint using the `compression_codec` field, the curl command would look like this:

```
$ curl -X PUT -H "Fastly-Key: YOUR_FASTLY_TOKEN"  -H "Accept: application/json" -
```

To update a Google Cloud Storage logging endpoint's compression format to 1 using the `gzip_level` field, the curl command would look like this:

```
$ curl -X PUT -H "Fastly-Key: YOUR_FASTLY_TOKEN"  -H "Accept: application/json" -
```

* * *

Changing log line formats

/en/guides/changing-log-line-formats

Fastly's Real-Time Log Streaming feature allows you to change the format that your log messages are delivered in on select logging endpoints. We allow you to choose one of several formats:

- **Blank** is the default. There's no prefix — just your log message. This is useful when writing to JSON and CSV files.
- **Classic** is a legacy format based on RFC 3164. Read Classic Format for more information.
- **Loggly** is a structured syslog prefix format based on RFC 5424.
- **Logplex** is a Heroku-style prefixed syslog format.

Using a different log line format

A number of logging endpoints can be updated to use a message format other than the default via either the web interface or the API.

Follow these instructions to update a logging endpoint using the web interface:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**.
5. Click the name of a logging endpoint you want to edit.
6. Click **Advanced options** near the bottom of the page.
7. In the **Select a log line format** section, select a log line format for the logging endpoint.
8. Click **Update**.
9. Click **Activate** to deploy your configuration changes.

Run the following command to update a logging endpoint using the API:

```
$ curl -X PUT -H 'Fastly-Key: FASTLY_API_TOKEN' -H 'Content-Type: application/json' 'ps
```

Keep in mind that the `message_type` field is a per-object field. Updating it on one logging object *will not* change it on any other objects.

For example, to update a Google Cloud Storage logging endpoint named `GCS Test` to use the `blank` message type, the curl command would look something like this:

```
$ curl -X PUT -H 'Fastly-Key: FASTLY_API_TOKEN' -H 'Content-Type: application/json' '
```

NOTE

The `log name` included a space that needed to be URL encoded (the space into `%20`).

Classic Format

Classic is a legacy format based on the RFC 3164 protocol. It is not, however, strictly compliant with RFC 3164. To make logging easier to synchronize across a global network, classic uses timestamps based on the RFC 3339 protocol, which specifies that timestamps include a timezone. RFC 3164, on the other hand, specifies local time without any timezone data.

The following example shows a message in the classic log format.

```
<134>2016-07-04T22:37:26Z cache-sjc3128 LogTest[62959]: <your log message>
```

The prefix begins with the message priority (always `<134>`, which means `Facility=Local0, Severity=Informational`), followed by the date and time the log was sent (`2016-07-04T22:37:26Z`), the cache node it came from (in this case, `cache-sjc3128`), the name of your log (`LogTest`) and the ID of the process sending it (`62959`).

* * *

Changing log placement

/en/guides/changing-log-placement

Fastly's Real-Time Log Streaming feature allows you to specify where the logging call should be placed in the generated VCL.

Available log placement options

You can choose one of the following log placement options:

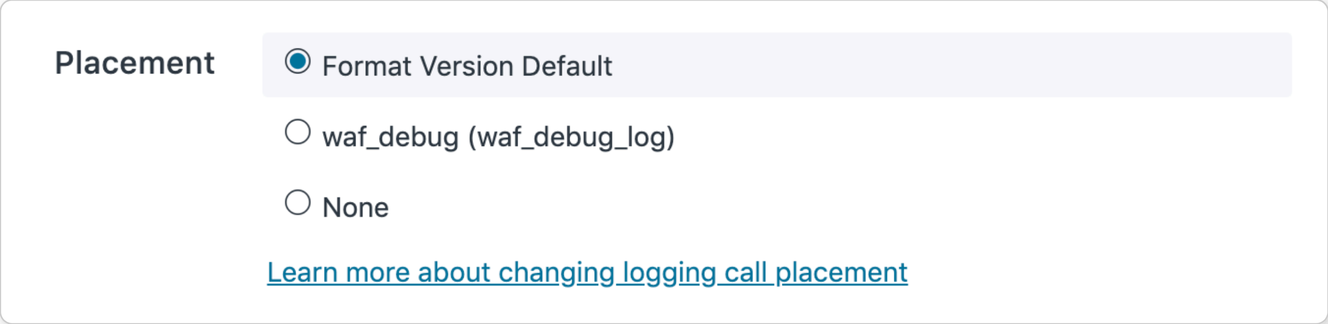
- **Format Version Default** puts the log statement in `vcl_log` if the logging endpoint is using version 2 log format. If the logging endpoint is using version 1 log format, puts the log statement in `vcl_deliver`.
- **None** prevents the log statement from being rendered in VCL. Use this option if you intend to write a log statement manually in custom VCL.

Changing log placement

Follow these instructions to update a logging endpoint's VCL placement using the web interface:

1. Log in to the Fastly web interface.

2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**.
5. Click the name of the logging endpoint you want to edit.
6. In the **Placement** section, select a placement for the logging endpoint.



Placement

● Format Version Default

○ waf_debug (waf_debug_log)

○ None

Learn more about changing logging call placement

7. Click **Update**.
8. Click **Activate** to deploy your configuration changes.

You can use the API to update a logging endpoint's VCL placement. The API provides a `placement` field for specifying where in the generated VCL the logging call should be placed. It can be one of the following:

- `""` uses the default version placement when not set. Logging endpoints using version 2 log format are placed in `vcl_log`. Logging endpoints using version 1 log format are placed in `vcl_deliver`.
- `none` prevents the log statement from being rendered in VCL. Use this option if you intend to write a log statement manually in custom VCL.

  For example, to update the logging placement to `none`, the curl command would look like this:

  ```
  $ curl -X PUT -H 'Fastly-Key: FASTLY_API_TOKEN' -H 'Content-Type: application/jso
  ```

* * *

Changing where log files are written

/en/guides/changing-where-log-files-are-written

For supported logging endpoints that write files to remote services, Fastly uses a combination of factors to ensure log files aren't overwritten, including:

- Using the file creation timestamp.
- Generating a unique ID.

- If a file with the same timestamp and UID combination exists, incrementing a counter and adding that to the end of the filename.

To change where log files are written, you can modify the `path` and `timestamp_format` variables on select endpoints. The logging system combines the `path`, `timestamp_format`, and `uid` variables to create the file name:

```
<path><timestamp>-<uid>.log<suffixes>
```

This guide explains how to use the `path` and `timestamp_format` variables to control where log files are written.

Timestamp format

You may want to consider changing the timestamp format to remove characters from the log filenames. For example, if you're working with Elastic MapReduce, you might need to remove the colons in the filename.

The `timestamp_format` variable is provided as a strftime compatible format. The default format is ISO 8601 Combined Date/Time Format:

```
%Y-%m-%dT%H:%M:%S.000
```

The variables are expanded when the file is created. For example, `%Y` will be replaced by the current year and `%m` by the current month number:

```
<year>-<2 digit month number>-<2 digit day number>T<hour>:<minute>:<second>
```

The timestamp for a file created at midnight on January 1st, 1970 would be `1970-01-01T00:00:00.000`.

Path

File-based (e.g., SFTP) and object-based (e.g., S3) endpoints use the `path` variable in different ways. File-based endpoints use `path` to create directories and files in the logging endpoint's storage. Directories will be created automatically if possible. Object-based endpoints use `path` plus the file name to create the key in an object store. No actual directories are created by object-based endpoints because they don't use directory structures in their storage.

File-based and object-based endpoints both look for `/` characters in `path`. If `path` ends in a trailing `/`, then `path` is treated as a directory by a file-based endpoint and as a *folder* (the logical equivalent of a directory) by an object-based endpoint. For example, if the value of `path` were `my_logs/`, then log files would be written to the directory (or *folder*) `my_logs`. However, if the value of `path` were `my_logs`, without the trailing `/`, then log files would be written to the top-level directory and would be prefixed with the string `my_logs`. Furthermore, if the value of `path` were `my_logs/foo`, then log files would be written to the `my_logs` directory and would be prefixed with the string `foo`.

TIP

The path can also be a `strftime` compatible string. For example, if the variable is set to `%Y/%m/%d`, the files are written to a directory based on the year, month, and date.

Suffixes

Fastly's logging system automatically adds suffixes to files as appropriate.

Suffix	File type
.log	Plain log file
.log.gz	Gzipped log file
.log.gpg	PGP encrypted log file
.log.gz.gpg	PGP encrypted, Gzipped log file

* * *

Configuring Google IAM service account impersonation to avoid storing keys on Fastly logging

/en/guides/configuring-google-iam-service-account-impersonation-for-fastly-logging

---

When adding Google Cloud Storage, BigQuery, or Pub/Sub logging endpoints, we recommend configuring Google IAM role-based service account impersonation to avoid storing secrets by using temporary credentials instead.

To configure role-based service account impersonation through the Google Cloud Console, follow the steps below:

1. Log in to the Google Cloud Console.
2. Navigate to the IAM & Admin page.
3. Review the project name to the left of the search field on the main toolbar and make sure this is the project configured for the Fastly Google endpoint. If not, use this project selection menu to select the correct project as necessary.
4. From the left navigation, click **Service Accounts**.
5. Click the email address of the service account you intend to use for the Logging endpoint.
6. Click **Permissions**.
7. Click **Grant Access**.
8. In the **New principals** field, enter:

   ```
   fastly-logging@datalog-bulleit-9e86.iam.gserviceaccount.com
   ```

9. Click the **Role** menu to expose the Filter field.
10. In the **Filter** field, enter `Service Account Token Creator` and then select it from the list of roles that appears.
11. Click **Save**.

TIP

Check out Google's docs for details on how to configure role-based service account impersonation through the command line interface.

* * *

Creating an AWS IAM role for Fastly logging

/en/guides/creating-an-aws-iam-role-for-fastly-logging

Before adding Amazon S3 or Amazon Kinesis as a logging endpoint for Fastly services, we recommend creating an Identity and Access Management (IAM) role in AWS specifically for Fastly. Using your Fastly customer account ID and the Fastly AWS account number, you can set up a role to give Fastly access to your S3 bucket or Kinesis data stream for log delivery using temporary credentials instead of long-term credentials like an access key and secret key pair.

You can do this through the AWS Management Console or the AWS CLI.

Creating an IAM role through the AWS Management Console

Follow the steps below to create an IAM role through the AWS Management Console.

1. Log in to the AWS Management Console and open the IAM console.
2. Create a permission policy that gives Fastly permission to write objects to AWS. Click **Create policy**.
3. Click **JSON**. Copy and paste one of the following templates, replacing the name in the resource field with the Amazon Resource Name (ARN) of the Amazon S3 bucket or Kinesis data stream you want Fastly to write logs to.

`Amazon S3 template` json

```json
{
     "Version": "2012-10-17",
     "Statement": {
         "Effect": "Allow",
         "Action": "s3:PutObject",
         "Resource": "arn:aws:s3:::YourS3BucketName/*"
     }
 }
```

`Amazon Kinesis template` json

```json
{
     "Version": "2012-10-17",
     "Statement": {
         "Effect": "Allow",
     "Action": [
        "kinesis:PutRecords",
        "kinesis:ListShards"
        ],
    "Resource": "arn:aws:kinesis:::YourKinesisStreamName"
    }
}
```

4. Click **Create policy**.
5. Create a role with the trust and permissions policies attached. Select **Roles** from the navigation panel, and then click **Create role**.
6. For **Select type of trusted entity**, select **Another AWS account**.
7. For **Account ID**, enter the Fastly AWS account ID (`717331877981`).
8. Select **Require external ID**.
9. In the **External ID** field, enter your Fastly customer account ID.
10. Click **Next: Permissions**.
11. Select the checkbox next to the permission policy you created above.
12. Click **Next: Tags**.
13. *(Optional)* Add metadata to the role by attaching tags as key–value pairs. For more information about using tags in IAM, check out Amazon's documentation on tagging IAM resources.
14. Click **Next: Review**. Complete the following fields:
    - In the **Role name** field, enter a name for your role. Role names must be unique within your AWS account. They are not distinguished by case. Because other AWS resources might reference the role, you can't edit the name of the role after it has been created.
    - *(Optional)* In the **Role description** field, enter a description for the new role.
15. Review the role and then click **Create role**. The role you created appears in the list of roles on the **Roles** tab.
16. Follow the instructions in Enabling Regional Tokens to ensure your AWS account is set up to allow tokens using regional STS endpoints.

After you create your new role, select the role from the **Roles** tab to view details about the role, including the Role ARN. You will need this ARN to create your logging endpoint.

Creating an IAM role through the AWS CLI

Follow the steps below to create an IAM role through the AWS CLI:

1. Create a trust policy in JSON using your Fastly customer account ID and the Fastly AWS account number using one of the following templates. Copy the template to a text editor and replace `FastlyCustomerAccountID` with your customer account ID and `Sid` with the name of your policy. Save the file to your file system.

`Amazon S3 template` json

```json
{
        "Version": "2012-10-17",
        "Statement": {
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "FastlyCustomerAccountID"
                }
            },
            "Action": "sts:AssumeRole",
            "Principal": {
                "AWS": "717331877981"
            },
            "Effect": "Allow",
            "Sid": "S3LoggingTrustPolicy"
        }
    }
```

Amazon Kinesis template json

```json
{
        "Version": "2012-10-17",
        "Statement": {
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "FastlyCustomerAccountID"
                }
            },
            "Action": "sts:AssumeRole",
            "Principal": {
                "AWS": "717331877981"
            },
            "Effect": "Allow",
            "Sid": "KinesisLoggingTrustPolicy"
        }
    }
```

2. Create a permission policy in JSON using the Amazon Resource Name (ARN) of the S3 bucket or Kinesis data stream you want Fastly to write logs to. Copy the template to a text editor and replace the name in the resource field with the name of the S3 bucket or Kinesis data stream. Save the file to your file system.

Amazon S3 template json

```json
{
     "Version": "2012-10-17",
     "Statement": {
         "Effect": "Allow",
         "Action": "s3:PutObject",
         "Resource": "arn:aws:s3:::YourS3BucketName/*"
     }
 }
```

Amazon Kinesis template json

```json
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
                    "kinesis:PutRecords",
                    "kinesis:ListShards"
                ],
        "Resource": "arn:aws:kinesis:::YourKinesisStreamName"
    }
}
```

3. From the command line, create a role and attach the trust policy to the role. Replace `YourRoleName` with the name of your role and `file://trust-policy-file.json` with the name and location of the file in which you created your trust policy.

```
$ aws --profile personal iam create-role --role-name YourRoleName --assume-role-poli-d
```

Here's what the successful response to this command looks like:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "YourRoleName",
        "RoleId": "ABCD1234ZHHQGKDRUMGFH",
        "Arn": "arn:aws:iam::AmazonResourceName:role/YourRoleName",
        "CreateDate": "2021-03-19T23:14:27+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": {
                "Condition": {
                    "StringEquals": {
                        "sts:ExternalId": "abc12345-defg-6789-hijk-lmno10111213"
                    }
                },
                "Action": "sts:AssumeRole",
                "Principal": {
                    "AWS": "717331877981"
                },
                "Effect": "Allow",
                "Sid": "RoleForS3"
                }
            }
        }
    }
```

NOTE

Take note of the value in the `Arn` field. This is the ARN for the role, which you will need to create your logging endpoint.

4. Create the permission policy. Replace `YourPolicyName` with the name of your policy and `file://permission-policy-file.json` with the name and location of the file in which you created your trust policy.

   ```
   $ aws --profile personal iam create-policy --policy-name YourPolicyName --policy-doc
   ```

   Here's what the successful response to this command looks like:

   ```
   {
   "Policy": {
       "PolicyName": "YourPolicyName",
       "PolicyId": "ABCDJH5Z123CTIKFWXYZ",
       "Arn": "arn:aws:iam::AmazonResourceName:policy/YourPolicyName",
       "Path": "/",
       "DefaultVersionId": "v1",
       "AttachmentCount": 0,
       "PermissionsBoundaryUsageCount": 0,
       "IsAttachable": true,
       "CreateDate": "2021-03-19T23:17:42+00:00",
       "UpdateDate": "2021-03-19T23:17:42+00:00"
       }
   }
   ```

5. Attach the permissions policy to the role. Replace `YourRoleName` with the name of your role and the value after `--policy-arn` with the ARN of your permission policy.

   ```
   $ aws --profile personal iam attach-role-policy --role-name YourRoleName --policy-ar
   ```

6. Follow the instructions in Enabling Regional Tokens to ensure your AWS account is set up to allow tokens using regional STS endpoints.

Enabling Regional Tokens

Fastly uses the AWS Security Token Service API to request temporary credentials to write logs to AWS endpoints like S3 and Kinesis. Tokens can be requested from a single, global endpoint or from regional endpoints, though AWS recommends the use of Regional STS endpoints for reduced latency and improved redundancy.

Regardless of where your AWS S3 bucket or Kinesis stream is located, you must enable all of the Regional endpoints that Fastly uses to acquire tokens to provide the highest level of redundancy and token validity. For example, even though your S3 bucket may be located in `us-west-1`, Fastly may acquire the token to write to your bucket from a different region (e.g., `us-east-2`).

Follow the instructions in Activating and deactivating AWS STS in an AWS Region to activate all of the following regional endpoints that may be used by Fastly to acquire tokens.

- US East (N. Virginia) *Always active*: `https://sts.us-east-1.amazonaws.com`
- US East (Ohio): `https://sts.us-east-2.amazonaws.com`
- US West (N. California): `https://sts.us-west-1.amazonaws.com`

- US West (Oregon): `https://sts.us-west-2.amazonaws.com`

What's next

Use the IAM role you created to add Amazon S3 or Amazon Kinesis as a logging endpoint.

* * *

Custom log formats

/en/guides/custom-log-formats

Fastly provides two versions of custom log formats for use when you set up remote log streaming. All new logging endpoints use the version 2 custom log format by default. You can upgrade version 1 logging endpoints to the version 2 custom log format. You can also make version 2 look like version 1 for the sake of continuity. We've described the key advantages of the version 2 custom log format below.

Version 2 log format

This table details version 2 of Fastly's custom log formats. All variables should be prefixed by a percent sign (`%`), as indicated in the table.

Format String	Description
`%%`	The percent sign.
`%a`	The client IP address of the request. Equivalent to `req.http.Fastly-Client-IP`.
`%A`	The local IP address. Equivalent to `server-ip`.
`%B`	The size of response in bytes, excluding HTTP headers. Equivalent to `resp.body_bytes_written`.
`%b`	The size of response in bytes, excluding HTTP headers. In Common Log Format (CLF), that means a `"-"` rather than a `0` when no bytes are sent.
`%{Foobar}C`	The contents of cookie `Foobar` in the request sent to the server.
`%D`	The time taken to serve the request, in microseconds. Equivalent to `time.elapsed.usec`.
`%{FOOBAR}e`	Not supported. Always returns `"-"`.
`%f`	The filename. Equivalent to `req.url` with any query string removed, because Varnish has no notion of a file being served.
`%h`	The remote IP address. Equivalent to `req.http.Fastly-Client-IP`.
`%H`	The request protocol. Equivalent to `req.proto`.
`%{Foobar}i`	The contents of `Foobar:` header lines in the request sent to the server.

Format String	Description
`%I`	Bytes received, including request and headers. Cannot be zero Equivalent to `req.bytes_read`.
`%k`	The number of keepalive requests handled on this connection. Always returns `0`.
`%l`	Not supported. Always returns `"-"`.
`%m`	The request method. Equivalent to `req.request`.
`%{Foobar}n`	Not supported. Always returns `"-"`.
`%{Foobar}o`	The contents of `Foobar:` header lines in the reply.
`%O`	Bytes sent, including headers. Cannot be zero. Equivalent to `resp.bytes_written`.
`%p`	The canonical port of the server serving the request. Always returns `80`. Equivalent to `server.port`.
`%{format}p`	The canonical port of the server serving the request. Valid formats are `canonical`, `local`, or `remote`. Returns `80` for HTTP requests and `443` for HTTPS requests.
`%P`	Not supported. Always returns `"-"`.
`%{format}P`	Not supported. Always returns `"-"`.
`%q`	The query string (prepended with a `?` if a query string exists, otherwise an empty string). Equivalent to `req.url`.
`%r`	The first line of the request.
`%R`	Not supported. Always returns `"-"`.
`%s`	The status. For requests that got internally redirected, this is the status of the *original* request. Use `%>s` for the final status. Equivalent to `resp.status`.
`%t`	The time the request was received, in Standard English format (e.g., `01/Jan/1970:00:00:00 -0700`). The last number indicates the timezone offset from GMT.
`%{format}t`	The time, in the form given by `format`, which should be in `strftime(3)` format (potentially localized). If the format starts with `begin:` (the default) the time is taken at the beginning of the request processing. If it starts with `end:` it is the time when the log entry gets written, close to the end of the request processing. In addition to the formats supported by `strftime(3)`, the following format tokens are supported: `sec` (number of seconds since the Epoch), `msec` (number of milliseconds since the Epoch), `usec` (number of microseconds since the Epoch), `msec_frac` (millisecond fraction), and `usec_frac` (microsecond fraction). Equivalent to `time.start`.
`%T`	The time taken to serve the request, in seconds. Equivalent to `time.elapsed.sec`.
`%u`	Not supported. Always returns `"-"`.
`%U`	The URL path requested, not including any query string. Equivalent to `req.url.path`.
`%v`	The domain name of the request. Equivalent to `req.http.host`.

Format String	Description
`%V`	The same as `%v`. Equivalent to `req.http.host`.
`%{vcl}V`	The literal VCL to include without quoting. This can be used to write VCL variables to your logs (e.g., `%{client.geo.country_code}V` or `%{tls.client.cipher}V`). This %-directive is a Fastly extension and is not found in Apache.
`%X`	The connection status when response is completed. Always set as `+` (connection may be kept alive after the response is sent).

Version 1 log format

This table details version 1 of Fastly's custom log formats. All variables should be prefixed by a percent sign (`%`), as indicated in the table.

Format String	Description
`%b`	The content size of the response, calculated using the `Content-Length` header rather than actually checking the length of the response (and may therefore be wrong).
`%h`	The remote IP address.
`%l`	The remote log name. Always returns the hardcoded value `"-"`.
`%r`	The HTTP verb and request path (e.g., `GET /index.html`). Unlike Apache and version 2 log formats, the protocol version is not included.
`%>s`	The status of the last request.
`%t`	The time the request was received, in Unix `ctime` format (e.g., `Thu, 01 Jan 1970 00:00:00 GMT`) rather than Apache's Standard English format (e.g., `01/Jan/1970:00:00:00 -0700`).
`%u`	Always returns `"-"`.

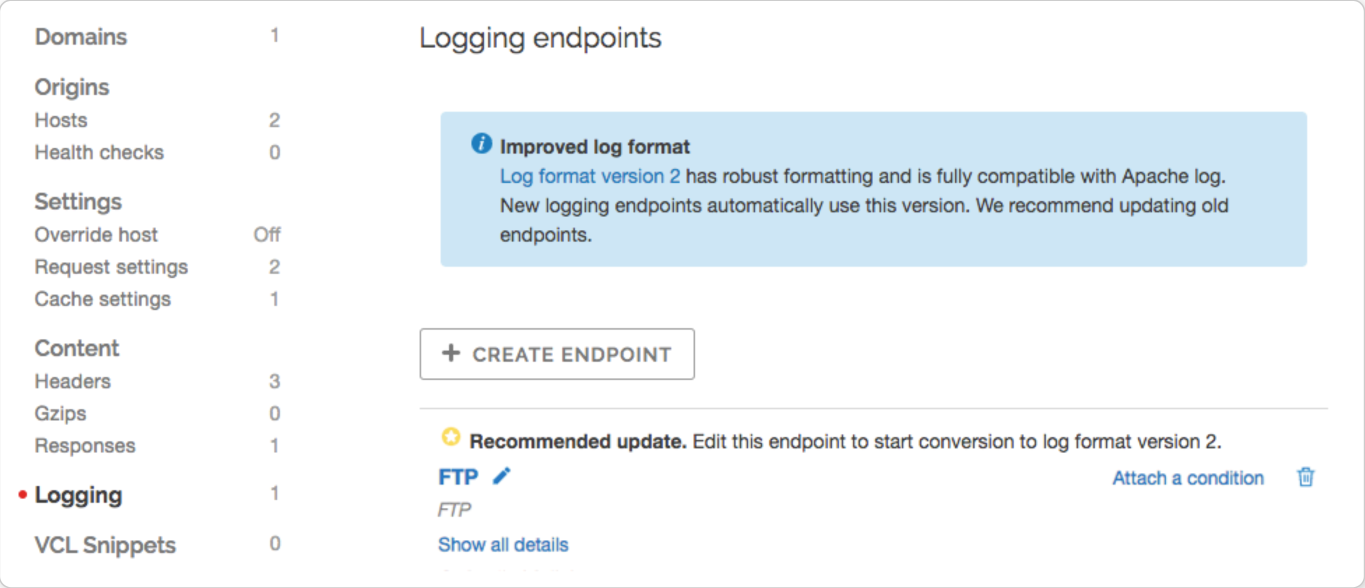
Upgrading endpoints to use version 2 log format

WARNING

Upgrading is a permanent change. Logging objects using version 2 formatting cannot be downgraded to version 1.

Follow these instructions to upgrade a logging endpoint to the version 2 custom log format:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**.



Domains 1
Origins
Hosts 2
Health checks 0
Settings
Override host Off
Request settings 2
Cache settings 1
Content
Headers 3
Gzips 0
Responses 1
Logging 1
VCL Snippets 0
CREATE ENDPOINT
Attach a condition
Show all details

Logging endpoints

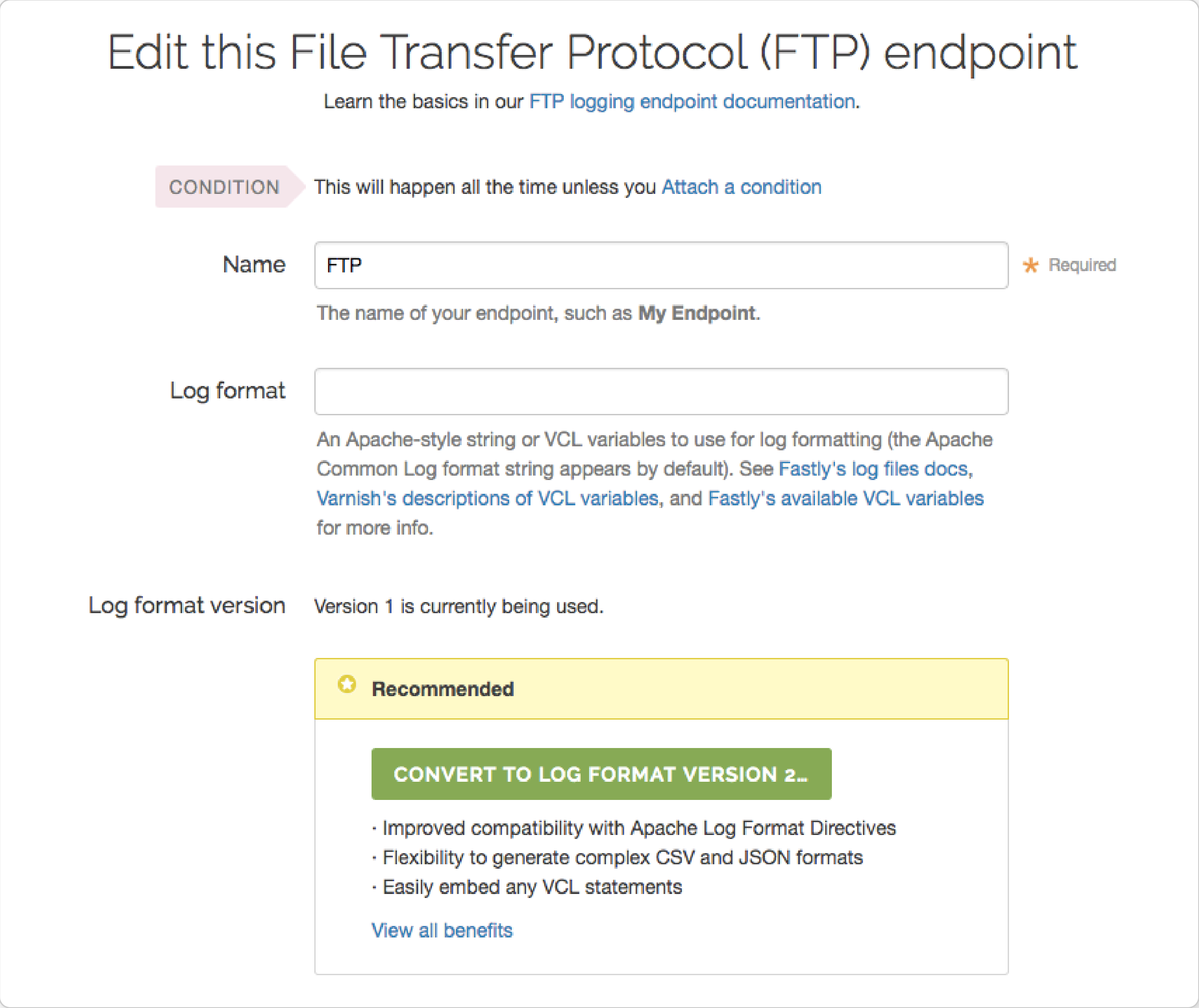
Improved log format
Log format version 2 has robust formatting and is fully compatible with Apache log. New logging endpoints automatically use this version. We recommend updating old endpoints.

Recommended update. Edit this endpoint to start conversion to log format version 2.

FTP

FTP

5. Click the name of a logging endpoint you want to edit.



CONVERT TO LOG FORMAT VERSION 2...
View all benefits

Edit this File Transfer Protocol (FTP) endpoint

Learn the basics in our FTP logging endpoint documentation.

CONDITION This will happen all the time unless you Attach a condition

Name: FTP ★ Required

The name of your endpoint, such as **My Endpoint**.

Log format:

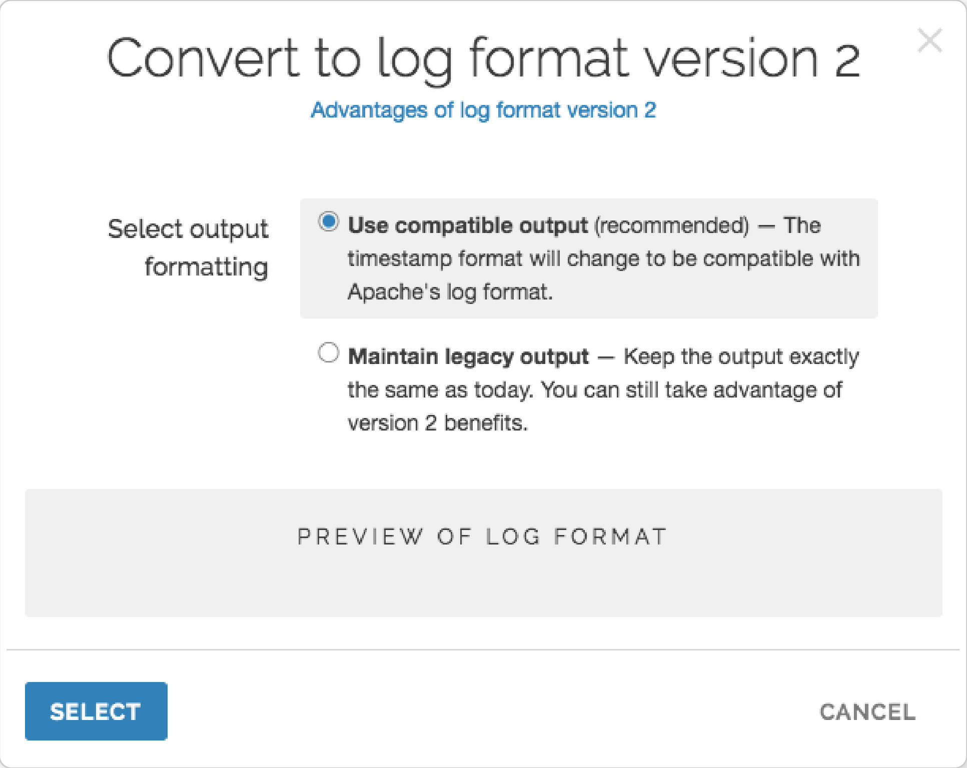
An Apache-style string or VCL variables to use for log formatting (the Apache Common Log format string appears by default). See Fastly's log files docs, Varnish's descriptions of VCL variables, and Fastly's available VCL variables for more info.

Log format version: Version 1 is currently being used.

Recommended

- Improved compatibility with Apache Log Format Directives
- Flexibility to generate complex CSV and JSON formats
- Easily embed any VCL statements

6. Click **Convert to Log Format Version 2**.


Convert to log format version 2
Advantages of log format version 2
PREVIEW OF LOG FORMAT
SELECT
CANCEL

Select output formatting

● **Use compatible output** (recommended) — The timestamp format will change to be compatible with Apache's log format.

○ **Maintain legacy output** — Keep the output exactly the same as today. You can still take advantage of version 2 benefits.

7. Select an output format:
    - **Use compatible output** is the recommended setting. This setting won't modify your timestamp format string, but your logs will be formatted differently. The new format will be compatible with Apache's log format.
    - **Maintain legacy output** uses the version 2 parser, but the generated log string will be the same. This means that any instances of `%t` need to be turned into `%{now}V`, any instances of `%r` need to be turned into `%{req.url}V`, and any instances of `%b` need to be turned into `%{resp.http.Content-Length}V`.
8. Click **Select**.
9. Click **Update** to upgrade the logging endpoint to the version 2 custom log format.
10. Click **Activate** to deploy your configuration changes.

Using the API to upgrade

To upgrade a logging endpoint using the Fastly API, either clone the active version of the service you need upgraded or choose a version of the service that is unlocked and not active, then run the following command on that in development version:

```
$ curl -X PUT -H 'Fastly-Key: FASTLY_API_TOKEN' -H 'Content-Type: application/json' 'htt:/
```

Keep in mind that the `format_version` field is a per-object field. Updating it on one logging object will *not* change it on any other objects.

For example, to update a Google Cloud Storage logging endpoint named "GCS Test," the curl command would look something like this:

```
$ curl -X PUT -H 'Fastly-Key: FASTLY_API_TOKEN' -H 'Content-Type: application/json' 'htt:/
```

NOTE

The `log name` included a space that needed to be URL encoded (the space into `%20`).

Determining which logging version is being used

To determine which logging version your service currently uses, issue the following curl command:

```
$ curl -X GET -H 'Fastly-Key: FASTLY_API_TOKEN'  'https://api.fastly.com/service/<your Ftl
```

The curl command will produce JSON output detailing the configuration of your service version. For example:

```
{
    "address": "logs.papertrailapp.com",
    "created_at": "2016-04-01T15:37:30+00:00",
    "deleted_at": null,
    "format": "time.start.msec time.to_first_byte time.elapsed req.body_bytes_read req.
    "format_version": "2",
    "hostname": "logs.papertrailapp.com",
    "name": "fastly",
    "port": "11111",
    "public_key": null,
    "response_condition": "LOG /",
    "service_id": "1a2b3c4d5e6f7g8h9j0k",
    "updated_at": "2016-04-01T19:47:47+00:00",
    "version": "123"
}
```

The `format_version` field displays either a `1` or a `2` as appropriate for the custom log format being used.

Advantages of using the version 2 custom log format

The key advantages of using the version 2 custom log format include the following:

- Log lines are generated in `vcl_log` instead of `vcl_deliver` to allow us to accurately set the various size variables because `vcl_log` is run after the object has been delivered to the browser.
- The `%t` time directive is compatible with Apache log format. In version 1, we used a non-standard time format.
- The `%r` "first line of request" directive is compatible with Apache log format. In version 1, we incorrectly left off the protocol.
- When using the `%b` directive, which represents the size of a response in bytes, excluding HTTP headers is more accurate. In version 1, we used the reported Content-Length from the origin, which could be inaccurate (especially with ESI).
- We've added all Apache logging directives that make sense. In version 1, we used a smaller subset.

Making version 2 logs look like version 1

The default logging format for version 1 is as follows:

```
%h %l %u %t %r %>s
```

Most of the directives in version 2 are exactly the same - only `%t` and `%r` are different. After you upgrade to version 2 log formats, you can recreate the appearance of version 1 logs using the new `%{...}V` directive, which allows you to specifically include VCL in logging directives:

```
%h %l %u %{now}V %{req.method}V %{req.url}V %>s
```

In addition, if you are using the `%b` directive in version 1, then you can use this directive instead:

```
%{resp.http.Content-Length}V
```

* * *

Encrypting logs

/en/guides/encrypting-logs

For supported logging endpoints, Fastly allows you to encrypt your log files before they are written to disk. The files are encrypted using OpenPGP (Pretty Good Privacy).

IMPORTANT

Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

Generating a PGP key pair

To use this feature, you'll need to use a PGP implementation (such as GPG) to generate a public and private PGP key pair. Typically, this involves running the following command in a terminal application on your personal computer:

```
$ gpg --gen-key
```

Follow the instructions shown in your terminal application. Enter your email address and set a passphrase when prompted. Remember the values you enter.

WARNING

Keep your private key safe! If you lose it, your encrypted log files will be permanently unreadable.

Exporting the PGP public key

After you generate the PGP key pair, you'll need to export your public key. Typically, this involves running the following command in a terminal application on your personal computer:

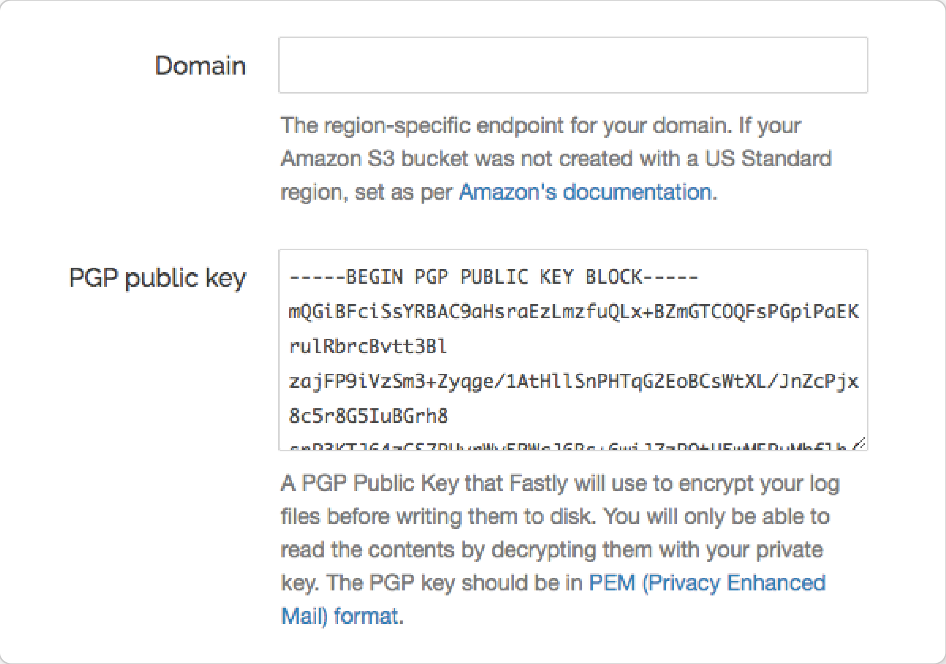
```
$ gpg --armor --export <your email>
```

The output will be in PEM (Privacy-Enhanced Mail) format and will look similar to the following:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBFciSsYRBAC9aHsraEzLmzfuQLx+BZmGTCOQFsPGpiPaEKrulRbrcBvtt3Bl
zajFP9iVzSm3+Zyqge/1AtHllSnPHTqG2EoBCsWtXL/JnZcPjx8c5r8G5IuBGrh8
snP3KTJ64zCS7PUvrWy5RWcJ6Rs+6wiJ7zPOtU5wMEPuMbflh/soy50zrwCg74XN
[...REDACTED...]
-----END PGP PUBLIC KEY BLOCK-----
```

Enabling log encryption

To enable PGP encryption for a logging endpoint that supports it, copy and paste your public PGP key into the **PGP public key** field in the Fastly web interface when creating or editing a supported logging endpoint.



Domain

The region-specific endpoint for your domain. If your Amazon S3 bucket was not created with a US Standard region, set as per Amazon's documentation.

PGP public key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
mQGiBFciSsYRBAC9aHsraEzLmzfuQLx+BZmGTCOQFsPGpiPaEK
rulRbrcBvtt3Bl
zajFP9iVzSm3+Zyqge/1AtHllSnPHTqG2EoBCsWtXL/JnZcPjx
8c5r8G5IuBGrh8
```

A PGP Public Key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy Enhanced Mail) format.

Decrypting log files

To read an encrypted log file, you'll need to download and decrypt it. Typically, this involves running the following command in a terminal application on your personal computer:

```
$ gpg --decrypt <encrypted log file>
```

Enter your passphrase to decrypt the log file.

* * *

Setting up remote log streaming for Compute

/en/guides/setting-up-remote-log-streaming-for-compute

Logs provide an important resource for troubleshooting connectivity problems, pinpointing configuration areas that could use performance tuning, and identifying the causes of service disruptions. We recommend setting up remote log streaming when you start using Fastly services.

Compute gives you multiple options for streaming logs. The quickest way to get started with logging is to use our log tailing feature, which lets you stream custom log messages from your Compute application to your local console. You can also configure Fastly's Real-Time Log Streaming feature to automatically save logs to a third-party service for storage and analysis. You can take advantage of these logging options separately or together.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

IMPORTANT

Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

How logs are streamed

Fastly uses several different log-server aggregation points and each will send logs files, none of which contain duplicate entries. These log files are created as soon as streaming starts and they're written to over the entire time period you specify (or the default). Once that time has passed, the files aren't touched any more and the logging process creates a new batch of files.

The number of log-server aggregation points may change over time in line with our capacity requirements. If you're sending logs to a storage endpoint and are concerned about the number of log files that will be created on your disk, consider choosing a logging endpoint that supports real-time ingestion, which will eliminate a need for pre-processing log files.

Configuring logging via log-tailing

The quickest way to try out logging with Compute is to use our log-tailing feature. Log tailing can be used to display output from `stdout` and `stderr` directly on your local console and doesn't require any third-party integrations or endpoints. This is useful for testing and debugging your code.

Testing and debugging on Compute has instructions on how to live-tail a service using one of our Compute starter kits.

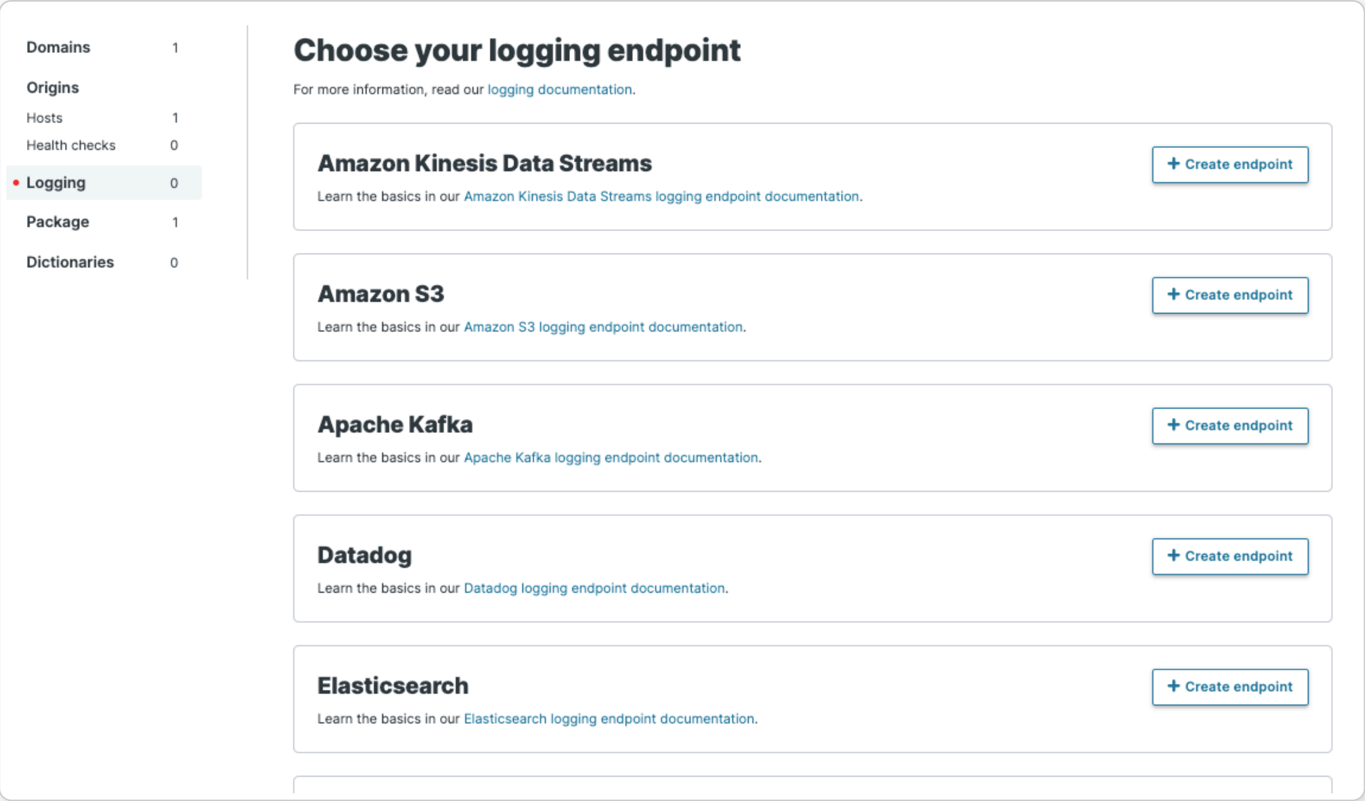
Configuring logging via third-party endpoints

Another way to configure logging on Compute is to configure one or more third-party logging endpoints. Before setting up remote log streaming, keep the following in mind:

- Be sure to double-check the delivery formats required by your logging provider and what you're delivering to them. Some providers have strict formatting requirements for the formats they allow (e.g., JSON).
- If you configure multiple logging endpoints for your service, logs will be sent to all of the logging endpoints.

Follow these instructions to access the logging settings:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**. If you've already added a logging endpoint, click **Create Endpoint**. The list of available logging endpoints appears.



Choose your logging endpoint

For more information, read our logging documentation.

Amazon Kinesis Data Streams

Learn the basics in our Amazon Kinesis Data Streams logging endpoint documentation.

Amazon S3

Learn the basics in our Amazon S3 logging endpoint documentation.

Apache Kafka

Learn the basics in our Apache Kafka logging endpoint documentation.

Datadog

Learn the basics in our Datadog logging endpoint documentation.

Elasticsearch

Learn the basics in our Elasticsearch logging endpoint documentation.

5. Follow the instructions in one of our logging endpoint guides to complete the set up process and deploy your changes.

Once you've clicked Activate to deploy your changes, events will begin being logged immediately. The logs may take a few moments to appear on your log server.

Advanced logging configuration

Depending on the starter kit you're using, there may be advanced configuration capabilities to give you more precise control of your log data. For example, the `log_fastly` crate in Rust has an option to integrate log tailing with your third-party logging endpoint, define a default endpoint for different logging levels, and support multiple logging endpoints. Advanced configuration capabilities are also possible with JavaScript and GO.

An example using Rust

The following example demonstrates how to format log data as JSON to send to a third party endpoint. It is based on the default starter kit for Rust.

In this example, we have configured an endpoint named `my_endpoint` that expects a JSON object like:

```
{
   "client_ip": "127.0.0.1",
   "host": "firstly-coral-feather",
   "request_method": "GET",
   "timestamp": "2022-09-22 17:30:17.225185 UTC",
   "trace_id": "cffc6f9a6b0645ad89adc569c99e91aa",
   "url": "https://firstly-coral-feather/"
}
```

To format log data as JSON, we add `serde_json` as a dependency in the **Cargo.toml** file along with Chrono for timestamp data.

```
[dependencies]
fastly = "0.8.7"
log-fastly = "0.8.7"
log = "0.4.17"
chrono = "0.4"
serde_json = "1.0.85"
```

Then, we modify the `src/main.rs` file from the starter kit:

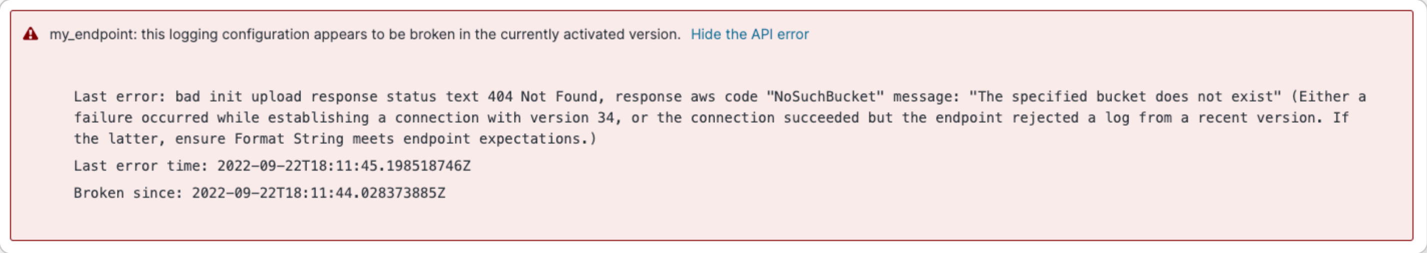
```
//! Compute logging demo
use fastly::http::{header, Method, StatusCode};
use fastly::{mime, Error, Request, Response};
use serde_json::json;
#[fastly::main]
fn main(req: Request) -> Result<Response, Error> {
   // Filter request methods...
   match req.get_method() {
      // Allow GET and HEAD requests.
      &Method::GET | &Method::HEAD => (),
      // Deny anything else.
      _ => {
            return Ok(Response::from_status(StatusCode::METHOD_NOT_ALLOWED)
               .with_header(header::ALLOW, "GET, HEAD")
               .with_body_text_plain("This method is not allowed\n"))
      }
   };
   // Pattern match on the path...
   match req.get_path() {
       // If request is to the `/` path...
       "/" => {
             // Initialize the logger with the one endpoint
             // Notice we are echoing to stdout, so we don't need separate println! for
             log_fastly::Logger::builder()
                .max_level(log::LevelFilter::Info)
                .default_endpoint("my_endpoint")
                .echo_stdout(true)
                .init();
             // Get some request data to log
             let ts = chrono::Utc::now();
             let record = json!({
                "timestamp": ts.format("%F %T%.6f %Z").to_string(),
                "trace_id": std::env::var("FASTLY_TRACE_ID").unwrap_or_else(|_| String::
                "client_ip": req.get_client_ip_addr().unwrap().to_string(),
                "host": req.get_header_str("Host"),
                "request_method": req.get_method_str(),
                "url": req.get_url_str(),
             });
             // Send the logs
             // note we didn't specify a target so it goes to `my_endpoint`, which we se
             // We could have also specified the target log::info!(target: "my_endpoint"
             log::info!("{}", record.to_string());
             // Send a default synthetic response.
             Ok(Response::from_status(StatusCode::OK)
                .with_content_type(mime::TEXT_HTML_UTF_8)
                .with_body(include_str!("welcome-to-compute@edge.html")))
       }
       // Catch all other requests and return a 404.
       _ => Ok(Response::from_status(StatusCode::NOT_FOUND)
             .with_body_text_plain("The page you requested could not be found\n")),
    }
```

Once ready, build and deploy the service and tail the service to see the logging output in your developer console.

Troubleshooting common logging errors

Logging errors fall into two general categories: configuration errors and formatting errors.

Configuration errors may include things like incorrect bucket names or authentication information. Configuration errors generally show up in the details of an error message displayed in the web interface.



⚠ my_endpoint: this logging configuration appears to be broken in the currently activated version. Hide the API error

```
Last error: bad init upload response status text 404 Not Found, response aws code "NoSuchBucket" message: "The specified bucket does not exist" (Either a
failure occurred while establishing a connection with version 34, or the connection succeeded but the endpoint rejected a log from a recent version. If
the latter, ensure Format String meets endpoint expectations.)
Last error time: 2022-09-22T18:11:45.198518746Z
Broken since: 2022-09-22T18:11:44.028373885Z
```

To resolve configuration errors, confirm that your bucket name and authentication keys are accurate and spelled correctly.

Formatting errors may include things like invalid JSON, a schema mismatch, or missing vendor-specific fields. These cause errors because some remote endpoints have specific formats for the ingested data, while others may be more flexible in what they allow.

If log messages are not showing up in your remote endpoint but are displayed in the log tail (assuming you have configured your logging endpoint for use with log tailing), check the format of the JSON in the log message. Verify that it is valid JSON and matches the format your provider expects.

Receiving logs

If an error in the Fastly web interface suggests that your logging configuration appears to be broken for the currently activated service version but you're still receiving some logs, not all of Fastly's log aggregators may be able to connect to your endpoint's server. It's likely the maximum number of concurrent connections has been reached. Try configuring your logging endpoint's server to allow a higher maximum number of inbound connections and then see if the error clears up after a couple of hours.

* * *

Setting up remote log streaming

/en/guides/setting-up-remote-log-streaming

Fastly's Real-Time Log Streaming feature allows you to automatically save logs to a third-party service for storage and analysis. Logs provide an important resource for troubleshooting connectivity problems, pinpointing configuration areas that could use performance tuning, and identifying the causes of service disruptions. We recommend setting up remote log streaming when you start using Fastly services.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Before you begin

Before setting up remote log streaming, keep the following in mind:

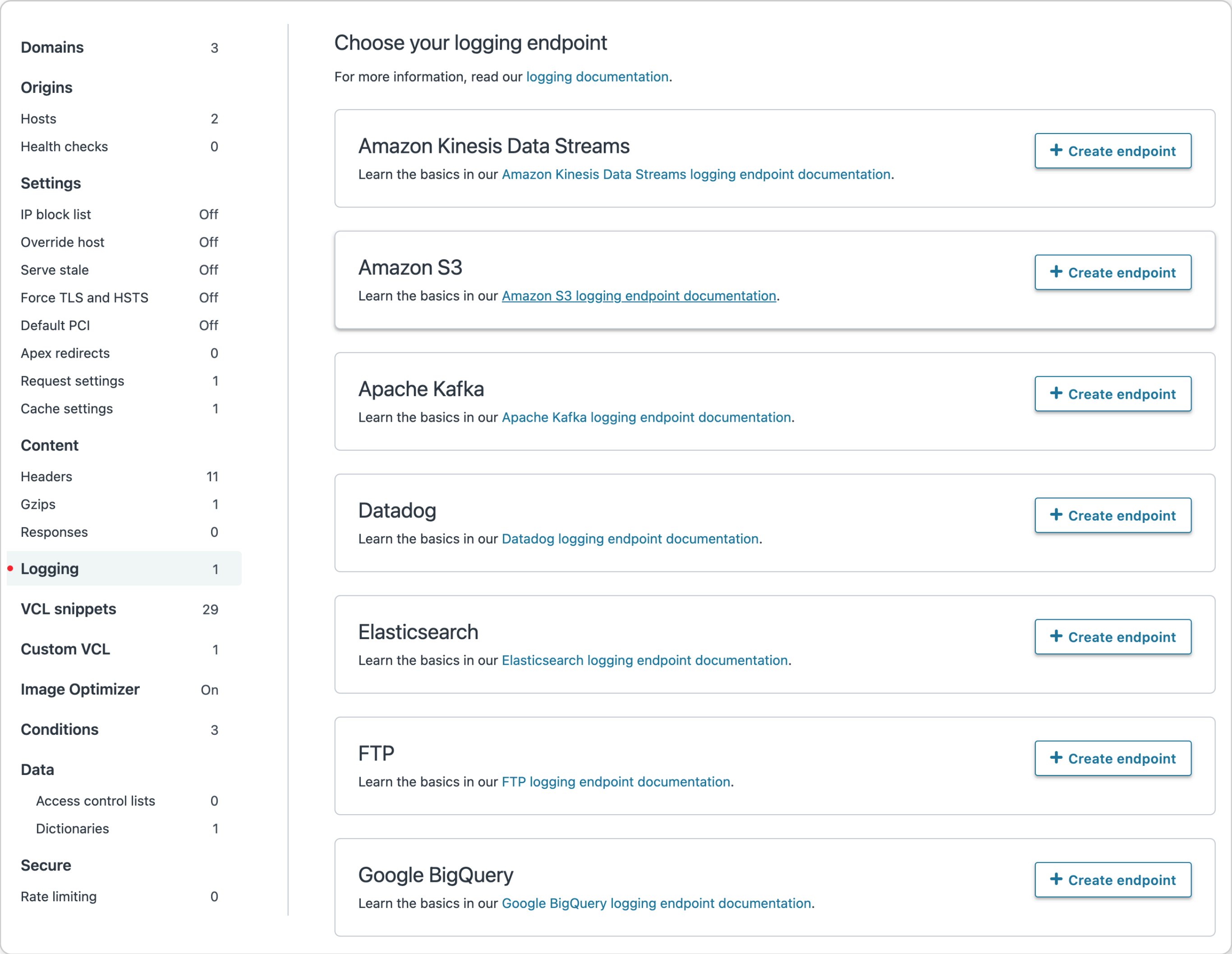
- Be sure to double-check the delivery formats required by your logging provider and what you're delivering to them. Some providers have strict formatting requirements for the formats they allow (e.g., JSON).
- If you configure multiple logging endpoints for your service, logs will be sent to all of the logging endpoints.

> IMPORTANT
>
> Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

Configuring logging endpoints

You can configure one or more logging endpoints for Fastly services. Follow these instructions to access the logging settings:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Logging**. The logging endpoints page appears. If you've already added a logging endpoint, click **Create Endpoint**.



Domains	3
Origins	
Hosts	2
Health checks	0
Settings	
IP block list	Off
Override host	Off
Serve stale	Off
Force TLS and HSTS	Off
Default PCI	Off
Apex redirects	0
Request settings	1
Cache settings	1
Content	
Headers	11
Gzips	1
Responses	0
Logging	1
VCL snippets	29
Custom VCL	1
Image Optimizer	On
Conditions	3
Data	
Access control lists	0
Dictionaries	1
Secure	
Rate limiting	0

Choose your logging endpoint

For more information, read our logging documentation.

Amazon Kinesis Data Streams

Learn the basics in our Amazon Kinesis Data Streams logging endpoint documentation.

Amazon S3

Learn the basics in our Amazon S3 logging endpoint documentation.

Apache Kafka

Learn the basics in our Apache Kafka logging endpoint documentation.

Datadog

Learn the basics in our Datadog logging endpoint documentation.

Elasticsearch

Learn the basics in our Elasticsearch logging endpoint documentation.

FTP

Learn the basics in our FTP logging endpoint documentation.

Google BigQuery

Learn the basics in our Google BigQuery logging endpoint documentation.

5. Follow the instructions in one of our logging endpoint guides to complete the set up process and deploy your changes.

Once you've clicked Activate to deploy your changes, events will begin being logged immediately. The logs may take a few moments to appear on your log server.

How, when, and where logs are streamed

To control log streaming, Fastly provides two versions of custom log formats, each of which uses Apache-style logging directives. The logging format strings in each of these versions are based on the Common Log Format (CLF).

Logs are streamed over TCP, not UDP, optionally using TLS for security with supported endpoints. Additionally, if you are using custom VCL be sure to include the `#FASTLY log` macro in your `vcl_log` handler.

By default, logs are placed in your root directory every hour using the file naming format `YYYY-mm-ddThh:mm:ss-<uid>`. You can change both the frequency and path of these files. Our guide on changing where log files are written provides more information.

If you've configured multiple logging endpoints for your service, the logs will be sent to all of the logging endpoints.

Fastly uses several different log-server aggregation points and each will send logs files, none of which contain duplicate entries. These log files are created as soon as streaming starts and they're written to over the entire time period you specify (or the default). Once that time has passed, the files aren't touched any more and the logging process creates a new batch of files.

The number of log-server aggregation points may change over time in line with our capacity requirements. If you're sending logs to a storage endpoint and are concerned about the number of log files that will be created on your disk, consider choosing a logging endpoint that supports real-time ingestion, which will eliminate a need for pre-processing log files.

Escaping characters in logs

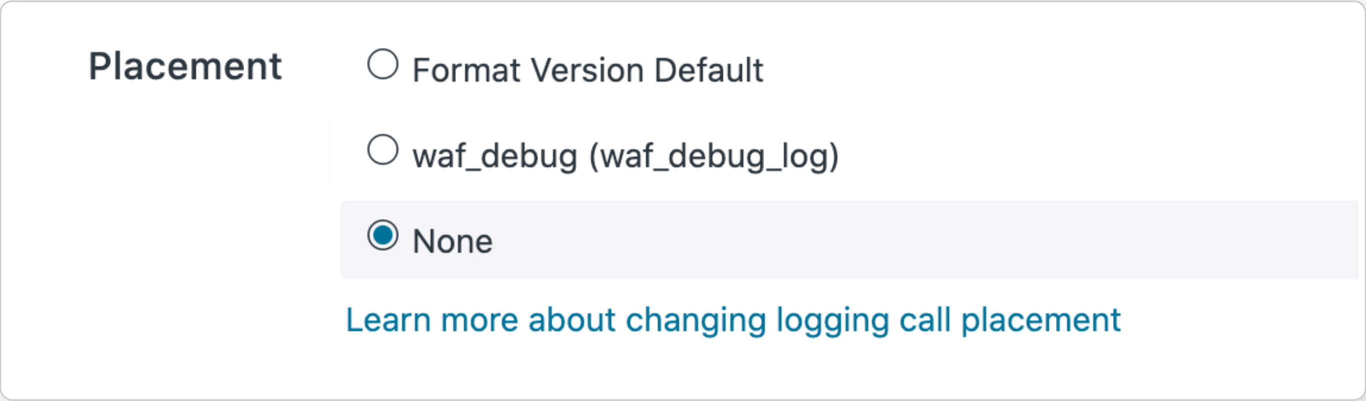
Logs respond to VCL like any other object. For example, the following code can escape quotes from User-Agent your log stream:

```
log {"syslog serviceid endpointname :: "} {"""} cstr_escape(req.http.user-agent);
```

Preventing duplicate log entries when using custom VCL

If you use custom VCL commands for logging, you may notice duplicate entries in your logs. This happens because logs are being generated by both Fastly and the custom VCL logging commands. You can eliminate the duplicate entries by following these steps:

1. On the Logging endpoints page, click the name of the logging endpoint you want to edit.
2. From the **Placement** menu, select **None**.



Placement

○ Format Version Default

○ waf_debug (waf_debug_log)

● None

3. Click **Update**.

4. Click **Activate** to deploy your configuration changes.

Fastly will stop generating log entries, and your logs will only contain entries generated by the custom VCL logging commands.

Troubleshooting common logging errors

The Fastly web interface displays errors with your logging configuration. You can also use the `logging_status` API endpoint to troubleshoot problems with your service's logging configuration:

```
$ curl -sg -H "Fastly-Key:$token" \
  "https://api.fastly.com/service/:SERVICE_ID/logging_status"
```

The output will indicate whether Fastly has detected an error. If `BrokenNow` is set to `false`, Fastly hasn't detected a problem with your logging configuration:

```
{"1234567890ABCDEF/my-service":{"LastErrorTime":null,"LastError":null,"BrokenNow":false}
```

If an error in the Fastly web interface suggests that your logging configuration appears to be broken for the currently activated service version but you're still receiving some logs, not all of Fastly's log aggregators may be able to connect to your endpoint's server. It's likely the maximum number of concurrent connections has been reached. Try configuring your logging endpoint's server to allow a higher maximum number of inbound connections and then see if the error clears up after a couple of hours.

* * *

Useful conditions for logging

/en/guides/useful-conditions-for-logging

In addition to the standard logging directives, the following conditions can be used for logging when you set up remote log streaming.

IMPORTANT

Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

Logging errors only

You can log errors only if you want a general purpose log that catches everything and a more detailed log if there’s an error:

```
fastly_info.state == "ERROR"
```

You can also log only 500 errors:

```
resp.status >= 500 && resp.status < 600`
```

Logging only specific URLs using a dictionary

Using a dictionary (e.g., `urls_to_log`), you can log specific URLs having issues:

```
table.lookup(urls_to_log, req.url.path) == "log"
```

If a URL becomes a problem, you can start logging it by using the API to add the URL's path to the dictionary as a key with the value "log".

Logging samples

If you have a high-volume service, you might want to log only a proportion of requests by using the `randombool` VCL function in a condition. The following example will log only one percent of all requests:

```
randombool(1,100)
```

You could combine that with a dictionary to change the percentage of requests logged without having to deploy a new version of your service. The following example uses a dictionary named `service_variables`:

```
randombool(std.atoi(table.lookup(service_variables, "logging_percentage", "0")), 100)
```

In the example above, if the key `logging_percentage` doesn't exist, nothing will be logged.

Using `false` to construct a log string in custom VCL

To construct a log string in custom VCL, you can use the `false` condition. This condition makes sure nothing is sent to Fastly logging objects.

* * *

Useful log formats

/en/guides/useful-log-formats

Different systems have standardized on different logging formats over time. Fastly believes logging should be as customizable as possible, working with whichever infrastructure you already have in place. This guide details some of the more complicated examples and custom logging strings (e.g., JSON, Key/Value, CSV, and URL-encoded) you can use to implement the logging formats mentioned in the Apache logging module.

NOTE

Fastly provides two versions of custom log formats. The version 2 logging formats, used by default when you create a new logging endpoint, improve compatibility with Apache's logging directives.

IMPORTANT

Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

TIP

You can log any Varnish variable or Fastly's extensions to VCL. Consider reading our guide to useful variables to log.

Common Log Format (CLF)

```
%h %l %u %t "%r" %>s %b
```

This is the default for many of our logging providers.

Common Log Format with Virtual Host

```
%v %h %l %u %t "%r" %>s %b
```

NCSA extended/combined log format

```
%h %l %u %t "%r" %>s %b "%{Referer}i" "%{User-agent}i"
```

Referer log format

```
%{Referer}i -> %U
```

Agent (Browser) log format

```
%{User-agent}i
```

Custom Tags to Loggly or RFC 5424 to another provider

You'll have to create a regular Syslog logging object pointing at your RFC 5424 compatible endpoint. For example, to send custom tags to Loggly, create a new Syslog object and set `hostname` to `logs-01.loggly.com`, `port` to `6514`, `use_tls` to `true`, and the `message format` field to `blank`. You should also make sure the `token` field is blank. Then, in the `format` field, you would put the following:

```
<134>1 %{%Y-%m-%dT%TZ}t %{server.datacenter}V <log name> - - [<token>@<PEN> tag="fastly" ag
```

The various fields you need to replace are:

- *log name* - this can be whatever you want but we recommend using the same name as you've used for the logging object in Fastly.
- *token* - the private token for your RFC5424 endpoint (if sending to Loggly this is your Customer Token).
- *PEN* - this is a Private Enterprise Number. For example the Loggly PEN (Private Enterprise Number) is 41058. If you want to send to another provider then you can look up their PEN on the IANA registry and use that.
- *tags* - these can be any key/value pairs you want. Two appear in the above example: `fastly` and `other-tag`. Valid tag values include all alpha-numeric characters, the dash (`-`), the period (`.`), and the underscore (`_`). Tag values with spaces aren't valid and will be dropped by many logging providers such as Loggly. Additional information about tags, their restrictions, and details about how Loggly parses tags, can be found in the Loggly documentation but is useful information for sending data to all RFC 5424 compatible endpoints.
- *regular format string* - this is regular Fastly logging directives, put whatever you want here (for example: the Common Log Format mentioned above).

Structured data

The examples below demonstrate different representations of the same variables and variable types:

Name	VCL Value	Type	Description
Protocol	`req.protocol`	string	The HTTP protocol version.
Epoch Seconds	`time.start.sec`	number	The time at the start of the request in seconds.
Start Time	`begin:%Y-%m-%dT%H:%M:%S%z`	time	The time at the start of the request in ISO 8601. format
User Agent	`req.http.User-Agent`	escaped string	The User-Agent request header.
Is IPv6	`req.is_ipv6`	boolean	Whether the request was made over IPv6 or not.
ID	`deadbeef`	literal string	A generic ID.
Some String	`dwayne "the rock" johnson`	escaped literal string	A string with quotation marks in it.
Version	`1.1`	literal number	A generic version number.

JSON

This example JSON may need to be adjusted for your specific logging endpoint. Be sure to check your logging endpoint provider's documentation for the exact format needed.

```
{
  "protocol" : "%H",
  "epoch_seconds" : %{time.start.sec}V,
  "time_start" : "%{begin:%Y-%m-%dT%H:%M:%S%z}t",
  "user_agent" : "%{User-Agent}i",
  "is_ipv6" : %{if(req.is_ipv6, "true", "false")}V,
  "some_string":"%{json.escape(\{"dwayne "the rock" johnson"\})}V",
  "id" : "deadbeef",
  "version" : 1.1
}
```

CSV

```
%H, %{time.start.sec}V, %{begin:%Y-%m-%dT%H:%M:%S%z}t, %{regsub(req.http.User-Agent, \{"
```

Key/Value

```
protocol:%H, epoch_seconds:%{time.start.sec}V, time_start:%{begin:%Y-%m-%dT%H:%M:%S%z}t,
```

URL Encoded

```
protocol=%H&epoch_seconds=%{time.start.sec}V&time_start=%{begin:%Y-%m-%dT%H:%M:%S%z}t&us
```

* * *

Useful variables to log

/en/guides/useful-variables-to-log

In addition to the standard logging directives, the following request and response variables can be used for logging when you set up remote log streaming. You can also log any Varnish variable. Consider taking advantage of some of Fastly's extensions to VCL as well.

NOTE

All variables should be prefixed by a percent sign (`%`). For more information about string formatting, check out our guide to custom log formats.

IMPORTANT

Be sure to take into account security, privacy, and compliance requirements when making configuration and endpoint decisions for the data you intend to include in streamed logs.

Time-related logging variables

These are the time-related variables that can be used for logging.

Variable	Description
`%{begin:%Y-%m-%dT%H:%M:%S%z}t`	The time of the start of the request in ISO 8601 format.
`%{end:%Y-%m-%dT%H:%M:%S%z}t`	The time of the end of the request in ISO 8601 format.
`%{time.elapsed.usec}V`	How long the request took in microseconds.
`%{time.start.sec}V`	When the request started in Epoch seconds.

Connection-related logging variables

These are the connection-related variables that can be used for logging.

Variable	Description
`%{if(req.is_ipv6, "true", "false")}V`	Whether the request was over IPv6 or not.
`%{if(req.is_ssl, "true", "false")}V`	Whether the request was over HTTPS or not.
`%{cstr_escape(tls.client.protocol)}V`	Which version of TLS was used by the client.
`%{cstr_escape(tls.client.servername)}V`	Which SNI server name the client sent.
`%{cstr_escape(tls.client.cipher)}V`	Which cipher the TLS request used.
`%{cstr_escape(tls.client.ciphers_sha)}V`	Which cipher the TLS request used.
`%{cstr_escape(tls.client.tlsexts_sha)}V`	A SHA of the TLS extension identifiers sent from the client as part of the TLS handshake, represented in Base64.
`%{if(fastly_info.is_h2, "true", "false")}V`	Whether or not this was an HTTP/2 request.
`%{if(fastly_info.h2.is_push, "true", "false")}V`	Whether or not this was an HTTP/2 Push response.
`%{fastly_info.h2.stream_id}V`	What the HTTP/2 Stream ID was.

Request- and response-related logging variables

These are the request- and response-related variables that can be used for logging.

Variable	Description
`%{Fastly-Orig-Host}i`	The original Host requested if a Host header override is present.
`%{Host}i`	The current Host request header (because it could have been modified to send to the origin).
`%{Referer}i`	The Referer request header. Specifically, which URL linked to this page.
`%{User-Agent}i`	The User-Agent request header. Specifically, which browser requested this page.

Variable	Description
`%{Accept}i`	The Accept request header. Specifically, the types of content the client can accept.
`%{Accept-Language}i`	The Accept-Language request header. Specifically, the human languages the client can respond with.
`%{Accept-Encoding}i`	The Accept-Encoding request header. Specifically, the content encoding the client is able to understand.
`%{Accept-Charset}i`	The Accept-Charset request header. Specifically, the character set encodings the client accepts.
`%{Connection}i`	The Connection request header. Specifically, whether or not the client can do keep-alive connections.
`%{DNT}i`	The DNT request header. Specifically, whether or not the client is sending a "Do Not Track" header.
`%{Forwarded}i`	The Forwarded request header. Specifically, the originating IP address of a request if this request is proxied.
`%{Via}i`	The Via request header. Specifically, the intermediate protocols and recipients between the user agent and the server on proxied requests.
`%{X-Requested-With}i`	The X-Requested-With request header. Generally used to identify Ajax requests that will send the value XMLHttpRequest.
`%{X-Requested-For}i`	The X-Requested-For request header. Specifically, the originating IP address of a request if this request is proxied.
`%{X-ATT-DeviceId}i`	The X-ATT-DeviceId request header. Specifically, the make, mode, or firmware of AT&T devices.
`%{Content-Type}o`	The Content-Type response header. Specifically, the MIME type of the content.
`%{TSV}o`	The TSV response header. Specifically, the Tracking Status Value suggested for sending in response to a DNT request.

Cache-related logging variables

These are the cache-related variables that can be used for logging.

Variable	Description
`%{If-Modified-Since}i`	The If-Modified-Since request header. Specifically, the server will send back the requested resource, with a 200 status, only if it has been last modified after the given date.
`%{If-None-Match}i`	The If-None-Match request header. Specifically, the server will send back the requested resource, with a 200 status, only if it doesn't have an ETag matching the given ones.

Variable	Description
`%{Cache-Control}o`	The Cache-Control response header. Specifically, whether or not all caching mechanisms from server to client may cache this object in seconds.
`%{Age}o`	The Age response header. Specifically, the age the object has been in a proxy cache in seconds.
`%{Expires}o`	The Expires response header. Specifically, the date and time after which the response is considered stale in "HTTP-date" format as defined by RFC 7231.
`%{Last-Modified}o`	The Last-Modified response header. Specifically, the last modified date for the requested object in "HTTP-date" format as defined by RFC 7231. Used in conjunction with the If-Modified-Since request header.
`%{ETag}o`	The ETag response header. Specifically, an identifier for a specific version of a resource. Used in conjunction with the If-None-Match Request header.
`%{obj.hits}V`	The number of hits this object has (cache specific).
`%{obj.lastuse}V`	The last time this object was used (cache specific).
`"cache_status":"%{fastly_info.state}V"`	(Fastly-specific) State of the request, with optional suffixes describing special cases.

Geographic logging variables

These are the geographic variables that can be used for logging.

Variable	Description
`%{server.datacenter}V`	Which Fastly data center this request hit.
`%{client.geo.city}V`	Which city Fastly thinks the request originated from.
`%{client.geo.city.ascii}V`	An alias of `client.geo.city`.
`%{client.geo.city.utf8}V`	The city or town name associated with the IP address, encoded using the UTF-8 character encoding.
`%{client.geo.country_code}V`	Which country Fastly thinks the request originated from.
`%{client.geo.continent_code}V`	Which continent Fastly thinks the request originated from.
`%{client.geo.region}V`	Which region Fastly thinks the request originated from.

Size-related logging variables

These are the size-related variables that can be used for logging.

Variable	Description
`%{req.header_bytes_read}V`	The size of the request headers.

Variable	Description
`%{req.body_bytes_read}V`	The size of the request body.
`%{resp.header_bytes_written}V`	The size of the response headers.
`%{resp.body_bytes_written}V`	The size of the response body.

Socket-related logging variables

These are the socket-related variables that can be used for logging.

Variable	Description
`%{client.socket.cwnd}V`	The client socket congestion window.
`%{client.socket.nexthop}V`	The IP address of the next gateway.
`%{client.socket.tcpi_rcv_mss}V`	The client socket max segment size for receiving.
`%{client.socket.tcpi_snd_mss}V`	The client socket max segment size for sending.
`%{client.socket.tcpi_rtt}V`	The client socket smoothed round-trip time in microseconds.
`%{client.socket.tcpi_rttvar}V`	The client socket round-trip time variance in microseconds.
`%{client.socket.tcpi_rcv_rtt}V`	The client socket receiver-side estimation of round-trip time in microseconds.
`%{client.socket.tcpi_rcv_space}V`	The current buffer space available for receiving data.
`%{client.socket.tcpi_last_data_sent}V`	The time since last data sent on client socket in microseconds.
`%{client.socket.tcpi_total_retrans}V`	The total number of packet retransmissions on the client socket.
`%{client.socket.tcpi_delta_retrans}V`	The change in number of packet retransmissions on the client socket.
`%{client.socket.ploss}V`	The client socket packet loss.

* * *

Category: Logging endpoints

These articles describe Fastly's support for protocols that allow you to stream logs to a variety of locations, including third-party services, for storage and analysis.

Log streaming: Amazon Kinesis Data Streams

🔗 /en/guides/log-streaming-amazon-kinesis-data-streams

Fastly's Real-Time Log Streaming feature can send log files to Amazon Kinesis Data Streams. Amazon Kinesis Data Streams (KDS) is a real-time data streaming service that can continuously capture data from a variety of sources.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

How Amazon Kinesis Data Streams work with Fastly log streaming

Amazon KDS sends data records to a *stream*. Each stream comprises one or more shards. A shard represents a fixed amount of processing capacity and the total processing capacity of a stream is determined by the number of shards. The number of shards may be increased or decreased over the lifetime of a stream. This is important because the Fastly Kinesis logging endpoint monitors the number of shards and attempts to uniformly distribute the log data records across the available shards. When the number of shards for a stream changes, the Fastly Kinesis logging endpoint automatically adjusts in response. The goal is to make the best use of the throughput capability of the stream while minimizing the configuration overhead required for our customers.

If the log volume exceeds the throughput capacity of the stream, Amazon KDS will return errors to our system that indicate that the stream is being throttled and that may prevent some logs from being delivered. AWS CloudWatch provides a metric for Kinesis Data Streams, `WriteProvisionedThroughputExceeded`, that can be used to monitor this so that adjustments to the stream capacity can be made as necessary.

> TIP
>
> For more information about working with Amazon KDS and understanding the capacity limits, refer to the Kinesis Developer Guide.

Prerequisites

Before adding Amazon KDS as a logging endpoint for Fastly services, we recommend creating Identity and Access Management (IAM) credentials in your AWS account specifically for Fastly. Our recommended way for doing this is by creating an AWS IAM role, which lets you grant temporary credentials. For more information, see Creating an AWS IAM Role for Fastly Logging. Alternatively, create an IAM user and grant the user `kinesis:PutRecords` and `kinesis:ListShards` permissions for the logging stream. For more information, see Amazon's guidance on understanding and getting your AWS credentials.

Adding Amazon Kinesis as a logging endpoint

After you've registered for an AWS account and created an IAM user in Amazon Kinesis, follow these instructions to add Amazon KDS as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Amazon Kinesis Data Streams area, click **Create endpoint**.
3. Fill out the **Create an Amazon Kinesis Data Streams endpoint** fields as follows:

- In the **Name** field, enter a human-readable name for the endpoint.
- In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
- In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
- In the **Access method** field, select either **User Credentials** or **IAM Role**.
- If you select **User Credentials**, enter the access key and secret key associated with the IAM user you created in your AWS account specifically for Fastly. Check out Amazon's documentation on security credentials for more information.

> NOTE
>
> Password management software may mistakenly treat the **Secret Key** field as a password field because of the way your web browser works. As such, that software may try to auto-fill this field with your Fastly account password. If this happens to you, the AWS integration with Fastly services won't work and you will need to enter **Secret Key** manually instead.

- If you select **IAM Role**, enter the Amazon Resource Name (ARN) for the IAM role granting Fastly access to KDS. For more information, check out Creating an AWS IAM Role for Fastly Logging.
- In the **Stream name** field, enter the name of the Kinesis stream to which log data will be sent.
- From the **Region** menu, select the region to stream logs to. This must match the region where you created your Kinesis stream.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Amazon KDS. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Amazon Kinesis Data Streams area, click **Create endpoint**.
3. Fill out the **Create an Amazon Kinesis Data Streams endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Access method** field, select either **User Credentials** or **IAM Role**.
    - If you select **User Credentials**, enter the access key and secret key associated with the IAM user you created in your AWS account specifically for Fastly. Check out Amazon's documentation on security credentials for more information.

    NOTE

    Password management software may mistakenly treat the **Secret Key** field as a password field because of the way your web browser works. As such, that software may try to auto-fill this field with your Fastly account password. If this happens to you, the AWS integration with Fastly services won't work and you will need to enter **Secret Key** manually instead.

    - If you select **IAM Role**, enter the Amazon Resource Name (ARN) for the IAM role granting Fastly access to KDS. For more information, check out Creating an AWS IAM Role for Fastly Logging.
    - In the **Stream name** field, enter the name of the Kinesis stream to which log data will be sent.
    - From the **Region** menu, select the region to stream logs to. This must match the region where you created your Kinesis stream.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Amazon S3

/en/guides/log-streaming-amazon-s3

Fastly's Real-Time Log Streaming feature can send log files to Amazon Simple Storage Service (Amazon S3). Amazon S3 is a static file storage service used by developers and IT teams. You can also use the instructions in this guide to configure log streaming to another S3-compatible service.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Amazon S3 as a logging endpoint for Fastly services, we recommend creating Identity and Access Management (IAM) credentials in your AWS account specifically for Fastly. Our recommended way for doing this is by creating an AWS IAM role, which lets you grant temporary credentials. For more information, see Creating an AWS IAM Role for Fastly Logging. Alternatively, create an IAM user and grant the user `s3:PutObject` permissions for the logging stream. For more information, see Amazon's guidance on understanding and getting your AWS credentials.

Adding Amazon S3 as a logging endpoint

After you've registered for an Amazon S3 account and created an IAM user in Amazon S3, follow these instructions to add Amazon S3 as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Amazon Web Services S3 area, click **Create endpoint**.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Amazon S3 bucket in which to store the logs.
    - In the **Access method** field, select either **User Credentials** or **IAM Role**.
    - If you select **User Credentials**, enter the access key and secret key associated with the IAM user you created in your AWS account specifically for Fastly. Check out Amazon's documentation on security credentials for more information.

> NOTE
>
> Password management software may mistakenly treat the **Secret Key** field as a password field because of the way your web browser works. As such, that software may try to auto-fill this field with your Fastly account password. If this happens to you, the AWS integration with Fastly services won't work and you will need to enter **Secret Key** manually instead.

- If you select **IAM Role**, enter the Amazon Resource Name (ARN) for the IAM role granting Fastly access to S3. For more information, see Creating an AWS IAM Role for Fastly Logging.
- *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any

previously created file object. This value defaults to `3600` seconds. Use the **Period** setting in conjunction with a known average log rate to approximate S3 log file objects of a preferred, uncompressed size. Size may fluctuate depending on actual log volume.

4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - *(Optional)* In the **Domain** field, enter the domain of the Amazon S3 endpoint. If your Amazon S3 bucket was not created in the US Standard region, you must set the domain to match the appropriate endpoint URL. Use the table in the S3 section of the Regions and Endpoints Amazon S3 documentation page. To use an S3-compatible storage system (such as DreamHost's DreamObjects), set the domain to match the domain name for that service (for example, in the case of DreamObjects, the domain name would be `objects.dreamhost.com`).
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
    - From the **Redundancy level** menu, select a setting. This value defaults to **Standard**. Amazon's Using Reduced Redundancy Storage Guide provides more information on using reduced redundancy storage.
    - *(Optional)* From the **ACL** menu, select an access control header. See Amazon's Access Control List (ACL) Specific Request Headers for more information.
    - *(Optional)* **Server side encryption** area, select an encryption method to protect files that Fastly writes to your Amazon S3 bucket. Valid values are **None**, **AES-256**, and **AWS Key Management Service**. If you select **AWS Key Management Service**, you'll have to provide an AWS KMS Key ID. See Amazon's guide on protecting data using server-side encryption for more information.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum file size in bytes. This value caps the file size and overrides any configured period if the size threshold is exceeded prior to the end of the period. If set to 0, no maximum is set and the file size will not be capped.

    > NOTE
    >
    > The minimum file size is 1048576 bytes (1 MiB).

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

> NOTE
>
> Although Fastly continuously streams logs into Amazon S3, the Amazon S3 website and API do not make files available for access until after their upload is complete.

Example format

The following is an example format string for sending data to Amazon S3. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Amazon Web Services S3 area, click **Create endpoint**.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Amazon S3 bucket in which to store the logs.
    - In the **Access method** field, select either **User Credentials** or **IAM Role**.
    - If you select **User Credentials**, enter the access key and secret key associated with the IAM user you created in your AWS account specifically for Fastly. Check out Amazon's documentation on security credentials for more information.

> NOTE
>
> Password management software may mistakenly treat the **Secret Key** field as a password field because of the way your web browser works. As such, that software may try to auto-fill this field with your Fastly account password. If this happens to you, the AWS integration with Fastly services won't work and you will need to enter **Secret Key** manually instead.

- If you select **IAM Role**, enter the Amazon Resource Name (ARN) for the IAM role granting Fastly access to S3. For more information, check out Creating an AWS IAM Role for Fastly Logging.
- *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds. Use the **Period** setting in conjunction with a known average log rate to approximate S3 log file objects of a preferred, uncompressed size. Size may fluctuate depending on actual log volume.

4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
   - *(Optional)* In the **Domain** field, enter the domain of the Amazon S3 endpoint. If your Amazon S3 bucket was not created in the US Standard region, you must set the domain to match the appropriate endpoint URL. Use the table in the S3 section of the Regions and Endpoints Amazon S3 documentation page. To use an S3-compatible storage system (such as DreamHost's DreamObjects), set the domain to match the domain name for that service (for example, in the case of DreamObjects, the domain name would be `objects.dreamhost.com`).
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
   - From the **Redundancy level** menu, select a setting. This value defaults to **Standard**. Amazon's Using Reduced Redundancy Storage Guide provides more information on using reduced redundancy storage.
   - *(Optional)* From the **ACL** menu, select an access control header. See Amazon's Access Control List (ACL) Specific Request Headers for more information.
   - *(Optional)* In the **Server side encryption** area, select an encryption method to protect files that Fastly writes to your Amazon S3 bucket. Valid values are **None**, **AES-256**, and **AWS Key Management Service**. If you select **AWS Key Management Service**, you'll have to provide an AWS KMS Key ID. See Amazon's guide on protecting data using server-side encryption for more information.
   - *(Optional)* In the **Maximum bytes** field, enter the maximum file size in bytes. This value caps the file size and overrides any configured period if the size threshold is exceeded prior to the end of the period. If set to 0, no maximum is set and the file size will not be capped.

> NOTE
>
> The minimum file size is 1048576 bytes (1 MiB).

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

NOTE

Although Fastly continuously streams logs into Amazon S3, the Amazon S3 website and API do not make files available for access until after their upload is complete.

Recommended log format

Log messages can take on any format you choose as long as they can be processed from Amazon S3.

* * *

Log streaming: Microsoft Azure Blob Storage

/en/guides/log-streaming-azure-blob-storage

Fastly's Real-Time Log Streaming feature can send log files to Microsoft Azure Blob Storage (Blob Storage). Blob Storage is a static file storage service used to control arbitrarily large amounts of unstructured data and serve them to users over HTTP and HTTPS.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Blob Storage as a logging endpoint for Fastly services, create an Azure storage account in the Azure portal. For help creating the account, check out Microsoft's account creation documentation.

We recommend creating a Shared Access Signature (SAS) user specifically for Fastly. For more information, check out Microsoft's shared access signatures (SAS) documentation, paying specific attention to the Account SAS URI examples.

Here is an example of a SAS token that provides write permissions to a blob:

`sv=2018-04-05&ss=b&st=2018-04-29T22%3A18%3A26Z&sr=b&se=2020-04-30T02%3A23%3A26Z&sp=w&sig=Z%2FRHIX5Xcg0Mq2rqI3OlWTjEg2tYkboXr1P9ZUXDtkk%3D`

The table breaks down each part of the token to understand how it contributes to the SAS:

Element	Example	Description
`sv`	`sv=2018-04-05`	Storage services version.
`ss`	`ss=b`	The `signedservice` field. This is required and should be `b` for "blob storage."
`st`	`st=2018-04-29T22%3A18%3A26Z`	The start time of the token, specified in UTC.
`sr`	`sr=b`	Store resources for which this token has access. We require

Element	Example	Description
		blob (`b`).
`se`	`se=2020-04-30T02%3A23%3A26Z`	The expiry time of the token, specified in UTC. Ensure you update your token before it expires or the logging functionality will not work.
`sp`	`sp=w`	The permissions granted by the SAS token. We require write (`w`).
`sig`	`sig=Z%2FRHIX5Xcg0Mq2...`	The signature to authorize access to the blob.

Adding Blob Storage as a logging endpoint

After you've registered for an Azure account and created a SAS token, follow these instructions to add Blob Storage as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Azure Blob Storage area, click **Create endpoint**.
3. Fill out the **Create a Microsoft Azure Blob Storage endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, enter a string formatted as a comma-separated value (CSV) to use for log formatting. See Ingesting data for Azure Data Explorer for more information.
    - In the **Storage account name** field, enter the unique Azure namespace in which your data objects will be stored.
    - In the **Container** field, enter the name of the Blob Storage container to store logs in. Check out Microsoft's Blob storage page for more information.
    - In the **SAS token** field, enter the token associated with the container.

    > TIP
    >
    > Ensure you update your token before it expires otherwise the logging functionality will not work.

    - *(Optional)* In the **Maximum bytes** field, enter the maximum file size in bytes.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
4. Click **Advanced options** and fill out the fields as follows:

  - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
  - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
  - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

> NOTE
>
> Although Fastly continuously streams logs into Azure Blob Storage, the storage portal and API do not make files available for access until after their upload is complete.

Ingesting data for Azure Data Explorer

Azure Data Explorer is a data exploration service for log and telemetry data. To ingest your data correctly, Data Explorer requires your logs to be formatted as comma-separated values (CSVs). When creating your logging endpoint:

- Set the **Log format** to a CSV string (`%H,%{time.start.sec}V,%{regsub(req.http.User-Agent, \{"""\}, \{""""\})}V`).
- Specify **blank** when you **Select a log line format** in the **Advanced options**.

Our guide on changing log line formats provides more information.

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Azure Blob Storage area, click **Create endpoint**.
3. Fill out the **Create a Microsoft Azure Blob Storage endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
   - In the **Storage account name** field, enter the unique Azure namespace in which your data objects will be stored.
   - In the **Container** field, enter the name of the Blob Storage container to store logs in. See Microsoft's Blob storage page for more information.
   - In the **SAS token** field, enter the token associated with the container.

   > TIP

Ensure you update your token before it expires otherwise the logging functionality will not work.

  - *(Optional)* In the **Maximum bytes** field, enter the maximum file size in bytes.
  - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
  - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.

4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

NOTE

Although Fastly continuously streams logs into Azure Blob Storage, the storage portal and API do not make files available for access until after their upload is complete.

Recommended log format

Log messages can take on any format you choose as long as they can be processed from Azure Blob Storage.

Ingesting data for Azure Data Explorer

Azure Data Explorer is a data exploration service for log and telemetry data. To ingest your data correctly, Data Explorer requires your logs to be formatted as comma-separated values (CSVs). When creating your logging endpoint:

- Set the log format to a CSV string (`${time_start_sec},"${req_http_user_agent}","${log_message}"`). In this example, `${time_start_sec}`, `${req_http_user_agent}`, and `${log_message}` are placeholders for variables in your Compute service code.
- Specify **blank** when you **Select a log line format** in the **Advanced options**.

Our guide on changing log line formats provides more information.

* * *

Log streaming: Cloud Files

/en/guides/log-streaming-cloudfiles

Fastly's Real-Time Log Streaming feature can send log file to Cloud Files. Operated by Rackspace, Cloud Files is a file storage service used by developers and IT teams.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

If you don't already have a Rackspace Cloud account, you'll need to register for one. Follow the instructions on Rackspace's website.

Creating a Cloud Files user and container

Start by creating a Cloud Files user with restricted permissions via Rackspace's cloud control panel.

1. Log in to Rackspace's cloud control panel.
2. From the user account menu, select **User Management**.
3. Click **Create User** and fill in all appropriate details.
4. In the **Product Access** section, set **User Role** to **Custom**.
5. Review the **Product Access** list. For all items in the **Product** column, set **Role** to **No Access** except the **Files** item.
6. Set the **Files** item **Role** to **Admin**. This will allow you to create the files to store the logs in, but not access any other services.

Next, find the API key for your Cloud Files account. You'll use this later to authenticate using the Cloud Files API.

1. From the user account menu, select **Account Settings**.
2. Show the API key in the **Login details** and make a note of it.

Now that you've created the Cloud Files user and found the API key, you can set up a Cloud Files container.

1. From the **Storage** menu, select **Files**.
2. Click **Create Container**.
3. Assign the container a meaningful name like `Fastly logs - my service`.
4. Choose a region to keep the files in and make sure the container is private.

5. Click **Create Container**.

Adding a Cloud Files logging endpoint

Once you have created the Cloud Files user and container, follow these instructions to add Cloud Files as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Rackspace Cloud Files area, click **Create endpoint**.
3. Fill out the **Create a Cloud Files endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Cloud Files container in which to store the logs.
    - In the **User** field, enter the username of the Cloud Files user you created above.
    - In the **Access key** field, enter the API key of your Cloud Files account.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
    - From the **Region** menu, select the region to stream logs to.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Cloud Files. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Rackspace Cloud Files area, click **Create endpoint**.
3. Fill out the **Create a Cloud Files endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Cloud Files container in which to store the logs.
    - In the **User** field, enter the username of the Cloud Files user you created above.
    - In the **Access key** field, enter the API key of your Cloud Files account.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.

  - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
  - From the **Region** menu, select the region to stream logs to.
  - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Coralogix

/en/guides/log-streaming-coralogix

Fastly's Real-Time Log Streaming feature can send log files to Coralogix. Coralogix provides an analytics platform that allows you to detect abnormal behavior via dynamic alerts, ratio alerts, flow anomaly detection, and threat discovery.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

If you don't already have a Coralogix account, you'll need to register for one by following the signup instructions on the Coralogix website. Once you've signed up, navigate to the **Send Your Logs** area in the **Settings** section of your Coralogix dashboard and make note of your unique private key. Coralogix uses this to associate data you send them with your account. You'll need it when you set up your endpoint with Fastly.

TIP

Consider reading Coralogix's documentation on integrating with Fastly.

Adding Coralogix as a logging endpoint

Follow these instructions to add Coralogix as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the HTTPS area, click **Create endpoint**.
3. Fill out the **Create an HTTPS endpoint** fields as follows:

- In the **Name** field, enter a human-readable name for the endpoint.
- In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
- In the **Log format** field, replace the placeholder log format and make the appropriate changes as shown in our log format and recommendations section below.
- In the **URL** field, enter the Coralogix REST API Singles Endpoint for your Coralogix Domain. For example, you could use `https://ingress.coralogix.us/logs/v1/singles` for the `coralogix.us` domain.
- In the **Maximum logs** field, leave the default value of `0`.
- In the **Maximum bytes** field, enter `2000000`.

4. Click **Advanced options** and fill out the fields as follows:
    - In the **Content type** field, enter `application/json`.
    - In the **Custom header name** field, enter `private_key`.
    - In the **Custom header value** field, enter your Coralogix private key.
    - From the **Method** controls, select **POST**.
    - From the **JSON log entry format** controls, select **Array of JSON**.
    - Leave the **Select a log line format** controls set to the defaults.
    - Leave the remaining fields blank.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

> NOTE
>
> For Coralogix, you do not need to configure anything to satisfy the HTTPS Proof of domain ownership requirement, and you can safely ignore any warning about it.

Log format and field setting recommendations

Use the following log format:

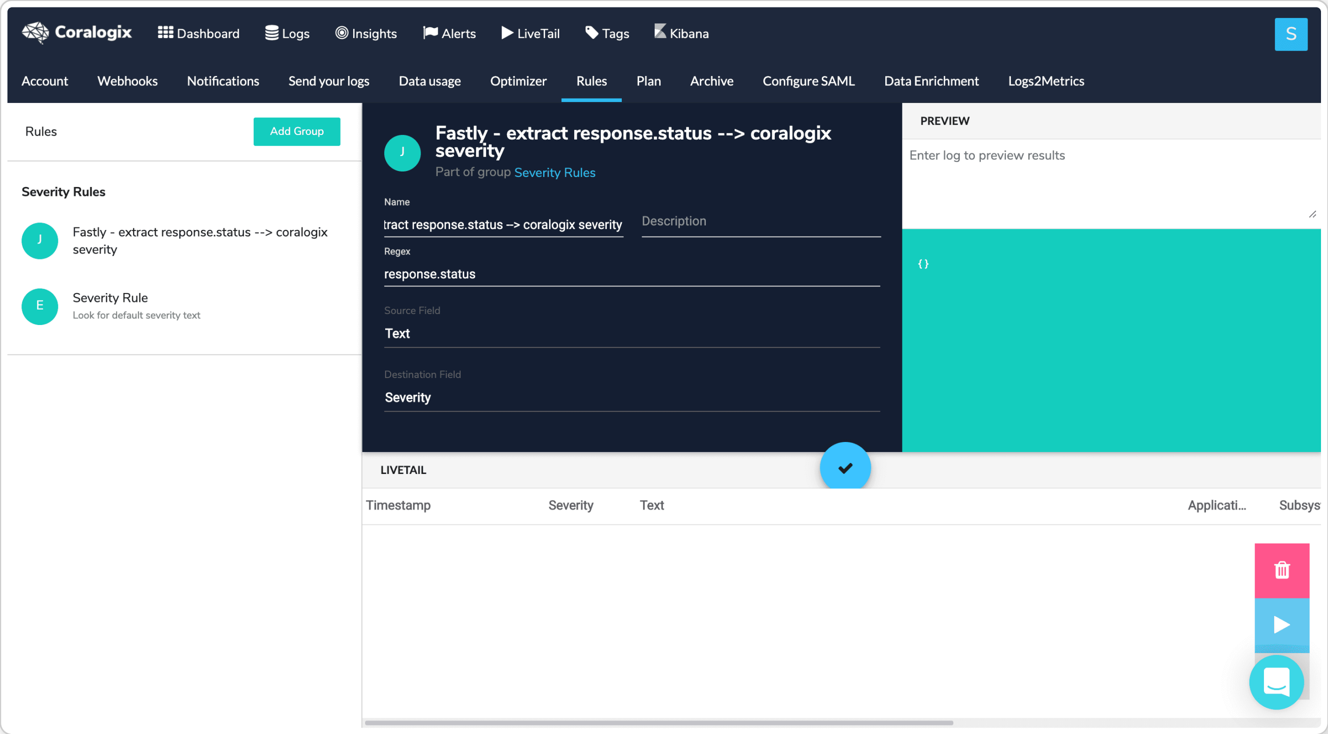
```
{
  "timestamp":%{time.start.msec}V,
  "applicationName":"fastly",
  "subsystemName":"%{req.service_id}V",
  "severity": 3,
  "json": {
    "time": {
        "start":"%{begin:%Y-%m-%dT%H:%M:%S%Z}t",
        "end":"%{end:%Y-%m-%dT%H:%M:%S%Z}t",
        "elapsed":%D
    },
    "cdn_server": {
        "ip_ipaddr":"%A",
        "code":"%{server.datacenter}V",
        "hostname":"%{server.hostname}V",
        "region_code":"%{server.region}V",
        "response_state":"%{fastly_info.state}V",
        "is_h2":%{if(fastly_info.is_h2, "true", "false")}V,
        "is_h2_push":%{if(fastly_info.h2.is_push, "true", "false")}V,
        "h2_stream_id":"%{fastly_info.h2.stream_id}V"
    },
    "client": {
        "city_name":"%{client.geo.city.utf8}V",
        "country_code":"%{client.geo.country_code}V",
        "country_name":"%{client.geo.country_name}V",
        "continent_code":"%{client.geo.continent_code}V",
        "region":"%{client.geo.region}V",
        "ip_ipaddr":"%h",
        "name":"%{client.as.name}V",
        "number":"%{client.as.number}V",
        "connection_speed":"%{client.geo.conn_speed}V",
        "location_geopoint": {
            "lat":%{client.geo.latitude}V,
            "lon":%{client.geo.longitude}V
        }
    },
    "response": {
        "status":%>s,
        "content_type":"%{Content-Type}o",
        "age":"%{Age}o",
        "cache_control":"%{Cache-Control}o",
        "expires":"%{Expires}o",
        "last_modified":"%{Last-Modified}o",
        "tsv":"%{TSV}o",
        "header_size":%{resp.header_bytes_written}V,
        "body_size":%B
    },
    "request": {
        "host":"%{req.http.host}V",
        "is_ipv6":%{if(req.is_ipv6, "true", "false")}V,
        "backend":"%{req.backend}V",
        "service_id":"%{req.service_id}V",
        "url":"%{cstr_escape(req.url)}V",
        "url_ext":"%{req.url.ext}V",
        "header_size":%{req.header_bytes_read}V,
        "body_size":%{req.body_bytes_read}V,
        "method":"%m",
        "protocol":"%H",
        "referer":"%{Referer}i",
        "user_agent":"%{User-Agent}i",
        "accept_content":"%{Accept}i",
        "accept_language":"%{Accept-Language}i",
        "accept_encoding":"%{Accept-Encoding}i",
        "accept_charset":"%{Accept-Charset}i",
        "connection":"%{Connection}i",
        "dnt":"%{DNT}i",
        "forwarded":"%{Forwarded}i",
        "via":"%{Via}i",
        "cache_control":"%{Cache-Control}i",
        "x_requested_with":"%{X-Requested-With}i",
        "x_att_device_id":"%{X-ATT-Device-Id}i",
        "x_forwarded_for":"%{X-Forwarded-For}i"
    },
    "socket": {
        "cwnd":%{client.socket.cwnd}V,
        "pace":%{client.socket.pace}V,
        "nexthop":"%{client.socket.nexthop}V",
        "tcpi_rcv_mss":%{client.socket.tcpi_rcv_mss}V,
        "tcpi_snd_mss":%{client.socket.tcpi_snd_mss}V,
        "tcpi_rtt":%{client.socket.tcpi_rtt}V,
        "tcpi_rttvar":%{client.socket.tcpi_rttvar}V,
        "tcpi_rcv_rtt":%{client.socket.tcpi_rcv_rtt}V,
        "tcpi_rcv_space":%{client.socket.tcpi_rcv_space}V,
        "tcpi_last_data_sent":%{client.socket.tcpi_last_data_sent}V,
        "tcpi_total_retrans":%{client.socket.tcpi_total_retrans}V,
        "tcpi_delta_retrans":%{client.socket.tcpi_delta_retrans}V,
        "ploss":%{client.socket.ploss}V
    }
  }
}
```

The first five fields of the recommended format are required:

- `timestamp`: Leave the format of this field unchanged.
- `applicationName`: Enter the name of the application in this field.
- `subsystemName`: Enter the name of the subsystem in this field. This is used to separate components. We use `req.service_id` in the example, which isn't particularly human readable. Use whatever subsystem name makes sense that helps you identify the subsystem.
- `severity`: Specify the severity and apply it to all logs using the following choices: 1 (debug), 2 (verbose), 3 (info), 4 (warning), 5 (error), 6 (critical). This can be changed later using an extract rule as described below.
- `json (object)`: Add or remove fields as necessary. Static fields can be added. Nested JSON formats are supported including any fields described in the Fastly VCL reference.

  Specifying a nested `response.status` field is a useful way to identify the status for servicing the request. Using the Coralogix parsing rules, you can set a JSON Extract rule to use the status code value from the log to populate the severity field in the Coralogix interface. Specifically, you can automatically map an HTTP status code to a severity value. For example, status code `2xx` will set the Coralogix severity as “**INFO**” and status code `4xx` will set Coralogix severity as “**ERROR**”.

  In the Coralogix web interface, it will look like this:


Coralogix
Dashboard
Logs
Insights
Alerts
LiveTail
Tags
Kibana
S
Account
Webhooks
Notifications
Send your logs
Data usage
Optimizer
Rules
Plan
Archive
Configure SAML
Data Enrichment
Logs2Metrics
Rules
Add Group
Severity Rules
J
Fastly - extract response.status --> coralogix severity
E
Severity Rule
Look for default severity text
PREVIEW
Enter log to preview results
{}
LIVETAIL
Timestamp
Severity
Text
Applicati...
Subsys

Fastly - extract response.status --> coralogix severity

Part of group Severity Rules

Name: tract response.status –> coralogix severity

Description

Regex: response.status

Source Field: Text

Destination Field: Severity

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the HTTPS area, click **Create endpoint**.

3. Fill out the **Create an HTTPS endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **URL** field, enter the Coralogix REST API Singles Endpoint for your Coralogix Domain. For example, you could use `https://ingress.coralogix.us/logs/v1/singles` for the `coralogix.us` domain.
   - In the **Maximum logs** field, leave the default of `0`.
   - In the **Maximum bytes** field, enter `2000000`.

4. Click **Advanced options** and fill out the fields as follows:
   - In the **Content type** field, enter `application/json`.
   - In the **Custom header name** field, enter `private_key`.
   - In the **Custom header value** field, enter your Coralogix private key.
   - From the **Method** controls, select **POST**.
   - From the **JSON log entry format** controls, select **Array of JSON**.
   - Leave the **Select a log line format** controls set to the defaults.
   - Leave the **TLS hostname**, **TLS CA certificate**, **TLS client certificate**, and **TLS client key** fields blank.

5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

NOTE

For Coralogix, you do not need to configure anything explicitly to satisfy the HTTPS Proof of domain ownership requirement, and you can safely ignore any warning about it.

Recommended log format

Data sent to Coralogix must be serialized in a way conforming to Coralogix's expectations. If your logs are not formatted properly, attempts at processing your logs by your Coralogix endpoint may fail. Here's an example format string for sending data to Coralogix:

```
{
    "timestamp": 1653088964764,
    "applicationName": "fastly",
    "subsystemName": "wasm",
    "severity": 3,
    "json": {
        "message": "Request happened",
        "response": {
          "status":200
       }
    }
}
```

You can follow the general JSON structure above regardless of the chosen language for your Compute service. The following fields are required:

- `timestamp`: The format of this field is in milliseconds.

- `applicationName`: Enter the name of the application.

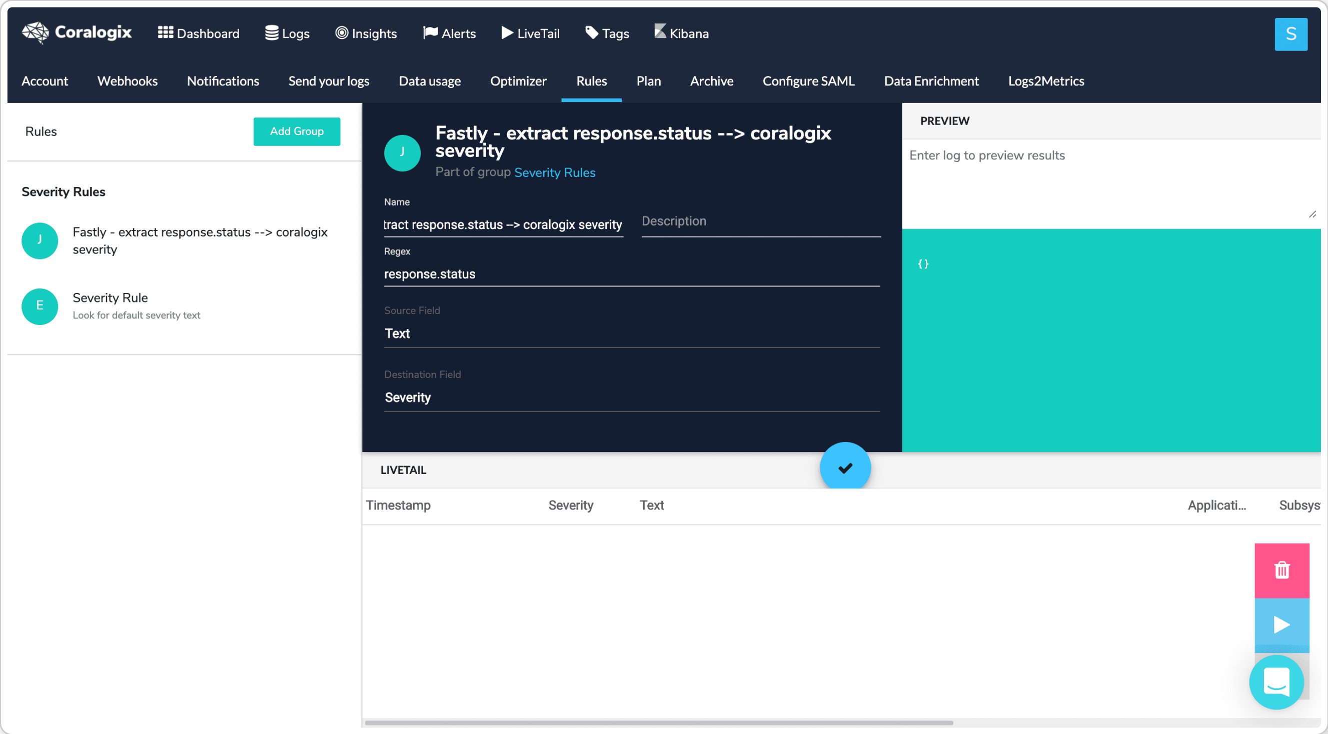
- `subsystemName`: Enter the name of the subsystem. This field is used to separate components. Use whatever subsystem name makes sense that helps you identify the subsystem.

- `severity`: The severity of the log. You can specify the severity to all logs using the following choices: 1 (debug), 2 (verbose), 3 (info), 4 (warning), 5 (error), 6 (critical). This can be changed later using an extraction rule as described in the field below.

- `json (object)`: Used to specify additional log details as necessary. Nested JSON formats are supported.

  Specifying a nested `response.status` field is a useful way to identify the status for servicing the request. Using the Coralogix parsing rules, you can set a JSON Extract rule to use the status code value from the log to populate the severity field in the Coralogix interface. Specifically, you can automatically map an HTTP status code to a severity value. For example, status code `2xx` will set the Coralogix severity as “**INFO**” and status code `4xx` will set Coralogix severity as “**ERROR**”.

  In the Coralogix web interface, it will look like this:


Coralogix
Dashboard
Logs
Insights
Alerts
LiveTail
Tags
Kibana
S
Account
Webhooks
Notifications
Send your logs
Data usage
Optimizer
Rules
Plan
Archive
Configure SAML
Data Enrichment
Logs2Metrics
Rules
Add Group
PREVIEW
Enter log to preview results
{}
LIVETAIL
Timestamp
Severity
Text
Applicati...
Subsys

Severity Rules

J Fastly - extract response.status --> coralogix severity

E Severity Rule
Look for default severity text

Fastly - extract response.status --> coralogix severity
Part of group Severity Rules

Name: tract response.status --> coralogix severity
Description
Regex: response.status
Source Field: Text
Destination Field: Severity

Configuring Coralogix dashboards and alerting

Coralogix provides tutorials for integrating their service with Fastly via dashboards and alerting. This includes examples of data dashboards created using Fastly data, including one for a general service overview, a visitor breakdown, and quality of service.

Their tutorials also describe how to set up user-defined alerts for situations like no logs being received from Fastly, outages at your origin, elevated error ratios and cache misses, unusual or suspicious requests of various types, as well as potential website defacement attempts.

* * *

Log streaming: Datadog

/en/guides/log-streaming-datadog

Fastly's Real-Time Log Streaming feature can be configured to send logs in a format readable by Datadog. Datadog is a cloud-based monitoring and analytics solution that allows you to see inside applications within your stack and aggregate the results.

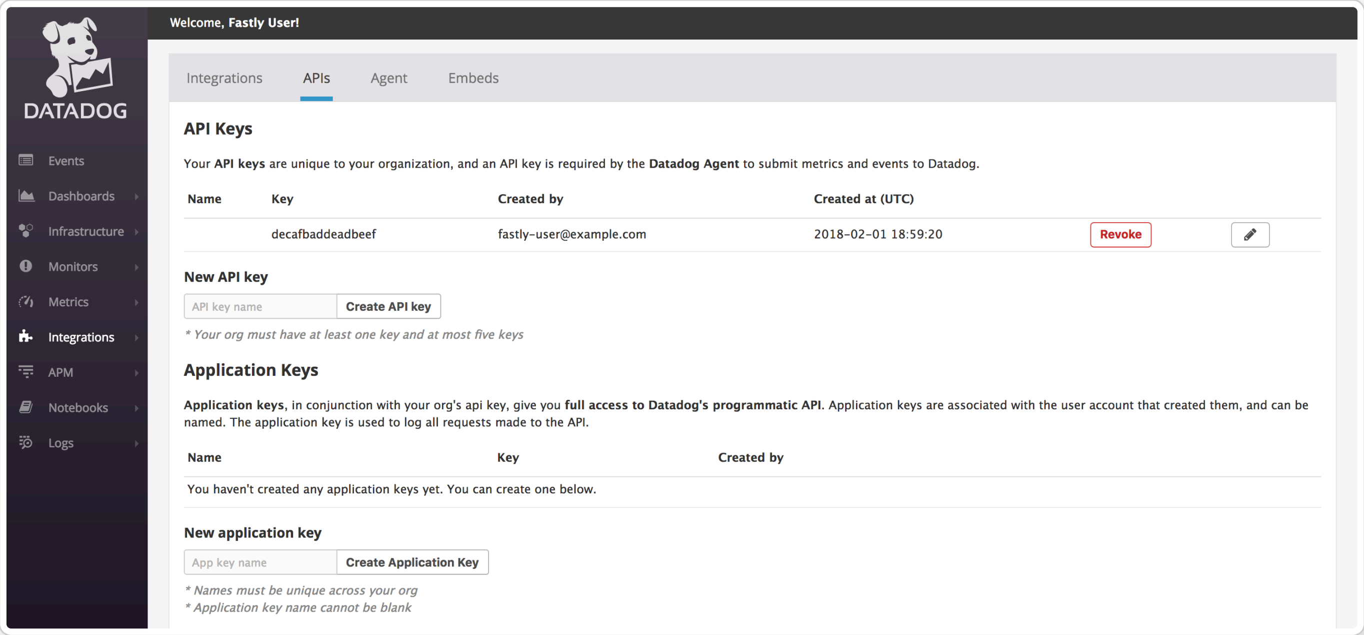
NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Datadog as a logging endpoint for Fastly services, you will need to:

- **Register for a Datadog account.** You can sign up for a Datadog account on their site. A free plan exists that has some restrictions or you can upgrade for more features. Where you register your Datadog setup, either in the United States (US) or the European Union (EU), will affect which commands you use during logging endpoint setup at Fastly.
- **Get your Datadog API key from your settings page on Datadog.** In the Datadog interface, navigate to Integrations → APIs where you'll be able to create or retrieve an API key.



API Keys

Your **API keys** are unique to your organization, and an API key is required by the **Datadog Agent** to submit metrics and events to Datadog.

Name	Key	Created by	Created at (UTC)		
	decafbaddeadbeef	fastly-user@example.com	2018-02-01 18:59:20	Revoke	✎

New API key

API key name

** Your org must have at least one key and at most five keys*

Application Keys

Application keys, in conjunction with your org's api key, give you **full access to Datadog's programmatic API**. Application keys are associated with the user account that created them, and can be named. The application key is used to log all requests made to the API.

Name	Key	Created by
You haven't created any application keys yet. You can create one below.		

New application key

App key name

** Names must be unique across your org*
** Application key name cannot be blank*

This example displays the key `decafbaddeadbeef`. Your API key will be different. Make a note of this key somewhere.

Adding Datadog as a logging endpoint

After you've created a Datadog account and noted your Datadog API key, follow the steps below to add Datadog as a logging endpoint for Fastly services.

1. Review the information in our guide to setting up remote log streaming.
2. In the Datadog area, click **Create endpoint**.
3. Fill out the **Create a Datadog endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **Log format** field, enter the data to send to Datadog. We've described the use of this format below with additional suggestions.
   - From the **Region** menu, select the region to stream logs to.
   - In the **API key** field, enter the API key of your Datadog account.
4. Click **Create** to create the new logging endpoint.

5. Click **Activate** to deploy your configuration changes.

Logs should begin appearing in your Datadog account a few seconds after you've created the endpoint and deployed your service changes. These logs can then be accessed via the Datadog Log Explorer on your Datadog account.

Using the JSON logging format

Data sent to Datadog must be serialized as a JSON object. Datadog automatically parses log files created as JSON, and is able to recognize several reserved fields, such as `service`, `host`, and `date`.

> NOTE
>
> The JSON in this example is formatted for ease of reading. For proper parsing, it must be added as a single line in the **Log format** field, removing all line breaks and indentation whitespace first.

For example, in the JSON below we've set `service` to the ID of the Fastly service that sent the log but you could also use a human-readable name or you could group all logs under a common name such as `fastly`.

```
{
  "ddsource": "fastly",
  "service": "%{req.service_id}V",
  "date": "%{begin:%Y-%m-%dT%H:%M:%S%z}t",
  "time_start": "%{begin:%Y-%m-%dT%H:%M:%S%Z}t",
  "time_end": "%{end:%Y-%m-%dT%H:%M:%S%Z}t",
  "http": {
    "request_time_ms": %{time.elapsed.msec}V,
    "method": "%m",
    "url": "%{json.escape(req.url)}V",
    "useragent": "%{User-Agent}i",
    "referer": "%{Referer}i",
    "protocol": "%H",
    "request_x_forwarded_for": "%{X-Forwarded-For}i",
    "status_code": "%s"
  },
  "network": {
    "client": {
      "ip": "%h",
      "name": "%{client.as.name}V",
      "number": "%{client.as.number}V",
      "connection_speed": "%{client.geo.conn_speed}V"
    },
    "destination": {
      "ip": "%A"
    },
  "geoip": {
  "geo_city": "%{client.geo.city.utf8}V",
  "geo_country_code": "%{client.geo.country_code}V",
  "geo_continent_code": "%{client.geo.continent_code}V",
  "geo_region": "%{client.geo.region}V"
  },
  "bytes_written": %B,
  "bytes_read": %{req.body_bytes_read}V
  },
  "host":"%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "origin_host": "%v",
  "is_ipv6": %{if(req.is_ipv6, "true", "false")}V,
  "is_tls": %{if(req.is_ssl, "true", "false")}V,
  "tls_client_protocol": "%{json.escape(tls.client.protocol)}V",
  "tls_client_servername": "%{json.escape(tls.client.servername)}V",
  "tls_client_cipher": "%{json.escape(tls.client.cipher)}V",
  "tls_client_cipher_sha": "%{json.escape(tls.client.ciphers_sha)}V",
  "tls_client_tlsexts_sha": "%{json.escape(tls.client.tlsexts_sha)}V",
  "is_h2": %{if(fastly_info.is_h2, "true", "false")}V,
  "is_h2_push": %{if(fastly_info.h2.is_push, "true", "false")}V,
  "h2_stream_id": "%{fastly_info.h2.stream_id}V",
  "request_accept_content": "%{Accept}i",
  "request_accept_language": "%{Accept-Language}i",
  "request_accept_encoding": "%{Accept-Encoding}i",
  "request_accept_charset": "%{Accept-Charset}i",
  "request_connection": "%{Connection}i",
  "request_dnt": "%{DNT}i",
  "request_forwarded": "%{Forwarded}i",
  "request_via": "%{Via}i",
  "request_cache_control": "%{Cache-Control}i",
  "request_x_requested_with": "%{X-Requested-With}i",
  "request_x_att_device_id": "%{X-ATT-Device-Id}i",
  "content_type": "%{Content-Type}o",
  "is_cacheable": %{if(fastly_info.state~"^(HIT|MISS)$", "true","false")}V,
  "response_age": "%{Age}o",
  "response_cache_control": "%{Cache-Control}o",
  "response_expires": "%{Expires}o",
  "response_last_modified": "%{Last-Modified}o",
  "response_tsv": "%{TSV}o",
  "server_datacenter": "%{server.datacenter}V",
  "req_header_size": %{req.header_bytes_read}V,
  "resp_header_size": %{resp.header_bytes_written}V,
  "socket_cwnd": %{client.socket.cwnd}V,
  "socket_nexthop": "%{client.socket.nexthop}V",
  "socket_tcpi_rcv_mss": %{client.socket.tcpi_rcv_mss}V,
  "socket_tcpi_snd_mss": %{client.socket.tcpi_snd_mss}V,
  "socket_tcpi_rtt": %{client.socket.tcpi_rtt}V,
  "socket_tcpi_rttvar": %{client.socket.tcpi_rttvar}V,
  "socket_tcpi_rcv_rtt": %{client.socket.tcpi_rcv_rtt}V,
  "socket_tcpi_rcv_space": %{client.socket.tcpi_rcv_space}V,
  "socket_tcpi_last_data_sent": %{client.socket.tcpi_last_data_sent}V,
  "socket_tcpi_total_retrans": %{client.socket.tcpi_total_retrans}V,
  "socket_tcpi_delta_retrans": %{client.socket.tcpi_delta_retrans}V,
  "socket_ploss": %{client.socket.ploss}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our

supported languages.

2. In the Datadog area, click **Create endpoint**.
3. Fill out the **Create a Datadog endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - From the **Region** menu, select the region to stream logs to.
    - In the **API key** field, enter the API key of your Datadog account.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Recommended log format

Data sent to Datadog must be serialized in a way conforming to Datadog's expectations.

If your logs are not formatted properly, attempts at processing your logs by your Datadog endpoint may fail. Here's an example format string for sending data to Datadog:

```
{
  "ddsource": "fastly",
  "ddtags": "env:production,version:1.0",
  "hostname": "hostname",
  "message": "2019-11-19T14:37:58,995 INFO Hello World",
  "service": "service_id"
}
```

You can follow the general JSON structure above regardless of the chosen language for your Compute service and include the specific details inside designated fields, such as `message`. The `ddsource` field is required. Nested JSON objects are supported. Refer to the Datadog documentation for other available options for Datadog log messages. The emitted logs must be formatted as valid JSON.

Logs should begin appearing in your Datadog account a few seconds after you've created the endpoint and deployed your service changes. These logs can then be accessed via the Datadog Log Explorer on your Datadog account.

* * *

Log streaming: DigitalOcean Spaces

/en/guides/log-streaming-digitalocean-spaces

Fastly's Real-Time Log Streaming feature can send log files to DigitalOcean Spaces. DigitalOcean Spaces is an Amazon S3-compatible static file storage service used by developers and IT teams.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding DigitalOcean Spaces as a logging endpoint for Fastly services, you'll need to create a DigitalOcean account if you don't already have one. Then you'll need to create a space with private access permissions on DigitalOcean's website, generate a secret key and an access key, and make a note of the endpoint.

Adding DigitalOcean Spaces as a logging endpoint

After you've created a DigitalOcean Space, follow these instructions to add DigitalOcean Spaces as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Spaces by DigitalOcean area, click **Create endpoint**.
3. Fill out the **Create a DigitalOcean endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Space name** field, enter the name of the DigitalOcean Space in which to store the logs.
    - In the **Access key** field, enter the access key associated with the DigitalOcean Space. Refer to the DigitalOcean Spaces Authentication Guide for more information.
    - In the **Secret key** field, enter the secret key associated with the DigitalOcean Space.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - In the **Domain** field, enter the region-specific endpoint for your domain. In most cases, this should be `nyc3.digitaloceanspaces.com`. If the DigitalOcean Space was not created in the `nyc3` region, refer to DigitalOcean's documentation to find the correct domain.
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your

private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.

- In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
- *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to DigitalOcean. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Spaces by DigitalOcean area, click **Create endpoint**.
3. Fill out the **Create a DigitalOcean endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Space name** field, enter the name of the DigitalOcean Space in which to store the logs.

   - In the **Access key** field, enter the access key associated with the DigitalOcean Space. Check out the DigitalOcean Spaces Authentication Guide for more information.
   - In the **Secret key** field, enter the secret key associated with the DigitalOcean Space.
   - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
   - In the **Domain** field, enter the region-specific endpoint for your domain. In most cases, this should be `nyc3.digitaloceanspaces.com`. If the DigitalOcean Space was not created in the `nyc3` region, refer to DigitalOcean's documentation to find the correct domain.
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Elasticsearch

/en/guides/log-streaming-elasticsearch

Fastly's Real-Time Log Streaming feature can send log files to Elasticsearch. Elasticsearch is a distributed, RESTful search and analytics engine.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Elasticsearch as a logging endpoint for Fastly services, ensure Elasticsearch is running on a remote server. You'll need to know the endpoint URL that includes a port to which logs should be sent (make sure it can receive traffic from Fastly) and also the name of the index to send logs to. For more information on setting up Elasticsearch, check out the Elasticsearch setup documentation.

This logging endpoint works with all actively supported versions of Elasticsearch as well as some versions that have already reached their end-of-life. We also work with OpenSearch server integration. Other distributions that are API-compatible with Elasticsearch may also work but have not been explicitly tested and are not guaranteed.

Required privileges

We send data using the Bulk API via the `index` action. When using basic authentication, ensure that the required index privileges to use the `index` action are granted to the user role.

We also require access to the root path API of the Elasticsearch server. This API returns metadata about the server, such as the version number, that allows our integration to make the best choice about which bulk data API to use for each customer's server. Access to this API allows us to properly work with the wide range of Elasticsearch versions used by our customers as well as other Elasticsearch-compatible distributions.

Adding Elasticsearch as a logging endpoint

Follow these instructions to add Elasticsearch as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Elasticsearch area, click **Create endpoint**.
3. Fill out the **Create an Elasticsearch endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, enter the data to send to Elasticsearch. See the example format section for details.
    - In the **URL** field, enter the Elasticsearch endpoint URL that includes a port to which logs should be sent. The URL must be sent using HTTPS on a port that can receive incoming TCP traffic from Fastly.
    - In the **Index** field, enter the name of the Elasticsearch index to send logs to. The index must follow the Elasticsearch index format rules. We support strftime interpolated variables inside braces prefixed with a pound symbol. For example, `#{%F}` will interpolate as YYYY-MM-DD with today's date.
    - *(Optional)* In the **Pipeline** field, enter the ID of the Elasticsearch ingest pipeline to apply pre-process transformations to before indexing (for example, `my_pipeline_id`).
    - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to append to a batch, if non-zero.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum size of the log batch.
    - *(Optional)* In the **BasicAuth user** field, enter your basic authentication username.
    - *(Optional)* In the **BasicAuth password** field, enter your basic authentication password.
    - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common

Names (CN) are not supported.

- *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the origin server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority.
- *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the origin server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client certificate. A TLS client certificate allows your server to authenticate that Fastly is performing the connection.
- *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the backend server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your server to authenticate that Fastly is performing the connection.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

Data sent to Elasticsearch must be serialized as a JSON object. Here's an example format string for sending data to Elasticsearch:

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Elasticsearch area, click **Create endpoint**.
3. Fill out the **Create an Elasticsearch endpoint** fields as follows:

  - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
  - In the **URL** field, enter the Elasticsearch endpoint URL that includes a port to which logs should be sent. The URL must be sent using HTTPS on a port that can receive incoming TCP traffic from Fastly.
  - In the **Index** field, enter the name of the Elasticsearch index to send logs to. The index must follow the Elasticsearch index format rules. We support strftime interpolated variables inside braces prefixed with a pound symbol. For example, `#{%F}` will interpolate as YYYY-MM-DD with today's date.
  - *(Optional)* In the **Pipeline** field, enter the ID of the Elasticsearch ingest pipeline to apply pre-process transformations to before indexing (for example, `my_pipeline_id`).
  - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to append to a batch.
  - *(Optional)* In the **Maximum bytes** field, enter the maximum size of the log batch.
  - *(Optional)* In the **BasicAuth user** field, enter your basic authentication username.
  - *(Optional)* In the **BasicAuth password** field, enter your basic authentication password.
  - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported.
  - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the origin server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority.
  - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the origin server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client certificate. A TLS client certificate allows your server to authenticate that Fastly is performing the connection.
  - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the backend server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your server to authenticate that Fastly is performing the connection.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: FTP

/en/guides/log-streaming-ftp

Fastly's Real-Time Log Streaming feature can send log files to password-protected and anonymous FTP servers.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Adding FTP as a logging endpoint

Follow these instructions to add FTP as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the FTP area, click **Create endpoint**.
3. Fill out the **Create a File Transfer Protocol (FTP) endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Address** field, enter the hostname or IP address of the FTP server. In the port field, enter the port number you're using for FTP (the default is `21`).
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - In the **User** field, enter the username used to authenticate to the FTP server. For anonymous access, use the username `anonymous`.
    - In the **Password** field, enter the password used to authenticate to the FTP server. For anonymous access, use an email address as the password.
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to an FTP logging endpoint. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the FTP area, click **Create endpoint**.
3. Fill out the **Create a File Transfer Protocol (FTP) endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Address** field, enter the hostname or IP address of the FTP server. In the port field, enter the port number you're using for FTP (the default is `21`).
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - In the **User** field, enter the username used to authenticate to the FTP server. For anonymous access, use the username `anonymous`.
    - In the **Password** field, enter the password used to authenticate to the FTP server. For anonymous access, use an email address as the password.

    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Google BigQuery

/en/guides/log-streaming-google-bigquery

Fastly's Real-Time Log Streaming feature can send log files to BigQuery, Google's managed enterprise data warehouse.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding BigQuery as a logging endpoint for Fastly services you will need to:

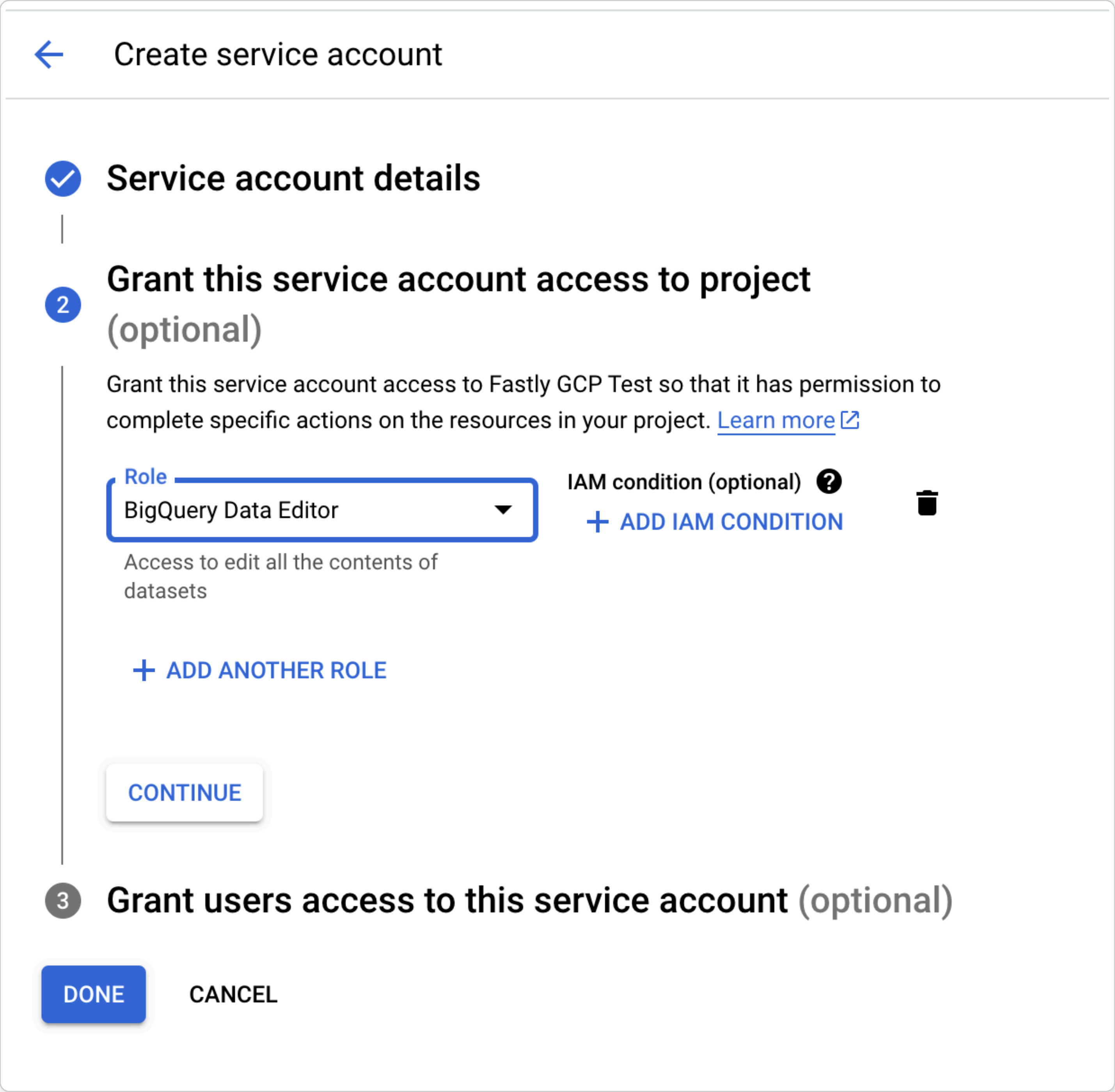
- Register for a Google Cloud Platform (GCP) account.
- Create a service account on Google's website.
- Use Google IAM service account impersonation or key-based access to configure the access method that Fastly uses to send logs.
- Enable the BigQuery API.
- Create a BigQuery dataset.
- Add a BigQuery table.

Creating a service account

Google Cloud Platform (GCP) uses service accounts for third-party application authentication and role-based access to Google Cloud resources. To create a new service account, follow the instructions in the Google Cloud documentation.

NOTE

When creating the service account for logging to BigQuery, the service account must be assigned the `Big Query Data Editor` role to write to the table you use for Fastly logging. Refer to Google's guide on BigQuery Roles for details about the default permissions assigned to the Big Query Data Editor role.



Create service account

Service account details

2 Grant this service account access to project (optional)

Grant this service account access to Fastly GCP Test so that it has permission to complete specific actions on the resources in your project. Learn more

Role: BigQuery Data Editor

Access to edit all the contents of datasets

3 Grant users access to this service account (optional)

Configure access for the service account

Fastly requires access to write logs to your BigQuery table. You can use either Google IAM service account impersonation or key-based access to allow Fastly to send logs to BigQuery.

Configuring Google IAM service account impersonation

If you elect to use Google IAM service account impersonation to avoid storing keys with Fastly, you may use the same service account created above. Our guide to creating an Google IAM role provides further details on configuring this feature.

Configuring key-based access

If you elect to use key-based access, refer to Google's guide on generating a service account credential.

When configuring key-based access, you must select the JSON format. Upon saving the configuration, a JSON file will be downloaded to your computer, containing the credentials for the GCS service account you just created. Open the file with a text editor and make a note of the `private_key` and `client_email`.

Enabling the BigQuery API

To send your Fastly logs to your BigQuery table, you'll need to enable the BigQuery API in the Google Cloud Platform API Manager.

Creating the BigQuery dataset

After you've enabled the BigQuery API, follow these instructions to create a BigQuery dataset:

1. Open the BigQuery page in the Cloud Console.
2. In the **Explorer** panel, select the project where you want to create the dataset.
3. In the details panel, click **Create dataset**.
4. In the **Dataset ID** field, enter a name for the dataset (e.g., `fastly_bigquery`).
5. Click **Create dataset**.

Adding a BigQuery table

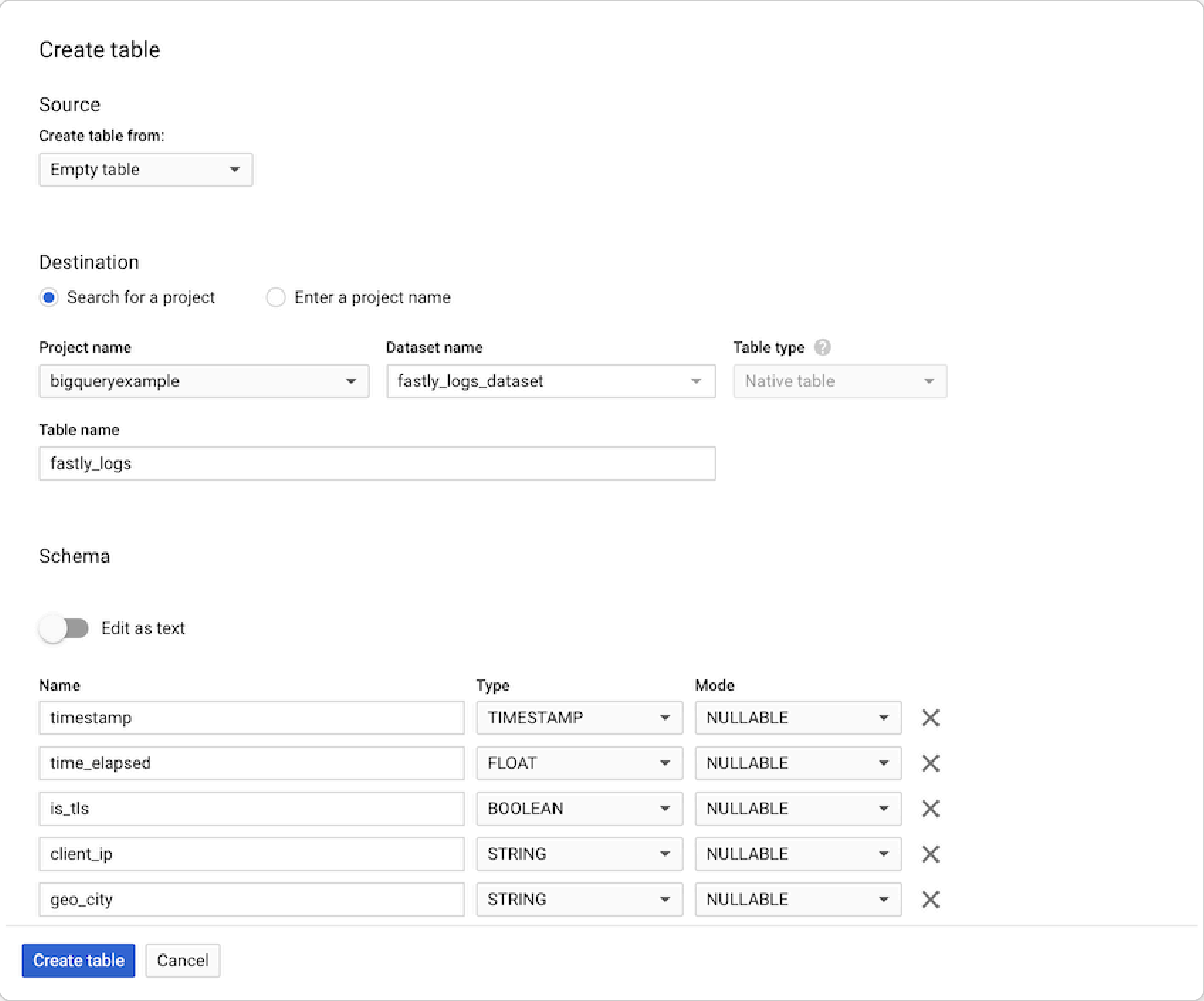
After you've created the BigQuery dataset, you'll need to add a BigQuery table. There are four ways of creating the schema for the table:

- Edit the schema using the BigQuery web interface.
- Edit the schema using the text field in the BigQuery web interface.
- Use an existing table.
- Set the table to automatically detect the schema.

> NOTE
>
> Setting the table to automatically detect the schema may give unpredictable results.

Follow these instructions to add a BigQuery table:

1. Open the BigQuery page in the Cloud Console.
2. In the Explorer panel, expand your project and select the BigQuery dataset you created previously.
3. In the **Source** section, select **Empty Table** from the **Create table from:** menu.



Create table

Source

Create table from: Empty table

Destination

● Search for a project ○ Enter a project name

Project name: bigqueryexample

Dataset name: fastly_logs_dataset

Table type: Native table

Table name: fastly_logs

Schema

Edit as text

Name	Type	Mode
timestamp	TIMESTAMP	NULLABLE
time_elapsed	FLOAT	NULLABLE
is_tls	BOOLEAN	NULLABLE
client_ip	STRING	NULLABLE
geo_city	STRING	NULLABLE

4. In the **Table name** field, enter a name for the table (e.g., `logs`).
5. In the **Schema** section of the BigQuery website, use the interface to add fields and complete the schema. Check out the example schema section for details.
6. Click **Create Table**.

Adding BigQuery as a logging endpoint

Follow these instructions to add BigQuery as a logging endpoint. As part of configuration, you can elect to configure Google IAM role-based service account impersonation to avoid storing secrets. Read our guide on creating a Google IAM role for more information on this feature.

1. Review the information in our guide to setting up remote log streaming.
2. In the Google BigQuery area, click **Create endpoint**.
3. Fill out the **Create a BigQuery endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log

placement for more information.

- In the **Log format** field, enter the data to send to BigQuery. Check out the example format section for details.
- In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Select either **User Credentials** or **IAM Role**.
- If you selected **User Credentials**, fill out the following fields:
  - In the **Email** field, enter the `client_email` address associated with the BigQuery service account.
  - In the **Secret key** field, enter the value of the `private_key` associated with your BigQuery service account.
  - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
  - In the **Dataset** field, enter the name of your BigQuery dataset.
  - In the **Table** field, enter the name of your BigQuery table.
  - *(Optional)* In the **Template** field, enter an `strftime` compatible string to use as the template suffix for your table.
- If you selected **IAM Role**, fill out the following fields:
  - In the **Service Account Name** field, enter the name of the service account email address you selected when configuring Google IAM service account impersonation.
  - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
  - In the **Dataset** field, enter the name of your BigQuery dataset.
  - In the **Table** field, enter the name of your BigQuery table.
  - *(Optional)* In the **Template** field, enter an `strftime` compatible string to use as the template suffix for your table.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

Data sent to BigQuery must be serialized as a JSON object, and every field in the JSON object must map to a string in your table's schema. The JSON can have nested data in it (e.g., the value of a key in your object can be another object). Here's an example format string for sending data to BigQuery:

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

Example schema

The BigQuery schema for the example format shown above would look something like this:

```
timestamp:TIMESTAMP,client_ip:STRING,geo_country:STRING,geo_city:STRING,host:STRING,u  ST
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the Google BigQuery area, click **Create endpoint**.

3. Fill out the **Create a BigQuery endpoint** fields as follows:

    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.

    - In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Select either **User Credentials** or **IAM Role**.

    - If you selected **User Credentials**, fill out the following fields:
        - In the **Email** field, enter the `client_email` address associated with the BigQuery service account.
        - In the **Secret key** field, enter the value of the `private_key` associated with your BigQuery service account.
        - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
        - In the **Dataset** field, enter the name of your BigQuery dataset.
        - In the **Table** field, enter the name of your BigQuery table.
        - *(Optional)* In the **Template** field, enter an `strftime` compatible string to use as the template suffix for your table.

    - If you selected **IAM Role**, fill out the following fields:
        - In the **Service Account Name** field, enter the name of the service account email address you selected when configuring Google IAM service account impersonation.
        - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
        - In the **Dataset** field, enter the name of your BigQuery dataset.
        - In the **Table** field, enter the name of your BigQuery table.

  - *(Optional)* In the **Template** field, enter an `strftime` compatible string to use as the template suffix for your table.

4. Click **Create** to create the new logging endpoint.

5. Click **Activate** to deploy your configuration changes.

Recommended log format

Data sent to BigQuery must be serialized as a JSON object, and every field in the JSON object must map to a field in your table's schema. The JSON can have nested data in it (e.g., the value for a key in your object can be another object). Here's an example format string for sending data to BigQuery:

```
{
  "client_ip": "127.0.0.1",
  "timestamp": "2022-05-17 15:09:24.037547 UTC",
  "geo_country": "USA",
  "geo_city": "boston",
  "host": "curiously-selected-polecat.edgecompute.app",
  "url": "https://curiously-selected-polecat.edgecompute.app/",
  "request_method": "GET",
  "request_protocol": "https",
  "request_referer": "",
  "request_user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/
  "response_status": "200",
  "response_reason": "OK",
  "response_body_size": "1234",
  "fastly_server": "IAD"
}
```

Example schema

The BigQuery schema for the example format shown above would look like this:

```
client_ip:STRING,timestamp:TIMESTAMP,geo_country:STRING,geo_city:STRING,host:STRING,u ST
```

* * *

Log streaming: Google Cloud Pub/Sub

/en/guides/log-streaming-google-cloud-pubsub

Fastly's Real-Time Log Streaming feature can send log files to Cloud Pub/Sub, Google's global messaging and event data ingestion product.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Cloud Pub/Sub as a logging endpoint for Fastly services, you will need to register for a Google Cloud Platform (GCP) account and then:

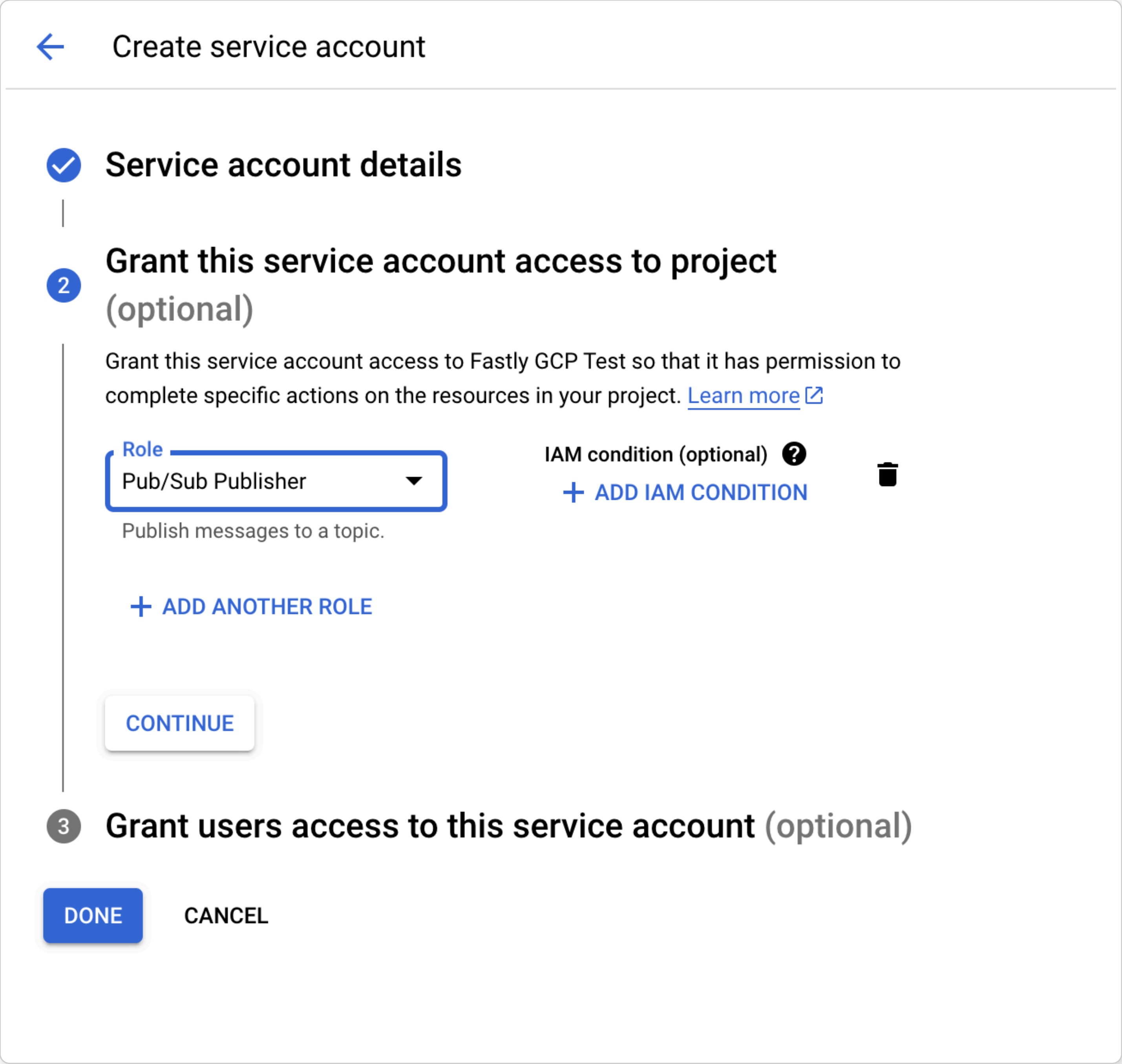
- Create a service account on Google's website.
- Navigate to the Pub/Sub section of the Google Cloud console. Follow the prompts to enable the API.
- Create a Pub/Sub topic.
- Use Google IAM service account impersonation or key-based access to configure the access method that Fastly uses to send logs.

Creating a service account

Google Cloud Platform (GCP) uses service accounts for third-party application authentication and role-based access to Google Cloud resources. To create a new service account, follow the instructions in the Google Cloud documentation.

NOTE

When creating the service account for logging to Google Pub/Sub, the service account must be assigned the `Pub/Sub Publisher` role to publish to the topic you created for Fastly logging.



Create service account

Service account details

Grant this service account access to project (optional)

Grant this service account access to Fastly GCP Test so that it has permission to complete specific actions on the resources in your project. Learn more

Role: Pub/Sub Publisher

Publish messages to a topic.

IAM condition (optional)
ADD IAM CONDITION
ADD ANOTHER ROLE
CONTINUE
Grant users access to this service account (optional)
DONE
CANCEL

Configure access for the service account

Fastly requires access to write logs to your Pub/Sub topic. You can use either role-based, service account impersonation, or key-based access to allow Fastly to send logs to Google Pub/Sub.

Configuring Google IAM service account impersonation

If you elect to use Google IAM service account impersonation to avoid storing keys with Fastly, you may use the same service account created above. Our guide to creating an Google IAM role provides further details on configuring this feature.

Configuring key-based access

If you elect to use key-based access, refer to Google's guide on generating a service account credential.

When configuring key-based access, you must select the JSON format. Upon saving the configuration, a JSON file will be downloaded to your computer, containing the credentials for the GCS service account you just created. Open the file with a text editor and make a note of the `private_key` and `client_email`.

NOTE

Read more about Cloud Pub/Sub in Google's documentation.

Adding Cloud Pub/Sub as a logging endpoint

Follow these instructions to add Cloud Pub/Sub as a logging endpoint. As part of configuration, you can elect to configure Google IAM role-based service account impersonation to avoid storing secrets. Read our guide on creating a Google IAM role for more information on this feature.

1. Review the information in our guide to setting up remote log streaming.
2. In the Google Cloud Pub/Sub area, click **Create endpoint**.
3. Fill out the **Create a Google Cloud Pub/Sub endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, enter the data to send to Google Cloud Pub/Sub. See the example format section for details.
    - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
    - In the **Email** field, enter the email address of the service account configured for your Pub/Sub topic.
    - In the **Topic** field, enter the Pub/Sub topic to which logs should be sent.
    - In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Valid values are **User Credentials** and **IAM Role**.
    - If you selected **User Credentials**, fill out the following fields:
        - In the **Email** field, enter the email address of the service account configured for your Pub/Sub topic.
        - In the **Secret Key** field, enter the exact value of the `private_key` associated with the service account configured for your Pub/Sub topic.
    - If you selected **IAM Role**, enter fill out following field:
        - In the **Service Account Name** field, enter the name of the service account email address you selected when configuring Google IAM service account impersonation.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

Data sent to Cloud Pub/Sub must be serialized as a JSON object, and every field in the JSON object must map to a string in your table's schema. The JSON can have nested data in it (e.g., the value of a key in your object can be another object). Here's an example format string for sending data:

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Google Cloud Pub/Sub area, click **Create endpoint**.
3. Fill out the **Create a Google Cloud Pub/Sub endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
   - In the **Project ID** field, enter the ID of your Google Cloud Platform project.
   - In the **Topic** field, enter the Pub/Sub topic to which logs should be sent.
   - In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Valid values are **User Credentials** and **IAM Role**.
   - If you selected **User Credentials**, fill out the following fields:
     - In the **Email** field, enter the email address of the service account configured for your Pub/Sub topic.
     - In the **Secret Key** field, enter the exact value of the private key associated with the service account configured for your Pub/Sub topic.
   - If you selected **IAM Role**, fill out the following field:
     - In the **Service Account Name** field, enter the name of the service account email address you selected when configuring Google IAM service account impersonation.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Google Cloud Storage

/en/guides/log-streaming-google-cloud-storage

Fastly's Real-Time Log Streaming feature can send log files to Google Cloud Storage (GCS). GCS is an online file storage service used for storing and accessing data on Google's infrastructure. One advantage of using GCS is that you can use Google BigQuery to analyze the log files.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding GCS as a logging endpoint for Fastly services, you will need to:

- Register for a GCS account.
- Create a bucket and service account on Google's website.
- Use Google IAM service account impersonation or key-based access to configure the access method that Fastly uses to send logs.
- Enable the Google Cloud Storage JSON API.

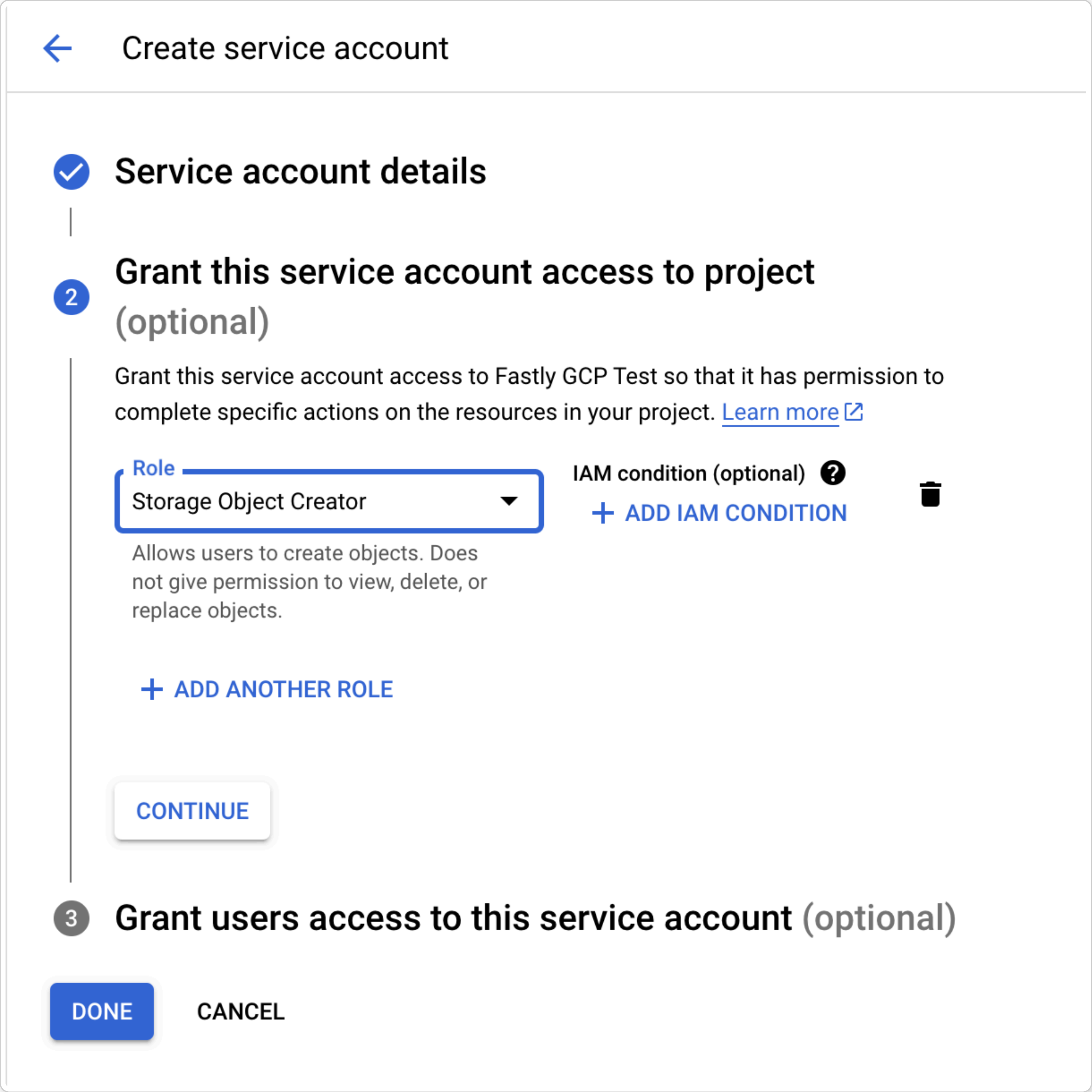
Creating a GCS bucket

You can create a new GCS bucket to hold the logs, or you can use an existing bucket. Be sure to note the name of the bucket as you will need it later. To learn how to create a GCS bucket, refer to Google's guide on creating a bucket.

Creating a service account

Google Cloud Platform (GCP) uses service accounts for third-party application authentication and role-based access to Google Cloud resources. To create a new service account, follow the instructions in the Google Cloud documentation.

> NOTE
>
> When creating the service account for logging to GCS, the service account must be assigned the `Storage Object Creator` role for the bucket you created above. Refer to the Storage Object Roles for details.



Create service account

Service account details

2 Grant this service account access to project (optional)

Grant this service account access to Fastly GCP Test so that it has permission to complete specific actions on the resources in your project. Learn more

Role: Storage Object Creator

Allows users to create objects. Does not give permission to view, delete, or replace objects.

IAM condition (optional)

3 Grant users access to this service account (optional)

Configure access for the service account

Fastly requires access to write logs to your GCS bucket. You can use either Google IAM service account impersonation or key-based access to allow Fastly to send logs to your GCS bucket

Configuring Google IAM service account impersonation

If you elect to use Google IAM service account impersonation to avoid storing keys with Fastly, you may use the same service account created above. Our guide to creating an Google IAM role provides further details on configuring this feature.

Configuring key-based access

If you elect to use key-based access, refer to Google's guide on generating a service account credential.

When configuring key-based access, you must select the JSON format. Upon saving the configuration, a JSON file will be downloaded to your computer, containing the credentials for the GCS service account you just created. Open the file with a text editor and make a note of the `private_key` and `client_email`.

Enabling the Google Cloud Storage JSON API

To ensure the Fastly logs are sent to your GCS bucket, you need to enable the Google Cloud Storage JSON API. For more information, refer to Google's instructions for activating the API.

Adding GCS as a logging endpoint

Follow these instructions to add GCS as a logging endpoint. As part of configuration, you can elect to configure Google IAM role-based service account impersonation to avoid storing secrets. Read our guide on creating a Google IAM role for more information on this feature.

1. Review the information in our guide to setting up remote log streaming.
2. In the Google Cloud Services area, click **Create endpoint**.
3. Fill out the **Create a Google Cloud Storage (GCS) endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Email** field, enter the `client_email` address listed in the JSON file associated with the service account you created on Google's website.
    - In the **Bucket name** field, enter the name of the GCS bucket in which to store the logs.
    - In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Valid values are **User Credentials** and **IAM Role**.
    - If you selected **User Credentials**, fill out the following fields:
        - In the **Email** field, enter the `client_email` address listed in the JSON file associated with the service account you created on Google's website.
        - In the **Secret key** field, enter the `private_key` value listed in the JSON file associated with the service account you created on Google's website. We strip out the JSON newline escape characters for you so don't worry about removing them.
    - If you selected **IAM Role**, fill out the following field:
        - In the **Service Account Name** field, enter the service account ID which was created for you from the service account name you entered when configuring Google IAM service account impersonation. The service account ID comes before the `@` in the service account email. For

example, if your service account email is `my-name@projectid.iam.gserviceaccount.com`, the service account ID is `my-name`.

    - In the **Project ID** field, enter the ID of your Google Cloud Platform project.

  - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.

  - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.

4. Click **Advanced options** and fill out the fields as follows:
  - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.

  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.

  - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to GCS. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the Google Cloud Services area, click **Create endpoint**.

3. Fill out the **Create a Google Cloud Storage (GCS) endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.

    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.

    - In the **Bucket name** field, enter the name of the GCS bucket in which to store the logs.

    - In the **Access Method** area, select how Fastly will access Google resources for purposes of log delivery. Valid values are **User Credentials** and **IAM Role**.

    - If you selected **User Credentials**, fill out the following fields:
        - In the **Email** field, enter the `client_email` address listed in the JSON file associated with the service account you created on Google's website.

        - In the **Secret key** field, enter the `private_key` value listed in the JSON file associated with the service account you created on Google's website. We strip out the JSON newline escape characters for you so don't worry about removing them.

    - If you selected **IAM Role**, fill out the following field:
        - In the **Service Account Name** field, enter the service account ID which was created for you from the service account name you entered when configuring Google IAM service account impersonation. The service account ID comes before the `@` in the service account email. For example, if your service account email is `my-name@projectid.iam.gserviceaccount.com`, the service account ID is `my-name`.

        - In the **Project ID** field, enter the ID of your Google Cloud Platform project.

    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.

    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.

4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.

    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.

    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Recommended log format

Log messages can take on any format you choose as long as they can be processed from GCS.

* * *

Log streaming: Honeycomb

/en/guides/log-streaming-honeycomb

Fastly's Real-Time Log Streaming feature can send logs in JSON format to Honeycomb. Honeycomb is a tool that allows developers to explore the operations of complex systems, microservices, and databases.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Honeycomb as a logging endpoint for Fastly services, you'll need to perform the following steps:

- Sign up for a Honeycomb account if you don't already have one.
- Obtain the Write Key for your team on the Honeycomb Account page.
- Choose a Dataset name. If you plan to collect data from multiple environments (like production, development, staging), Honeycomb recommends creating a Dataset for each environment and naming your Datasets accordingly (e.g., `prod.queries`, `dev.queries`, and `staging.queries`). If a Dataset doesn't exist, Honeycomb will create one automatically.

Adding Honeycomb as a logging endpoint

1. Review the information in our guide to setting up remote log streaming.
2. In the Honeycomb area, click **Create endpoint**.
3. Fill out the **Create a Honeycomb endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, enter the data to send to Honeycomb. See the example format section for details.
    - In the **Write Key** field, enter the write key for your Honeycomb team. This is available on the Honeycomb Account page.

   - In the **Dataset** field, enter the name of the Honeycomb Dataset (e.g., `myDataset`).

4. Click **Create** to create the new logging endpoint.

5. Click **Activate** to deploy your configuration changes.

Example format

Data sent to Honeycomb must be serialized as a JSON object. Here's an example format string for sending data to Honeycomb:

```
{
  "time":"%{begin:%Y-%m-%dT%H:%M:%SZ}t",
  "data":  {
    "service_id":"%{req.service_id}V",
    "time_elapsed":%D,
    "request":"%m",
    "host":"%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)
    "url":"%{cstr_escape(req.url)}V",
    "protocol":"%H",
    "is_ipv6":%{if(req.is_ipv6, "true", "false")}V,
    "is_tls":%{if(req.is_ssl, "true", "false")}V,
    "is_h2":%{if(fastly_info.is_h2, "true", "false")}V,
    "client_ip":"%h",
    "geo_city":"%{client.geo.city.utf8}V",
    "geo_country_code":"%{client.geo.country_code}V",
    "server_datacenter":"%{server.datacenter}V",
    "request_referer":"%{Referer}i",
    "request_user_agent":"%{User-Agent}i",
    "request_accept_content":"%{Accept}i",
    "request_accept_language":"%{Accept-Language}i",
    "request_accept_charset":"%{Accept-Charset}i",
    "cache_status":"%{regsub(fastly_info.state, "^(HIT-(SYNTH)|(HITPASS|HIT|MISS|PASS
    "status":"%s",
    "content_type":"%{Content-Type}o",
    "req_header_size":%{req.header_bytes_read}V,
    "req_body_size":%{req.body_bytes_read}V,
    "resp_header_size":%{resp.header_bytes_written}V,
    "resp_body_size":%{resp.body_bytes_written}V
  }
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the Honeycomb area, click **Create endpoint**.

3. Fill out the **Create a Honeycomb endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.

  - In the **Write Key** field, enter the write key for your Honeycomb team. This is available on the Honeycomb Account page.
  - In the **Dataset** field, enter the name of the Honeycomb Dataset (e.g., `myDataset`).

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Recommended log format

Data sent to Honeycomb must be serialized in a way conforming to Honeycomb's expectations. If your logs are not formatted properly, attempts at processing your logs by your Honeycomb endpoint may fail. Here's an example format string for sending data to Honeycomb:

```
{
    "time": "2022-05-11T23:35:30.384Z",
    "data": {
        "message": "Something happened",
        "severity": "INFO"
    }
}
```

You can follow the general JSON structure above regardless of the chosen language for your Compute service and include the specific details inside the nested `data` structure. Consult the Honeycomb documentation for other available options for Honeycomb log messages. The emitted logs must be formatted as valid JSON.

* * *

Log streaming: HTTPS

/en/guides/log-streaming-https

Fastly's Real-Time Log Streaming feature can send log files to an HTTPS endpoint.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

When sending logs to a HTTPS endpoint, Fastly requires proof that you control the domain name specified in the **URL** field by using a HTTP challenge on a well-known path. If, for example, your **URL** field is `foo.example.com/some/log/path`, then the following challenge path must send a 200 response:

```
foo.example.com/.well-known/fastly/logging/challenge
```

Responses must include the hex representation of the SHA-256 of your Fastly service ID and it must appear on its own line in the response. For example:

```
$ sha256sum <SERVICEID>

ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c
```

If multiple service IDs are used, multiple hex(sha256) lines can be added to that challenge body. In addition, an asterisk (`*`) can be used on a line to allow any service to post to the HTTP endpoint. For example:

```
ef537f25c895bfa782526529a9b63d97aa631564d5d789c2b765448c8635fb6c
06ae6402e02a9dad74edc71aa69c77c5747e553b0840bfc56feb7e65b23f0f61
*
```

Adding HTTPS as a logging endpoint

Follow these instructions to add HTTPS as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the HTTPS area, click **Create endpoint**.
3. Fill out the **Create an HTTPS endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **URL** field, enter the URL to which log data will be sent (e.g., `https://logs.example.com/`).
    - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to send as a batch.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum size of a log batch.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Content type** field, enter the content type to use when sending logs (e.g., `application/json`).
    - *(Optional)* In the **Custom header name** field, enter a custom header to use when sending logs (e.g., `Authorization`).
    - *(Optional)* In the **Custom header value** field, enter a custom header value to use when sending logs (e.g., `Bearer <token>`).
    - *(Optional)* In the **Method** area, select the appropriate HTTP method to use.
    - In the **JSON log entry format** area, select the appropriate log entry format to use. The JSON log entry format enforces valid JSON formatting. Selecting **Array of JSON** wraps JSON log batches in an array. Selecting **Newline delimited** places each JSON log entry onto a new line in a batch.

  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.

5. Fill out the **Using your own certificate authority (CA)** section of the **Advanced options** area as follows:
   - *(Optional)* In the **TLS Hostname** field, enter the hostname used to verify the logging endpoint server's certificate. This can be either the Common Name (CN) or Subject Alternative Name (SAN). This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify the logging endpoint server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate Fastly to the logging endpoint server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your logging endpoint server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the logging endpoint server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your logging endpoint server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
6. Click **Create** to create the new logging endpoint.
7. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to an HTTPS logging endpoint. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the HTTPS area, click **Create endpoint**.
3. Fill out the **Create an HTTPS endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
   - In the **URL** field, enter the URL to which log data will be sent (e.g., `https://logs.example.com/`).
   - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to send as a batch.
   - *(Optional)* In the **Maximum bytes** field, enter the maximum size of a log batch.
4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Content type** field, enter the content type to use when sending logs (e.g., `application/json`).
   - *(Optional)* In the **Custom header name** field, enter a custom header to use when sending logs (e.g., `Authorization`).
   - *(Optional)* In the **Custom header value** field, enter a custom header value to use when sending logs (e.g., `Bearer <token>`).
   - *(Optional)* In the **Method** area, select the appropriate HTTP method to use.
   - In the **JSON log entry format** area, select the appropriate log entry format to use. The JSON log entry format enforces valid JSON formatting. Selecting **Array of JSON** wraps JSON log batches in an array. Selecting **Newline delimited** places each JSON log entry onto a new line in a batch.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
5. Fill out the **Using your own certificate authority (CA)** section of the **Advanced options** area as follows:
   - *(Optional)* In the **TLS Hostname** field, enter the hostname used to verify the logging endpoint server's certificate. This can be either the Common Name (CN) or Subject Alternative Name (SAN). This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify the logging endpoint server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate Fastly to the logging endpoint server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your logging endpoint server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the logging endpoint server. The TLS client key you upload must be in PEM format and must be

accompanied by a TLS client certificate. A TLS client key allows your logging endpoint server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

6. Click **Create** to create the new logging endpoint.
7. Click **Activate** to deploy your configuration changes.

Firewall considerations

Your HTTPS endpoint may have limited security features. For this reason, it's best to create a firewall for your HTTP endpoint server and only accept TCP traffic on your configured port from our address blocks. Our list of IP address blocks is dynamic, so we recommend programmatically obtaining the list whenever possible.

* * *

Log streaming: Hydrolix

/en/guides/log-streaming-hydrolix

Fastly's Real-Time Log Streaming feature can send log files to Hydrolix, a cloud-based time-series data platform. Hydrolix provides a native integration for Fastly log storage and analysis through Fastly's HTTPS logging endpoint. Hydrolix lets you ingest and query those logs in real-time.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

If you don't already have a Hydrolix account, you'll need to sign up on the Hydrolix website. You'll also need to know the following about the target Hydrolix environment:

- Instance name
- Project name
- Table name

TIP

Consider reading Hydrolix's documentation on integrating with Fastly.

Adding Hydrolix as a logging endpoint

NOTE

This logging endpoint is only available for Fastly's Deliver (VCL-based) services, not for Compute services.

Follow these instructions to add Hydrolix as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the HTTPS area, click **Create endpoint** button.
3. Fill out the **Create an HTTPS endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, replace the placeholder log format and make the appropriate changes as shown in our log format and recommendations section below.
    - In the **URL** field, enter `https://<hydrolix-instance>.hydrolix.live/ingest/event`, replacing `<hydrolix-instance>` with the name of your Hydrolix instance.
    - In the **Maximum logs** field, leave as `0` (the default).
    - In the **Maximum bytes** field, enter `0`.
4. Click the **Advanced options** link of the **Create an HTTPS endpoint** page. The Advanced options appear.
5. Fill out the **Advanced options** of the **Create an HTTPS endpoint** page as follows:
    - In the **Content type** field, enter `application/json`.
    - In the **Custom header name** field, enter `x-hdx-table`.
    - In the **Custom header value** field, enter `<hydrolix-project-name>.<hydrolix-table-name>`, substituting the values for your Hydrolix project and table names.
    - From the **Method** controls, select **POST**.
    - From the **JSON log entry format** controls, select **Newline delimited**.
    - Leave the **Select a log line format** and **Placement** controls set to the defaults.
    - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported.
    - Leave **TLS CA certificate** field, **TLS client certificate** field and **TLS client key** field all empty.
6. Click **Create** to create the new logging endpoint.
7. Click **Activate** to deploy your configuration changes.

Log format recommendations

For this example we use the log format shown below. You can customize this format with any values you want as long as you also modify your **Transform** and **View** configurations.

```
{
  "timestamp":"%{begin:%Y-%m-%dT%H:%M:%S}t",
  "time_elapsed":%{time.elapsed.usec}V,
  "is_tls":%{if(req.is_ssl, "true", "false")}V,
  "client_ip":"%{req.http.Fastly-Client-IP}V",
  "geo_city":"%{client.geo.city}V",
  "geo_country_code":"%{client.geo.country_code}V",
  "request":"%{req.request}V",
  "host":"%{req.http.Fastly-Orig-Host}V",
  "url":"%{json.escape(req.url)}V",
  "request_referer":"%{json.escape(req.http.Referer)}V",
  "request_user_agent":"%{json.escape(req.http.User-Agent)}V",
  "request_accept_language":"%{json.escape(req.http.Accept-Language)}V",
  "request_accept_charset":"%{json.escape(req.http.Accept-Charset)}V",
  "cache_status":"%{regsub(fastly_info.state, "^(HIT-(SYNTH)|(HITPASS|HIT|MISS|PASS|E
}
```

Configuring Hydrolix Streaming Intake

Once you have your `project` and `table` setup and Fastly is configured to send logs to your Hydrolix instance, you can focus on setting up the Fastly Log streaming ingest pipeline by defining an ingest transform schema.

Creating a transform schema

Below is the suggested transform schema to use with the recommended log format described above. Be sure to replace `<table uuid>` with the UUID of your target Hydrolix table. You need to have this transform setup as the default, so `"is_default"` is set to `true`.

```
{
    "name": "fastly_transform",
    "type": "json",
    "table": "<table uuid>",
    "description": "fastly https logs",
    "settings": {
        "is_default": true,
        "output_columns": [
                            {
                                "name": "timestamp",
                                "datatype": {
                                    "type": "datetime",
                                    "primary": true,
                                    "format": "2006-01-02T15:04:05"
                                }
                            },
                            {
                                "name": "time_elapsed",
                                "datatype": {
                                    "type": "uint64"
                                }
                            },
                            {
                                "name": "is_tls",
                                "datatype": {
                                    "type": "bool"
                                    }
                                },
                                {
                                    "name": "client_ip",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "geo_city",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "geo_country_code",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "request",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "host",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "url",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "request_referer",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
                                    "name": "request_user_agent",
                                    "datatype": {
                                        "type": "string"
                                    }
                                },
                                {
```

```
                                "name": "request_accept_language",
                                "datatype": {
                                    "type": "string"
                                }
                            },
                            {
                                "name": "request_accept_charset",
                                "datatype": {
                                    "type": "string"
                                }
                            },
                            {
                                "name": "cache_status",
                                "datatype": {
                                    "type": "string"
                                }
                            }
                        ]
        }
    }
```

Once you define the transform schema, Hydrolix is configured to accept the incoming Fastly log data.

Leveraging views

Hydrolix supports having many different query formats for a single dataset. The query data structure, or view, for a given dataset allows you to customized the queried data and restrict a user’s access to a specific set of columns.

Hydrolix automatically generates a default view upon transform creation that can be used to immediately query the dataset - no additional configuration is required. However, you are encouraged to familiarize yourself with the view concept and the benefits that the feature can provide. More detailed information can be found on the Hydrolix site.

Further reading

Hydrolix provides a tutorial for querying and analyzing Fastly data from within their system.

* * *

Log streaming: Kafka

/en/guides/log-streaming-kafka

Fastly's Real-Time Log Streaming feature can send logs to Apache Kafka. Kafka is an open-source, high-throughput, low-latency platform for handling real-time data feeds.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Before you begin

Before adding Apache Kafka as a logging endpoint for Fastly services, ensure Kafka is running on a remote server. You'll need to know the hostname or IP address of one or more servers (brokers) and the category or feed name to which messages will be stored (topic). For more information on setting up Kafka check out the Apache Kafka Quickstart guide.

Each emitted Fastly log constitutes an individual Kafka record, and Fastly follows standard Kafka client protocols of producing multiple records for multiple Kafka partitions in batches. To ensure adequate log message throughput, be sure to adjust the **Maximum bytes** and **Compression codec** settings to match the capabilities of your log processing infrastructure.

Kafka is a shared resource by default, and other users of Kafka within your environment may affect throughput for Fastly logs. If you are operating your own Kafka infrastructure, Fastly recommends designating exclusive disk storage volumes to the partitions of the Kafka topic where you will send logs. You should also monitor IOPS and consumed bandwidth against the storage maximums to anticipate the need to expand capacity.

Adding Kafka as a logging endpoint

Follow these instructions to add Kafka as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Apache Kafka area, click **Create endpoint**.
3. Fill out the **Create an Apache Kafka endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **Brokers** field, enter the hostname or IP address of one or more servers (Kafka brokers). By default, it will use port 9092. Be sure to append the port using the format `:[port]` (e.g., `:9093`) to the hostname if it is different from the default port. You can specify multiple servers using a comma-separated string.
    - In the **Topic** field, enter the name of the topic to send logs to.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum size of a Kafka produce request message in bytes.
    - *(Optional)* From the **Parse key-values** controls, select whether or not to parse any key-value pairs within the log format into Kafka record headers. Key-value pairs must lead a formatted log line as `key=val` and must be comma-separated. Neither keys nor values can begin with a double-quote. Keys can be preceded by an arbitrary number of spaces, cannot contain spaces, and must have at least one character. Values can be empty as in `key=`. Specify a record key for partitioning by including a

`__record_key` key-value pair in your log line. Logs will be partitioned according to Kafka's uniform sticky partitioner strategy. In the absence of a designated record key, Fastly's Kafka client will distribute log messages evenly across the available partitions of a topic.

  - *(Optional)* In the **Write acknowledgement** area, select the appropriate write acknowledgement a leader Kafka broker must receive for a produce request to be successful. Fastly's Kafka client will attempt limited redelivery of failed produce requests with exponential backoff and jitter to reduce thundering herd scenarios.
  - *(Optional)* In the **Compression codec** area, select the appropriate codec to use for compression of your logs.
  - *(Optional)* From the **Use SASL** controls, select whether or not to enable SASL authentication. SASL authentication can be enabled concurrently with TLS encryption. When you select Yes, additional SASL authentication fields appear.
  - From the **SASL authentication mechanism** menu, select the appropriate challenge-response mechanism to use for authenticating the SASL client authentication username and password.
  - In the **User** field, enter the SASL client authentication username.
  - In the **Password** field, enter the SASL client authentication password.
  - *(Optional)* From the **Use TLS** controls, select whether or not to enable TLS encryption for the Kafka endpoint. TLS encryption can be enabled concurrently with SASL authentication. When you select Yes, additional TLS fields appear.
  - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. If the hostname is not specified, the hostname of the first broker in the Brokers field will be used. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the Kafka broker's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the Kafka broker. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your Kafka broker to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the Kafka broker. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your Kafka broker to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Apache Kafka. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the Apache Kafka area, click **Create endpoint**.

3. Fill out the **Create an Apache Kafka endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Brokers** field, enter the hostname or IP address of one or more servers (Kafka brokers). By default, it will use port 9092. Be sure to append the port using the format `:[port]` (e.g., `:9093`) to the hostname if it is different from the default port. You can specify multiple servers using a comma-separated string.
    - In the **Topic** field, enter the name of the topic to send logs to.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum size of a Kafka produce request message in bytes.
    - *(Optional)* From the **Parse key-values** controls, select whether or not to parse any key-value pairs within the log format into Kafka record headers. Key-value pairs must lead a formatted log line as `key=val` and must be comma-separated. Neither keys nor values can begin with a double-quote. Keys can be preceded by an arbitrary number of spaces, cannot contain spaces, and must have at least one character. Values can be empty as in `key=`. Specify a record key for partitioning by including a `__record_key` key-value pair in your log line. Logs will be partitioned according to Kafka's uniform sticky partitioner strategy. In the absence of a designated record key, Fastly's Kafka client will distribute log messages evenly across the available partitions of a topic.
    - *(Optional)* In the **Write acknowledgement** area, select the appropriate write acknowledgement a leader Kafka broker must receive for a produce request to be successful. Fastly's Kafka client will attempt

limited redelivery of failed produce requests with exponential backoff and jitter to reduce thundering herd scenarios.

  - *(Optional)* In the **Compression codec** area, select the appropriate codec to use for compression of your logs.
  - *(Optional)* From the **Use SASL** controls, select whether or not to enable SASL authentication. SASL authentication can be enabled concurrently with TLS encryption. When you select Yes, additional SASL authentication fields appear.
  - From the **SASL authentication mechanism** menu, select the appropriate challenge-response mechanism to use for authenticating the SASL client authentication username and password.
  - In the **User** field, enter the SASL client authentication username.
  - In the **Password** field, enter the SASL client authentication password.
  - *(Optional)* From the **Use TLS** controls, select whether or not to enable TLS encryption for the Kafka endpoint. TLS encryption can be enabled concurrently with SASL authentication. When you select Yes, additional TLS fields appear.
  - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. If the hostname is not specified, the hostname of the first broker in the Brokers field will be used. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the Kafka broker's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the Kafka broker. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your Kafka broker to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
  - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the Kafka broker. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your Kafka broker to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

---

* * *

Log streaming: Log Shuttle

/en/guides/log-streaming-log-shuttle

Fastly's Real-Time Log Streaming feature can send log files to Log Shuttle. Log Shuttle is an open source application designed to provide simpler encrypted and authenticated log delivery.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Adding Log Shuttle as a logging endpoint

Follow these instructions to add Log Shuttle as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Log Shuttle area, click **Create endpoint**.
3. Fill out the **Create a Log Shuttle endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **Token** field, enter the data authentication token. This is required for some endpoints like Heroku's Log Integration.
    - In the **URL** field, enter the URL to which log data will be sent (e.g., `https://logs.example.com/`).
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Log Shuttle. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Log Shuttle area, click **Create endpoint**.
3. Fill out the **Create a Log Shuttle endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Token** field, enter the data authentication token. This is required for some endpoints like Heroku's Log Integration.
    - In the **URL** field, enter the URL to which log data will be sent (e.g., `https://logs.example.com/`).
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: LogDNA

/en/guides/log-streaming-logdna

Fastly's Real-Time Log Streaming feature can be configured to send logs in a format that is readable by LogDNA. LogDNA is a cloud-based log management system that aggregates system and application logs into a single location.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding LogDNA as a logging endpoint for Fastly services, you'll need to perform the following steps:

- Sign up for a LogDNA account if you don't already have one. You can sign up for a free (but restricted plan) or upgrade a LogDNA plan to include more features.
- Set up a new LogDNA syslog source via the LogDNA web application by following their account-tailored log source instructions. Be sure to make note of the port number displayed at the end of the syslog URL when you complete set up. This is the port number you'll enter when setting up LogDNA as a logging endpoint for Fastly.

Adding LogDNA as a logging endpoint

1. Review the information in our guide to setting up remote log streaming.
2. In the LogDNA (via Syslog) area, click **Create endpoint**.
3. Fill out the **Create a Syslog endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
   - In the **Syslog address** field, enter `syslog-a.logdna.com`. In the port field after the colon, enter the LogDNA port number you noted during your LogDNA account setup.
   - From the **TLS** menu, select **Yes** to enable encryption for the syslog endpoint. The TLS Hostname and TLS CA Certificate fields will both appear.
   - In the **TLS Hostname** field, enter `syslog-a.logdna.com`. This is the hostname Fastly will use to verify the syslog server's certificate.
4. Click the **Advanced options** link of the **Create a Syslog endpoint** page and decide which of the optional fields to change, if any.
5. Fill out the **Advanced options** of the **Create a Syslog endpoint** page as follows:
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
6. Click **Create** to create the new logging endpoint.
7. Click **Activate** to deploy your configuration changes.

Logs should begin appearing in your LogDNA account a few seconds after you've created the endpoint and deployed your service.

Example format

The following is an example format string for sending data to LogDNA. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the LogDNA (via Syslog) area, click **Create endpoint**.
3. Fill out the **Create a Syslog endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Syslog address** field, enter `syslog-a.logdna.com`. In the port field after the colon, enter the LogDNA port number you noted during your LogDNA account setup.
    - From the **TLS** menu, select **Yes** to enable encryption for the Syslog endpoint. The TLS Hostname and TLS CA Certificate fields will both appear.
    - In the **TLS Hostname** field, enter `syslog-a.logdna.com`. This is the hostname Fastly will use to verify the Syslog server's certificate.
4. *(Optional)* To change the log line format, click **Advanced options**. In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information. This must be compatible with your LogDNA configuration.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Logs should begin appearing in your LogDNA account a few seconds after you've created the endpoint and deployed your service.

Recommended log format

Log messages can take on any format you choose because the log line format selected above will affect the overall line sent to and parsed by LogDNA.

* * *

Log streaming: Loggly

/en/guides/log-streaming-loggly

Fastly's Real-Time Log Streaming feature can send log files to Loggly. Loggly is an agent-less log collection and management tool.

NOTE

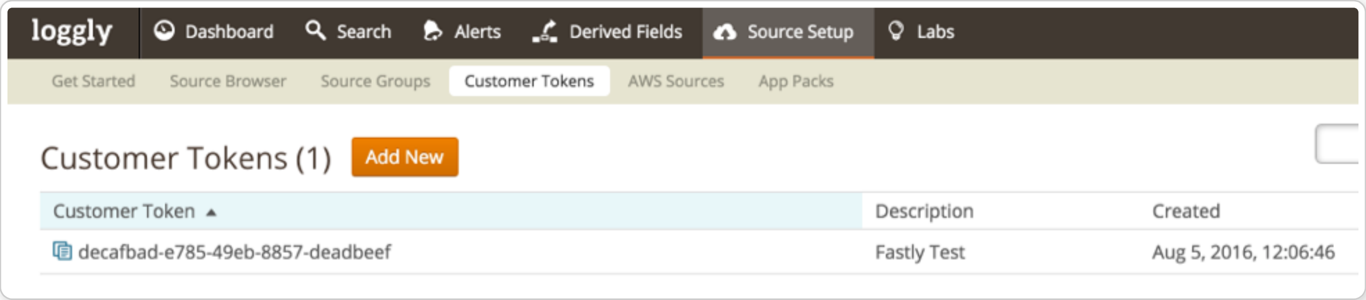
Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

If you don't already have a Loggly account, you'll need to register for one. Follow the signup instructions on the Loggly website.

Follow the steps below to find your Loggly customer token:

1. Navigate to the **Customer Tokens** area in the **Source Setup** on your Loggly dashboard.



Customer Token ▲	Description	Created
decafbad-e785-49eb-8857-deadbeef	Fastly Test	Aug 5, 2016, 12:06:46

2. Make note of your Loggly customer token. Loggly uses this to associate data you send them with your account.

Adding Loggly as a logging endpoint

After you've created a Loggly account and obtained your customer token, follow these instructions to add Loggly as a logging endpoint for Fastly services:

1. Review the information in our guide to setting up remote log streaming.
2. In the Loggly area, click **Create endpoint**.
3. Fill out the **Create a Loggly endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **Token** field, enter your Loggly customer token.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Loggly. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Loggly area, click **Create endpoint** button.
3. Fill out the **Create a Loggly endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
   - In the **Token** field, enter your Loggly customer token.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Heroku's Logplex

/en/guides/log-streaming-logplex

As part of our Real-Time Log Streaming feature, if you use our Heroku add-on, you can send log files directly to Heroku's Logplex system. Logplex is Heroku's distributed syslog router that collates and distributes log entries from a variety of sources into a single channel.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before continuing, you will need the token from your Heroku Logplex account. If you don't have a Heroku Logplex account, now is the time to set one up by signing up for Heroku.

Once enabled, your Fastly logs will be available in exactly the same way as your regular app and hosted service logs. You can view them using the Heroku command line log viewer or send them to a logging add-on.

Adding Heroku Logplex as a logging endpoint

Follow these instructions to add Heroku Logplex as a logging endpoint for Fastly services:

1. Review the information in our guide to setting up remote log streaming.
2. In the Heroku Logplex area, click **Create endpoint**.
3. Fill out the **Create a Heroku Logplex endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **Token** field, enter your Heroku Logplex token.
    - In the **URL** field, enter `https://1.us.logplex.io/logs` unless otherwise instructed by our support staff.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Logplex. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Heroku Logplex area, click **Create endpoint**.
3. Fill out the **Create a Heroku Logplex endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Token** field, enter your Heroku Logplex token.
    - In the **URL** field, enter `https://1.us.logplex.io/logs` unless otherwise instructed by our support staff.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: New Relic Logs

/en/guides/log-streaming-newrelic-logs

Fastly's Real-Time Log Streaming feature can send log files to New Relic Logs.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding New Relic Logs as a logging endpoint for Fastly services, you will need to:

- Register for a New Relic account.
- Obtain your license key.

Adding New Relic Logs as a logging endpoint

Follow these instructions to add New Relic Logs as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the New Relic Logs area, click **Create endpoint**.
3. Fill out the **Create a New Relic Logs endpoint** fields as follows:

  - In the **Name** field, enter a human-readable name for the endpoint.
  - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
  - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
  - In the **License key / Insert key** field, enter your New Relic license key or Insert API key.
  - From the **Region** controls, select the region to stream logs to.

4. Click **Create** to create the new logging endpoint.

5. Click **Activate** to deploy your configuration changes.

Example format

Data sent to New Relic Logs must be serialized as a JSON object. Here's the recommended example format string for sending data to New Relic Logs:

```
{
  "timestamp": %{time.start.msec}V,
  "cache_status":"%{regsub(fastly_info.state, "^(HIT-(SYNTH)|(HITPASS|HIT|MISS|PASS|E
  "client_ip":"%h",
  "client_device_type":"%{client.platform.hwtype}V",
  "client_os_name":"%{client.os.name}V",
  "client_os_version":"%{client.os.version}V",
  "client_browser_name":"%{client.browser.name}V",
  "client_browser_version":"%{client.browser.version}V",
  "client_as_name":"%{client.as.name}V",
  "client_as_number":"%{client.as.number}V",
  "client_connection_speed": "%{client.geo.conn_speed}V",
  "client_port": %{client.port}V,
  "client_rate_bps":%{client.socket.tcpi_delivery_rate}V,
  "client_recv_bytes":%{client.socket.tcpi_bytes_received}V,
  "client_requests_count":%{client.requests}V,
  "client_resp_body_size_write": %{resp.body_bytes_written}V,
  "client_resp_header_size_write": %{resp.header_bytes_written}V,
  "client_resp_ttfb": %{time.to_first_byte}V,
  "client_rtt_us":%{client.socket.tcpi_rtt}V,
  "content_type":"%{Content-Type}o",
  "domain": "%{Fastly-Orig-Host}i",
  "fastly_datacenter": "%{server.datacenter}V",
  "fastly_host": "%{server.hostname}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V,
  "fastly_region": "%{server.region}V",
  "fastly_server": "%{json.escape(server.identity)}V",
  "host": "%v",
  "origin_host":"%v",
  "origin_name":"%{req.backend.name}V",
  "request":"%{req.request}V",
  "request_method":"%m",
  "request_accept_charset":"%{json.escape(req.http.Accept-Charset)}V",
  "request_accept_language":"%{json.escape(req.http.Accept-Language)}V",
  "request_referer":"%{json.escape(req.http.Referer)}V",
  "request_user_agent":"%{json.escape(req.http.User-Agent)}V",
  "resp_status":"%s",
  "response": "%{resp.response}V",
  "service_id":"%{req.service_id}V",
  "service_version": "%{req.vcl.version}V",
  "status":"%s",
  "time_start":"%{begin:%Y-%m-%dT%H:%M:%SZ}t",
  "time_end":"%{end:%Y-%m-%dT%H:%M:%SZ}t",
  "time_elapsed":%D,
  "tls_cipher":"%{json.escape(tls.client.cipher)}V",
  "tls_version":"%{json.escape(tls.client.protocol)}V",
  "url":"%{json.escape(req.url)}V",
  "user_agent":"%{json.escape(req.http.User-Agent)}V",
  "user_city":"%{client.geo.city.utf8}V",
  "user_country_code":"%{client.geo.country_code}V",
  "user_continent_code":"%{client.geo.continent_code}V",
  "user_region":"%{client.geo.region}V"
}
```

Logging a timestamp

If a timestamp field is present in the Fastly log, it must be specified as milliseconds since Epoch to override the New Relic timestamp. If not included, Fastly will generate a timestamp.

Using New Relic Instant Observability's prebuilt Fastly dashboard

New Relic I/O is an open source ecosystem of community-contributed quickstarts for hundreds of tools and technologies. We've worked with New Relic to develop a prebuilt dashboard that highlights certain key metrics we think are important and useful. Setting up the dashboard is easy and the code is open source in case you want to customize it.

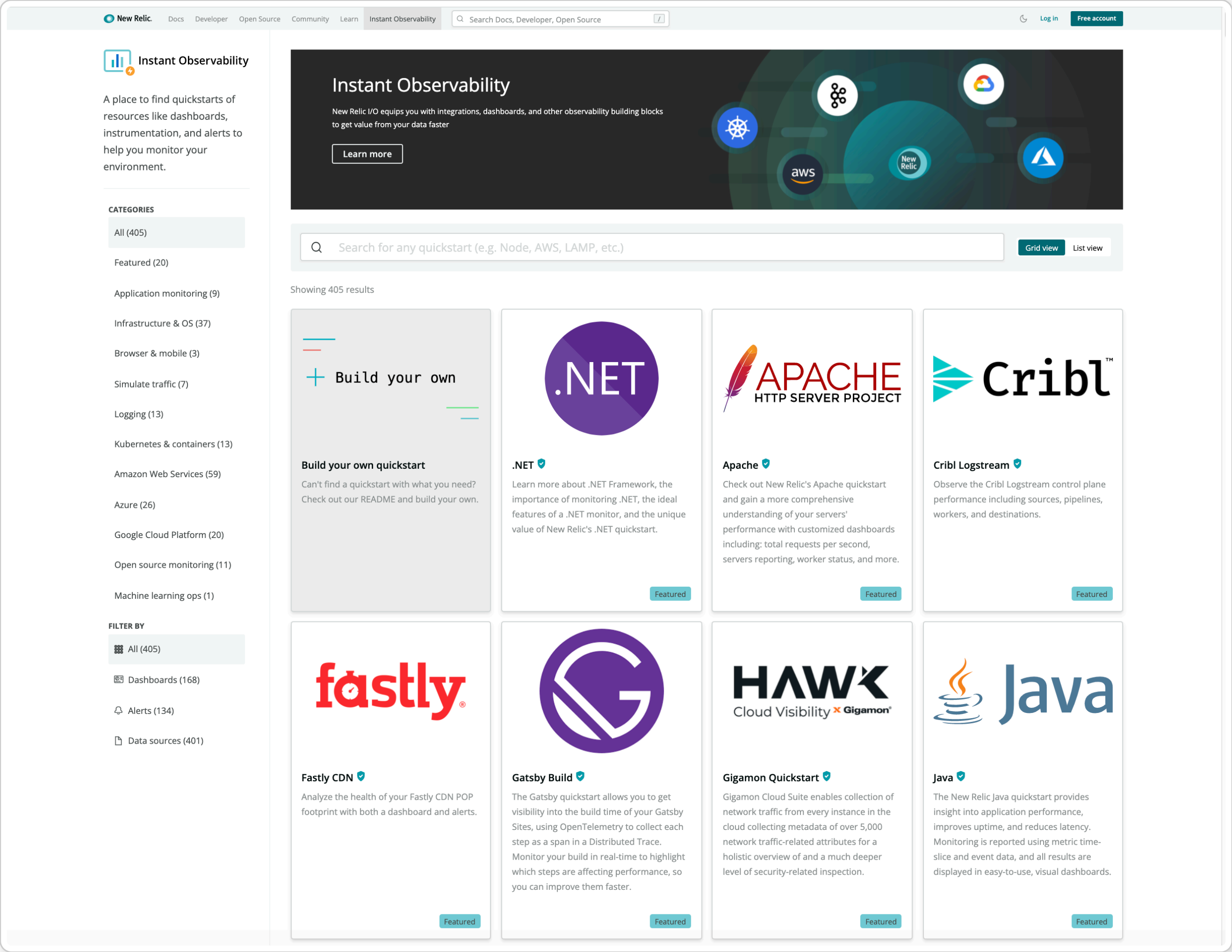
Configure your service to use the recommended expanded logging format

Since the prebuilt dashboard expects certain fields to be present, we encourage using the recommended logging format. That said, there is nothing to stop you from adding fields for your own purposes or to maintain backward compatibility with existing dashboards you've built. The dashboard won't break if you don't send some fields, but certain charts won't have data.

Install the Fastly dashboard

Follow these instructions to install the Fastly dashboard quickstart:

1. Select the Fastly dashboard quickstart from the New Relic marketplace.


New Relic.
Docs
Developer
Open Source
Community
Learn
Instant Observability
Search Docs, Developer, Open Source
Log in
Free account
Instant Observability
CATEGORIES
All (405)
Featured (20)
Application monitoring (9)
Infrastructure & OS (37)
Browser & mobile (3)
Simulate traffic (7)
Logging (13)
Kubernetes & containers (13)
Amazon Web Services (59)
Azure (26)
Google Cloud Platform (20)
Open source monitoring (11)
Machine learning ops (1)
FILTER BY
All (405)
Dashboards (168)
Alerts (134)
Data sources (401)
Learn more
aws
New Relic
Search for any quickstart (e.g. Node, AWS, LAMP, etc.)
Grid view
List view
Build your own
.NET
APACHE
HTTP SERVER PROJECT
Cribl
fastly
HAWK
Cloud Visibility × Gigamon
Java
Featured
Featured
Featured
Featured
Featured
Featured
Featured

A place to find quickstarts of resources like dashboards, instrumentation, and alerts to help you monitor your environment.

Instant Observability

New Relic I/O equips you with integrations, dashboards, and other observability building blocks to get value from your data faster

Showing 405 results

Build your own quickstart
Can't find a quickstart with what you need? Check out our README and build your own.

.NET
Learn more about .NET Framework, the importance of monitoring .NET, the ideal features of a .NET monitor, and the unique value of New Relic's .NET quickstart.

Apache
Check out New Relic's Apache quickstart and gain a more comprehensive understanding of your servers' performance with customized dashboards including: total requests per second, servers reporting, worker status, and more.

Cribl Logstream
Observe the Cribl Logstream control plane performance including sources, pipelines, workers, and destinations.

Fastly CDN
Analyze the health of your Fastly CDN POP footprint with both a dashboard and alerts.

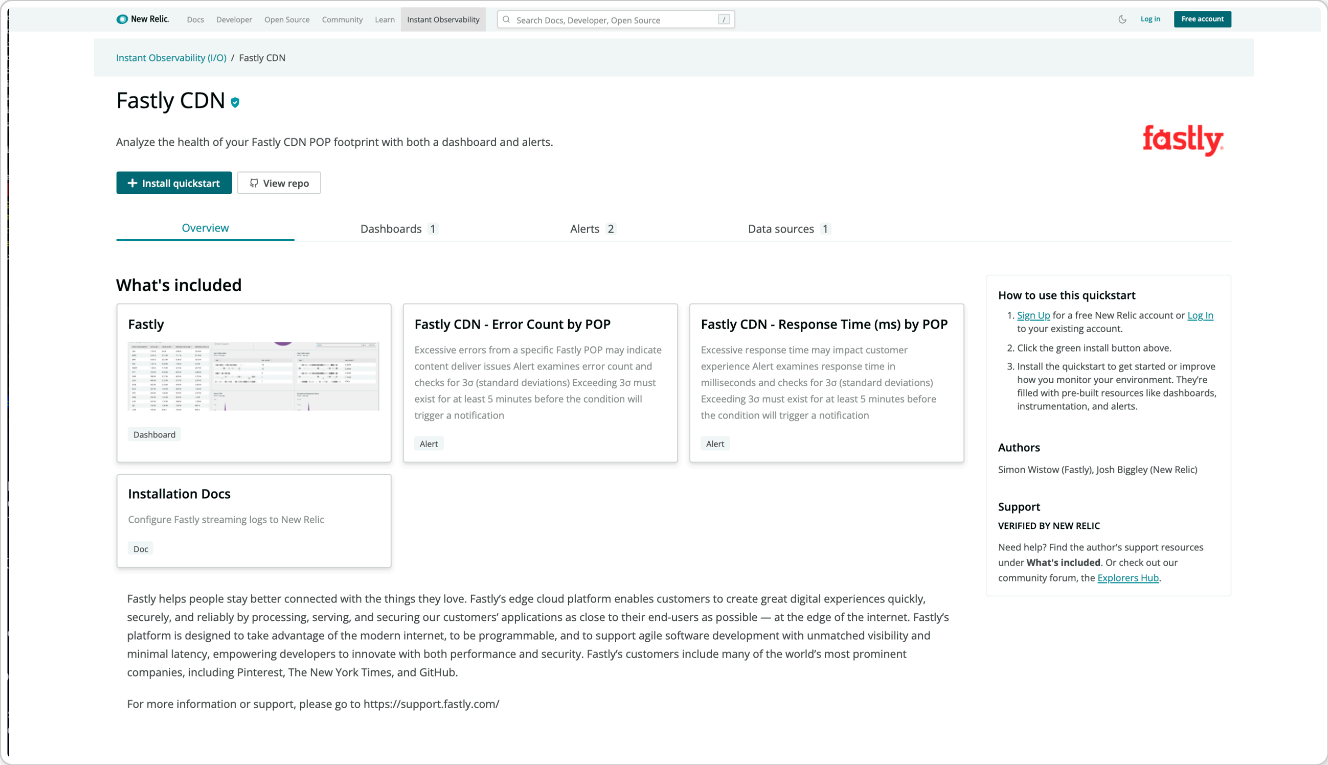
Gatsby Build
The Gatsby quickstart allows you to get visibility into the build time of your Gatsby Sites, using OpenTelemetry to collect each step as a span in a Distributed Trace. Monitor your build in real-time to highlight which steps are affecting performance, so you can improve them faster.

Gigamon Quickstart
Gigamon Cloud Suite enables collection of network traffic from every instance in the cloud collecting metadata of over 5,000 network traffic-related attributes for a holistic overview of and a much deeper level of security-related inspection.

Java
The New Relic Java quickstart provides insight into application performance, improves uptime, and reduces latency. Monitoring is reported using metric time-slice and event data, and all results are displayed in easy-to-use, visual dashboards.

Or navigate directly to the Fastly dashboard page.

2. Click **Install quickstart**.



New Relic.
Docs
Developer
Open Source
Community
Learn
Instant Observability
Search Docs, Developer, Open Source
Log in
Free account
fastly

Instant Observability (I/O) / Fastly CDN

Fastly CDN

Analyze the health of your Fastly CDN POP footprint with both a dashboard and alerts.

Install quickstart
View repo
Overview
Dashboards 1
Alerts 2
Data sources 1

What's included

Fastly

Dashboard

Fastly CDN - Error Count by POP

Excessive errors from a specific Fastly POP may indicate content deliver issues Alert examines error count and checks for 3σ (standard deviations) Exceeding 3σ must exist for at least 5 minutes before the condition will trigger a notification

Alert

Fastly CDN - Response Time (ms) by POP

Excessive response time may impact customer experience Alert examines response time in milliseconds and checks for 3σ (standard deviations) Exceeding 3σ must exist for at least 5 minutes before the condition will trigger a notification

Alert

Installation Docs

Configure Fastly streaming logs to New Relic

Doc

Fastly helps people stay better connected with the things they love. Fastly's edge cloud platform enables customers to create great digital experiences quickly, securely, and reliably by processing, serving, and securing our customers' applications as close to their end-users as possible — at the edge of the internet. Fastly's platform is designed to take advantage of the modern internet, to be programmable, and to support agile software development with unmatched visibility and minimal latency, empowering developers to innovate with both performance and security. Fastly's customers include many of the world's most prominent companies, including Pinterest, The New York Times, and GitHub.

For more information or support, please go to https://support.fastly.com/

How to use this quickstart

1. Sign Up for a free New Relic account or Log In to your existing account.
2. Click the green install button above.
3. Install the quickstart to get started or improve how you monitor your environment. They're filled with pre-built resources like dashboards, instrumentation, and alerts.

Authors

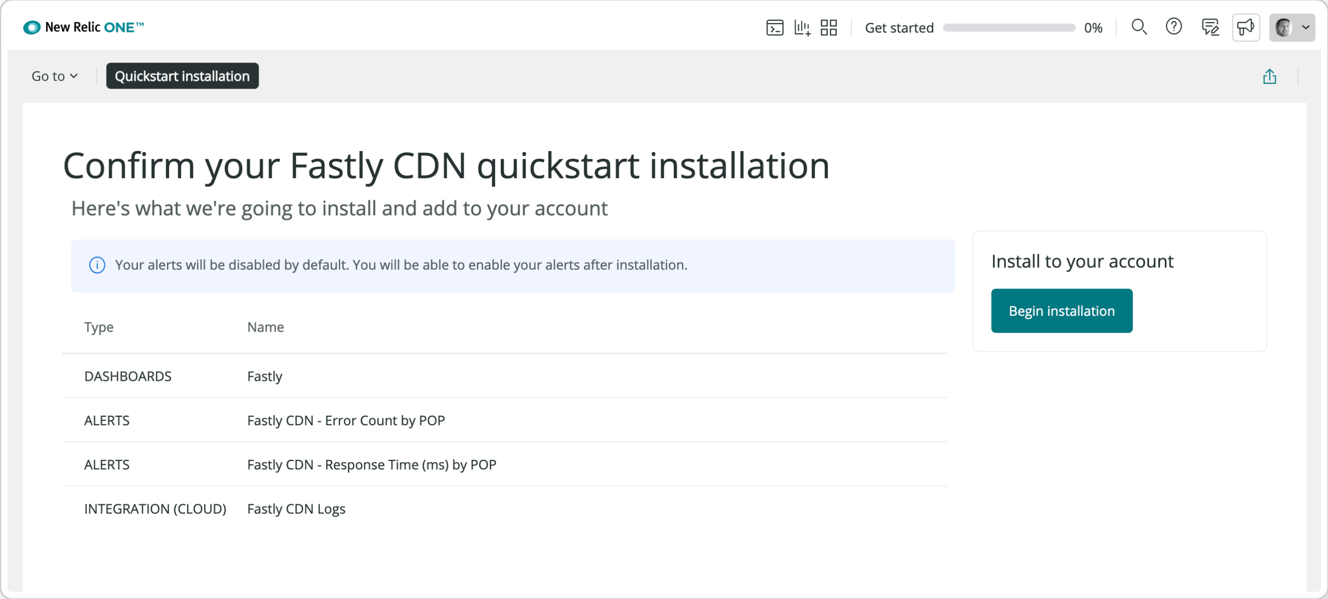
Simon Wistow (Fastly), Josh Biggley (New Relic)

Support

VERIFIED BY NEW RELIC

Need help? Find the author's support resources under **What's included**. Or check out our community forum, the Explorers Hub.

3. Click **Begin installation**.



New Relic ONE™
Get started
0%
Go to
Quickstart installation
Begin installation

Confirm your Fastly CDN quickstart installation

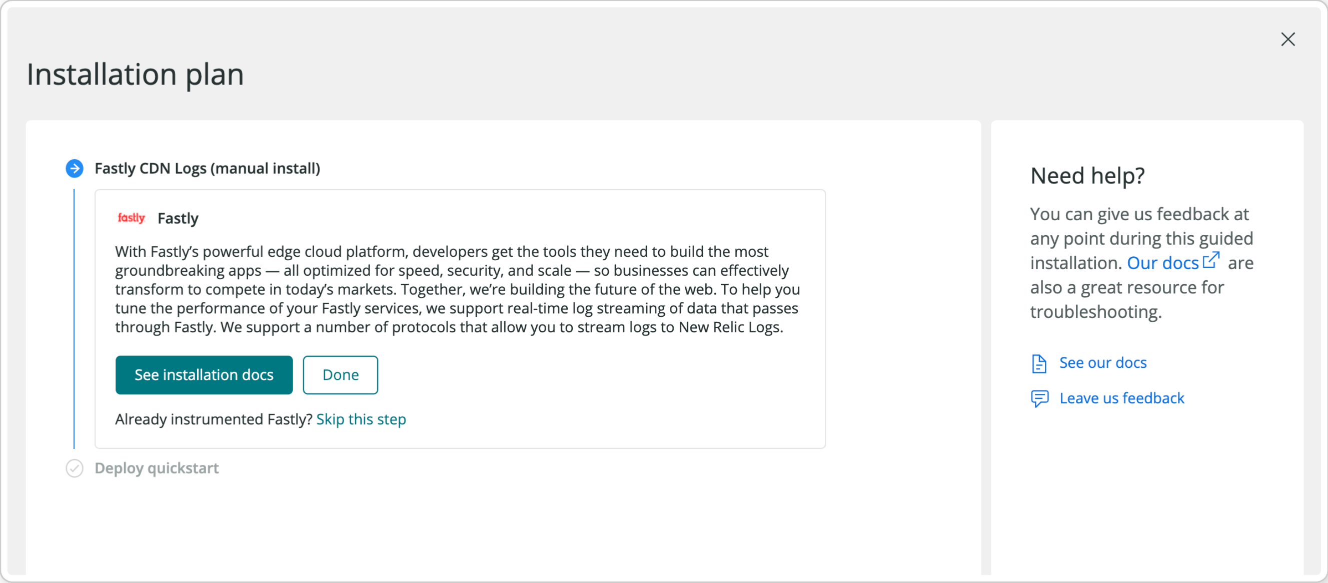
Here's what we're going to install and add to your account

Your alerts will be disabled by default. You will be able to enable your alerts after installation.

Type	Name
DASHBOARDS	Fastly
ALERTS	Fastly CDN - Error Count by POP
ALERTS	Fastly CDN - Response Time (ms) by POP
INTEGRATION (CLOUD)	Fastly CDN Logs

Install to your account

4. Click **Done** or **Skip this step**.



Installation plan

Fastly CDN Logs (manual install)

Fastly

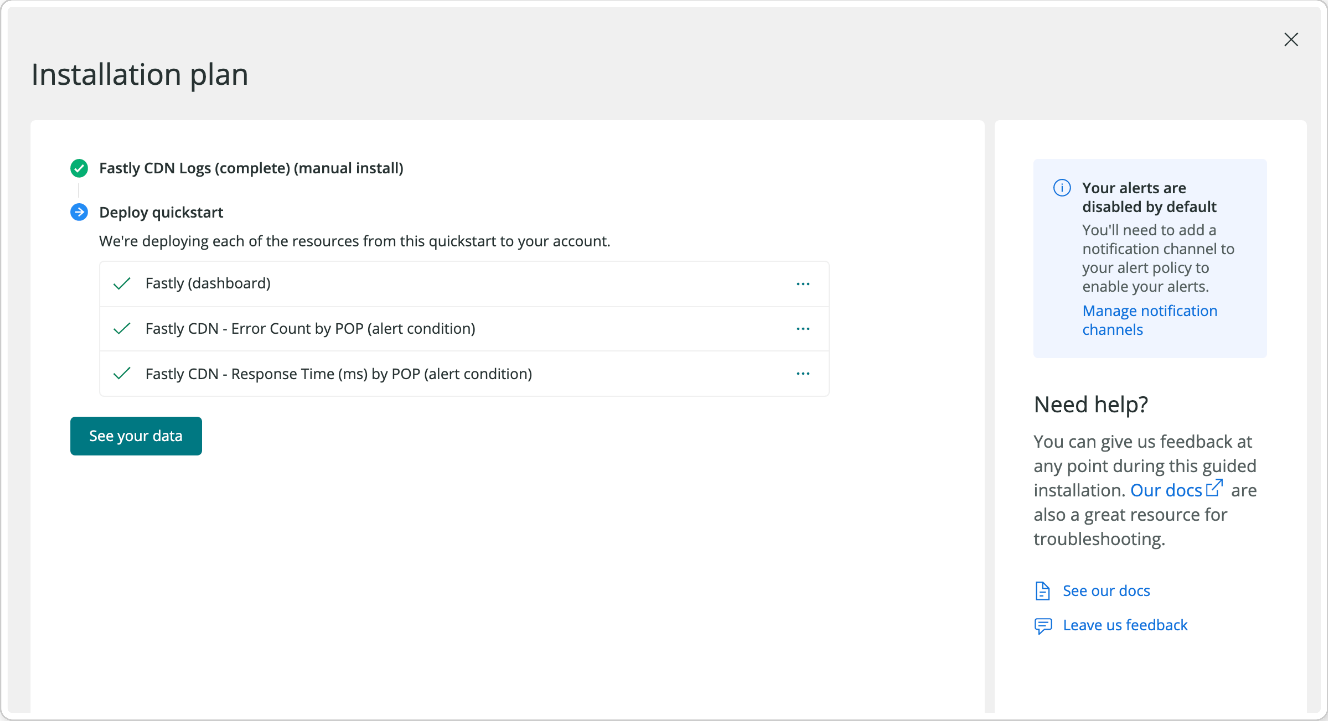
With Fastly's powerful edge cloud platform, developers get the tools they need to build the most groundbreaking apps — all optimized for speed, security, and scale — so businesses can effectively transform to compete in today's markets. Together, we're building the future of the web. To help you tune the performance of your Fastly services, we support real-time log streaming of data that passes through Fastly. We support a number of protocols that allow you to stream logs to New Relic Logs.

Already instrumented Fastly? Skip this step

Need help?

You can give us feedback at any point during this guided installation. Our docs are also a great resource for troubleshooting.

5. Click **See your data**.



Installation plan

Fastly CDN Logs (complete) (manual install)

Deploy quickstart

We're deploying each of the resources from this quickstart to your account.

- ✓ Fastly (dashboard)
- ✓ Fastly CDN - Error Count by POP (alert condition)
- ✓ Fastly CDN - Response Time (ms) by POP (alert condition)

Your alerts are disabled by default

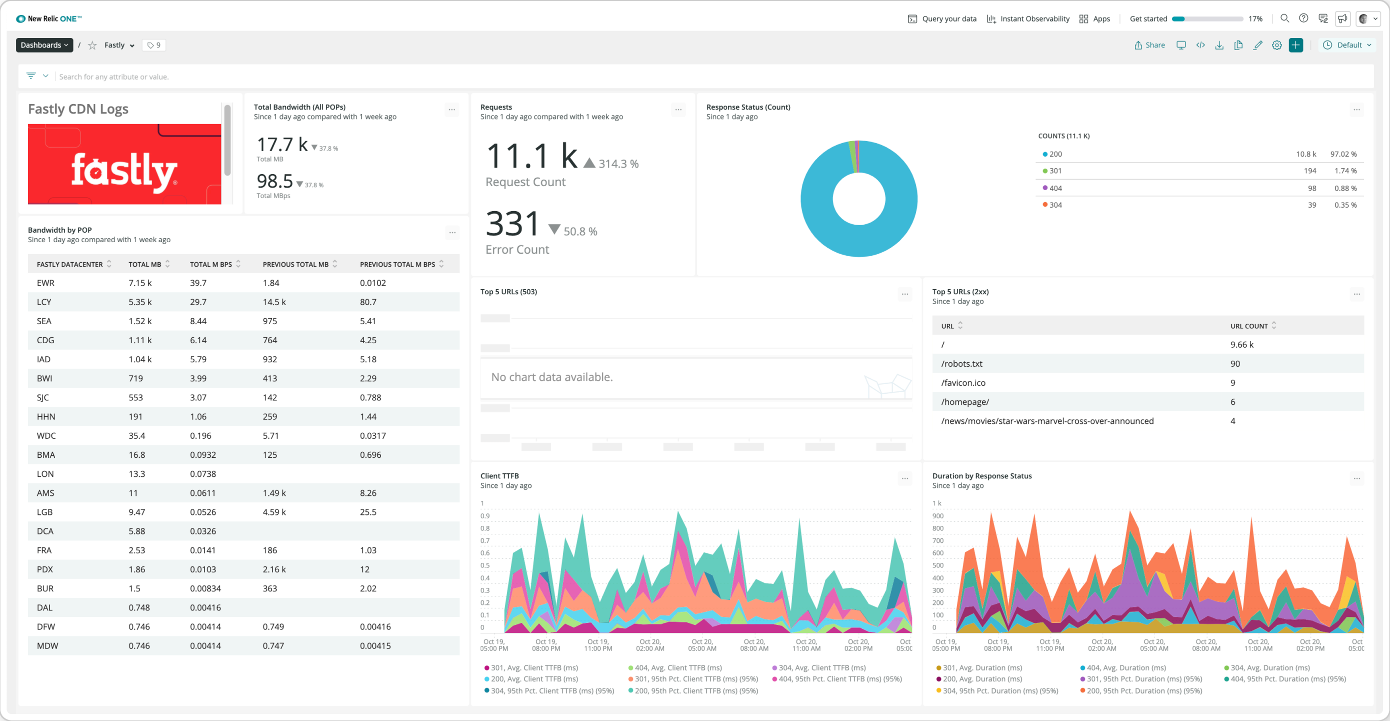
You'll need to add a notification channel to your alert policy to enable your alerts.

Manage notification channels

Need help?

You can give us feedback at any point during this guided installation. Our docs are also a great resource for troubleshooting.

The Fastly dashboard appears.


New Relic ONE™
Query your data
Instant Observability
Apps
Get started
17%
Dashboards
Fastly
9
Share
Default
Search for any attribute or value.
fastly
Oct 19, 05:00 PM
Oct 19, 08:00 PM
Oct 19, 11:00 PM
Oct 20, 02:00 AM
Oct 20, 05:00 AM
Oct 20, 08:00 AM
Oct 20, 11:00 AM
Oct 20, 02:00 PM
301, Avg. Client TTFB (ms)
404, Avg. Client TTFB (ms)
304, Avg. Client TTFB (ms)
200, Avg. Client TTFB (ms)
301, 95th Pct. Client TTFB (ms) (95%)
404, 95th Pct. Client TTFB (ms) (95%)
304, 95th Pct. Client TTFB (ms) (95%)
200, 95th Pct. Client TTFB (ms) (95%)
301, Avg. Duration (ms)
404, Avg. Duration (ms)
304, Avg. Duration (ms)
200, Avg. Duration (ms)
301, 95th Pct. Duration (ms) (95%)
404, 95th Pct. Duration (ms) (95%)
304, 95th Pct. Duration (ms) (95%)
200, 95th Pct. Duration (ms) (95%)

Fastly CDN Logs

Total Bandwidth (All POPs)
Since 1 day ago compared with 1 week ago

17.7 k ▼ 37.8 %
Total MB

98.5 ▼ 37.8 %
Total MBps

Requests
Since 1 day ago compared with 1 week ago

11.1 k ▲ 314.3 %
Request Count

331 ▼ 50.8 %
Error Count

Response Status (Count)
Since 1 day ago

COUNTS (11.1 K)		
● 200	10.8 k	97.02 %
● 301	194	1.74 %
● 404	98	0.88 %
● 304	39	0.35 %

Bandwidth by POP
Since 1 day ago compared with 1 week ago

FASTLY DATACENTER	TOTAL MB	TOTAL M BPS	PREVIOUS TOTAL MB	PREVIOUS TOTAL M BPS
EWR	7.15 k	39.7	1.84	0.0102
LCY	5.35 k	29.7	14.5 k	80.7
SEA	1.52 k	8.44	975	5.41
CDG	1.11 k	6.14	764	4.25
IAD	1.04 k	5.79	932	5.18
BWI	719	3.99	413	2.29
SJC	553	3.07	142	0.788
HHN	191	1.06	259	1.44
WDC	35.4	0.196	5.71	0.0317
BMA	16.8	0.0932	125	0.696
LON	13.3	0.0738		
AMS	11	0.0611	1.49 k	8.26
LGB	9.47	0.0526	4.59 k	25.5
DCA	5.88	0.0326		
FRA	2.53	0.0141	186	1.03
PDX	1.86	0.0103	2.16 k	12
BUR	1.5	0.00834	363	2.02
DAL	0.748	0.00416		
DFW	0.746	0.00414	0.749	0.00416
MDW	0.746	0.00414	0.747	0.00415

Top 5 URLs (503)

No chart data available.

Top 5 URLs (2xx)
Since 1 day ago

URL	URL COUNT
/	9.66 k
/robots.txt	90
/favicon.ico	9
/homepage/	6
/news/movies/star-wars-marvel-cross-over-announced	4

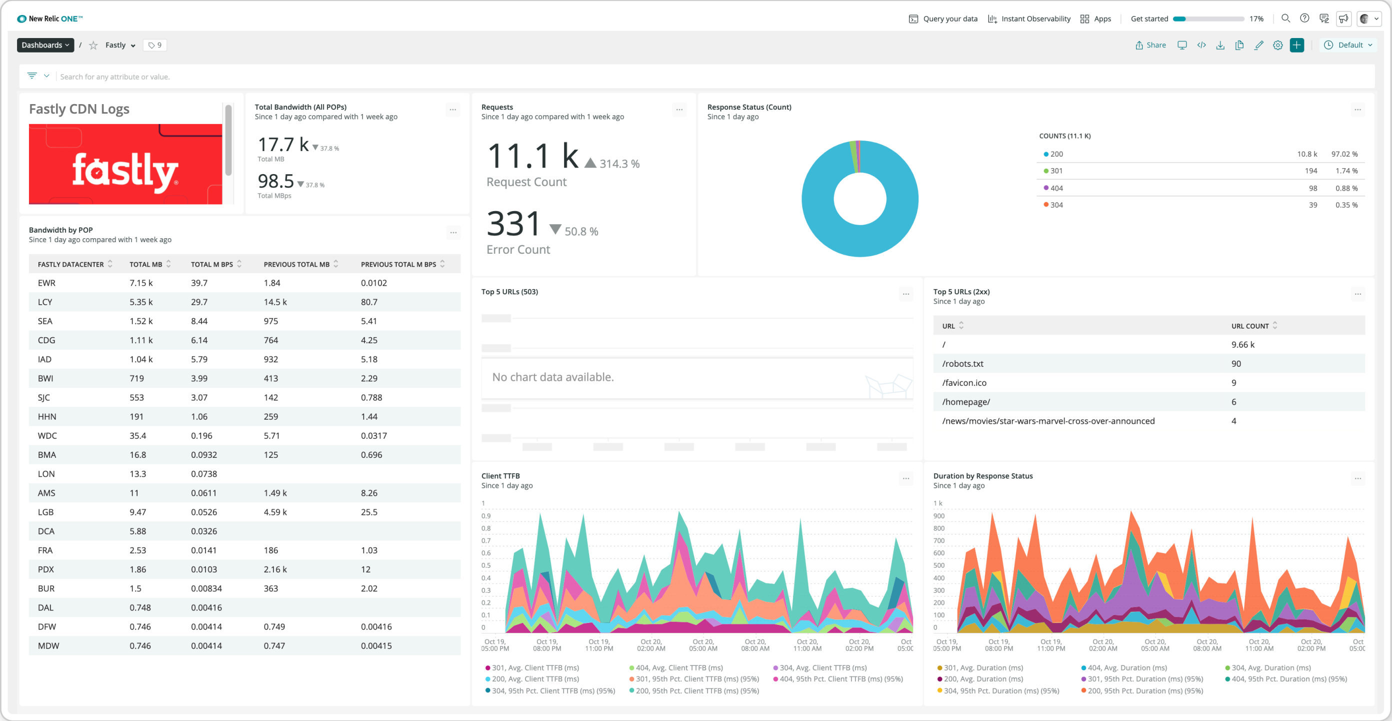
Client TTFB
Since 1 day ago

Duration by Response Status
Since 1 day ago

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the New Relic Logs area, click **Create endpoint**.
3. Fill out the **Create a New Relic Logs endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **License key / Insert key** field, enter your New Relic license key or Insert API key.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Using New Relic Instant Observability's prebuilt Fastly dashboard

New Relic I/O is an open source ecosystem of a wide variety of community-contributed monitoring quickstarts. We worked with New Relic to develop a prebuilt dashboard that highlights key metrics. Because the code is open source, you can customize it.


New Relic ONE™
Query your data
Instant Observability
Apps
Get started
17%
Dashboards
Fastly
9
Share
Default
Search for any attribute or value.
Fastly CDN Logs
fastly
Oct 19, 05:00 PM
Oct 19, 08:00 PM
Oct 19, 11:00 PM
Oct 20, 02:00 AM
Oct 20, 05:00 AM
Oct 20, 08:00 AM
Oct 20, 11:00 AM
Oct 20, 02:00 PM
Oct 20, 05:00 PM
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1 k
900
800
700
600
500
400
300
200
100
0
301, Avg. Client TTFB (ms)
404, Avg. Client TTFB (ms)
304, Avg. Client TTFB (ms)
200, Avg. Client TTFB (ms)
301, 95th Pct. Client TTFB (ms) (95%)
404, 95th Pct. Client TTFB (ms) (95%)
304, 95th Pct. Client TTFB (ms) (95%)
200, 95th Pct. Client TTFB (ms) (95%)
301, Avg. Duration (ms)
404, Avg. Duration (ms)
304, Avg. Duration (ms)
200, Avg. Duration (ms)
301, 95th Pct. Duration (ms) (95%)
404, 95th Pct. Duration (ms) (95%)
304, 95th Pct. Duration (ms) (95%)
200, 95th Pct. Duration (ms) (95%)

Total Bandwidth (All POPs)
Since 1 day ago compared with 1 week ago

17.7 k ▼ 37.8 %
Total MB

98.5 ▼ 37.8 %
Total MBps

Requests
Since 1 day ago compared with 1 week ago

11.1 k ▲ 314.3 %
Request Count

331 ▼ 50.8 %
Error Count

Response Status (Count)
Since 1 day ago

COUNTS (11.1 K)

● 200	10.8 k	97.02 %
● 301	194	1.74 %
● 404	98	0.88 %
● 304	39	0.35 %

Bandwidth by POP
Since 1 day ago compared with 1 week ago

FASTLY DATACENTER	TOTAL MB	TOTAL M BPS	PREVIOUS TOTAL MB	PREVIOUS TOTAL M BPS
EWR	7.15 k	39.7	1.84	0.0102
LCY	5.35 k	29.7	14.5 k	80.7
SEA	1.52 k	8.44	975	5.41
CDG	1.11 k	6.14	764	4.25
IAD	1.04 k	5.79	932	5.18
BWI	719	3.99	413	2.29
SJC	553	3.07	142	0.788
HHN	191	1.06	259	1.44
WDC	35.4	0.196	5.71	0.0317
BMA	16.8	0.0932	125	0.696
LON	13.3	0.0738		
AMS	11	0.0611	1.49 k	8.26
LGB	9.47	0.0526	4.59 k	25.5
DCA	5.88	0.0326		
FRA	2.53	0.0141	186	1.03
PDX	1.86	0.0103	2.16 k	12
BUR	1.5	0.00834	363	2.02
DAL	0.748	0.00416		
DFW	0.746	0.00414	0.749	0.00416
MDW	0.746	0.00414	0.747	0.00415

Top 5 URLs (503)

No chart data available.

Top 5 URLs (2xx)
Since 1 day ago

URL	URL COUNT
/	9.66 k
/robots.txt	90
/favicon.ico	9
/homepage/	6
/news/movies/star-wars-marvel-cross-over-announced	4

Client TTFB
Since 1 day ago

Duration by Response Status
Since 1 day ago

Configure your log messages

Data sent to New Relic Logs must be serialized as a JSON object. The JSON object should include the following fields:

- `timestamp`: when the request occurred. The timestamp must be specified as milliseconds since Epoch to override the New Relic timestamp. If timestamp is not included, Fastly will generate a timestamp.
- `client_ip`: the IP address of the client making the HTTP request.
- `url`: the URL of the request.
- `resp_status`: the HTTP status code of the request.
- `client_resp_header_size_write`: the size of the response header.
- `client_resp_body_size_write`: the size of the response body.
- `fastly_datacenter`: the three-character identifying code of the FASTLY_POP in which the current instance is running.
- `time_elapsed`: the time since the request started.
- `service_id`: the identifier for the Fastly service that is processing the current request.

An example of the suggested more descriptive JSON log object is shown below.

```
{
  "timestamp": 1661976797605,
  "cache_status": "ERROR",
  "client_ip": "127.0.0.1",
  "client_device_type": "Chromebook",
  "client_os_name": "Ubuntu",
  "client_os_version": "17.10 (Artful Aardvark)",
  "client_browser_name": "Firefox",
  "client_browser_version": "113.0",
  "client_as_name": "zayo bandwidth",
  "client_as_number": "1234",
  "client_connection_speed": "broadband",
  "client_port": 63850,
  "client_rate_bps": 0,
  "client_recv_bytes": 74,
  "client_requests_count": 1,
  "client_resp_body_size_write": 56789,
  "client_resp_header_size_write": 1234,
  "client_resp_ttfb": 1.342,
  "client_rtt_us": 6818,
  "content_type": "text/html; charset=utf-8",
  "domain": "example.com",
  "fastly_datacenter": "HNL",
  "fastly_host": "cache-hnl00001",
  "fastly_is_edge": true,
  "fastly_region": "US-Pacific",
  "fastly_server": "cache-hnl00001-HNL",
  "host": "example.com",
  "origin_host": "example.com",
  "origin_name": "n/a",
  "request": "GET",
  "request_method": "GET",
  "request_accept_charset": "utf-8",
  "request_accept_language": "en-US",
  "request_referer": "",
  "request_user_agent": "curl/7.68.0",
  "resp_status": "503",
  "response": "Backend unavailable, connection timeout",
  "service_id": "000q0j0WE0f00z0KEVj5I0",
  "service_version": "29",
  "status": "503",
  "time_start": "2023-05-18T23:21:52Z",
  "time_end": "2023-05-18T23:21:53Z",
  "time_elapsed": 237,
  "tls_cipher": "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256",
  "tls_version": "TLS 1.2",
  "url": "/",
  "user_agent": "curl/7.68.0",
  "user_city": "san francisco",
  "user_country_code": "US",
  "user_continent_code": "NA",
  "user_region": "CA"
}
```

While the prebuilt dashboard won't break if the JSON object doesn't include all of the expected fields, certain charts won't have data.

In addition to the suggested fields, you can include fields of your choosing in the JSON object for your own purposes or to maintain backward compatibility with existing dashboards you've built.

Install the Fastly dashboard

Follow these instructions to install the Fastly dashboard quickstart:

1. Select the Fastly dashboard quickstart from the New Relic marketplace or navigate directly to the Fastly dashboard page.
2. Click **Install quickstart**.
3. Click **Begin installation**.
4. Click **Done** or **Skip this step**.
5. Click **See your data**. The Fastly dashboard appears.

* * *

Log streaming: New Relic OTLP

/en/guides/log-streaming-newrelic-otlp

Fastly's Real-Time Log Streaming feature can send log files to New Relic OTLP.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding New Relic OTLP as a logging endpoint for Fastly services, you will need to:

- Register for a New Relic account.
- Obtain your license key for reporting data.
- If you are using New Relic Infinite tracing, find your New Relic Trace Observer URL in the New Relic Infinite Tracing Settings. You'll need to copy the endpoint value under **For OTLP HTTP**.

Adding New Relic OTLP as a logging endpoint

Follow these instructions to add New Relic OTLP as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the **New Relic OTLP** area, click **Create endpoint**.
3. Fill out the **Create a New Relic OTLP endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.

  - In the **License key / Insert key** field, enter your New Relic license key or Insert API key.
  - From the **Region** controls, select the region to stream logs to.
4. *(Optional)* If you are using New Relic Infinite Tracing, click **Advanced options** and enter your **New Relic Trace Observer URL** in the **Trace Observer URL** field.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.
7. Follow the instructions below to instrument your VCL service.

Instrumenting your VCL service

After setting up your endpoint, you will need to add custom VCL to gather the data and generate Open Telemetry traces at the edge.

1. Place the following functions at the top of your custom VCL file.

```
sub time_now_ns STRING {
  declare local var.time_now INTEGER;
  set var.time_now = std.atoi(time.start.usec);
  set var.time_now += std.atoi(time.elapsed.usec);
  set var.time_now *= 1000;
  return var.time_now;
}
sub random_8bit_identifier STRING {
  declare local var.id STRING;
  set var.id = randomstr(16, "0123456789abcdef");
  return var.id;
}
sub random_16bit_identifier STRING {
  declare local var.id STRING;
  set var.id = randomstr(32, "0123456789abcdef");
  return var.id;
}

sub otel_resource STRING {
    declare local var.str STRING;
    set var.str = {"{ "attributes": [ "}
    {"{ "key": "service.name",           "value": { "stringValue": "Fastly www"
    {"{ "key": "telemetry.sdk.language", "value": { "stringValue": "vcl"
    {"{ "key": "telemetry.sdk.name",     "value": { "stringValue": "opentelemetry
    {"{ "key": "telemetry.sdk.version",  "value": { "stringValue": "1.0.1"
    {"{ "key": "host.name",              "value": { "stringValue": ""} server.ide
    {"], "droppedAttributesCount": 0 }"};
    return var.str;
}

sub otel_attributes_general STRING {
    declare local var.data STRING;
    set var.data = ""
```

```
   {"{ "key": "http.method",    "value": { "stringValue": ""} req.method {"" } },
   {"{ "key": "http.target",    "value": { "stringValue": ""} req.url {"" } },"}
   {"{ "key": "http.host",      "value": { "stringValue": ""} req.http.host {""
   {"{ "key": "http.protocol",  "value": { "stringValue": ""} req.protocol {"" }
   {"{ "key": "http.client_ip", "value": { "stringValue": ""} client.ip {"" } },

   {"{ "key": "fastly.restarts",     "value": { "stringValue": ""} req.restarts
   {"{ "key": "fastly.visits_this_service", "value": { "stringValue": ""} fastly
   {"{ "key": "fastly.server_role",  "value": { "stringValue": ""} req.http.x-tra
   {"{ "key": "fastly.server_ip",    "value": { "stringValue": ""} server.ip {""
   {"{ "key": "fastly.server_id",    "value": { "stringValue": ""} server.identi
   {"{ "key": "fastly.server_role",  "value": { "stringValue": ""} req.http.x-tra
   {"{ "key": "fastly.vcl_version",  "value": { "stringValue": ""} req.vcl.versi
   {"{ "key": "fastly.pop",          "value": { "stringValue": ""} server.datace
   {"{ "key": "fastly.workspace.overflowed",  "value": { "stringValue": ""} works
   {"{ "key": "fastly.workspace.bytes_total", "value": { "stringValue": ""} works
   {"{ "key": "fastly.workspace.bytes_free",  "value": { "stringValue": ""} works
   ;
   return var.data;
}

sub telem_start_backend_fetch {
  set bereq.http.traceparent = "00-" req.http.x-trace-id + "-" + req.http.x-trace

  # Avoid leaking internal headers to backends
  unset bereq.http.x-trace-id;
  unset bereq.http.x-trace-parent-span-id;
  unset bereq.http.x-trace-server-role;

  # Leapfrog cloud service infra that creates 'ghost spans'
  set bereq.http.x-traceparent = bereq.http.traceparent;
}
```

2. Add the following VCL to the start of the `vcl_recv` subroutine to record a span for the whole of a request's lifetime within a Fastly POP.

```
if (req.restarts == 0) {
  set req.http.x-trace-vcl-span-id = random_8bit_identifier();
  if (req.http.traceparent ~ "^\d+-(\w+)-(\w+)-\d+$") {
    set req.http.x-trace-id = re.group.1;
    set req.http.x-trace-parent-span-id = re.group.2;
  } else {
    set req.http.x-trace-id = random_16bit_identifier();
  }
  set req.http.x-trace-server-role = if (fastly.ff.visits_this_service == 0, "ed
}
```

3. Add the following VCL to the end of the `vcl_log` subroutine, replacing `otel_collector_http` with the name of your endpoint.

```
declare local var.otel_resource STRING;
declare local var.otel_attribs STRING;
declare local var.time_start_ns STRING;
declare local var.time_now_ns STRING;
set var.time_start_ns = time.start.usec "000";
set var.time_now_ns = time_now_ns();
set var.otel_resource = otel_resource();
set var.otel_attribs = otel_attributes_general();
log "syslog " req.service_id " otel_collector_http :: "
  {"{ "resourceSpans": [ { "}
    {""resource": "} var.otel_resource {", "}
    {""instrumentationLibrarySpans": [ { "spans": [ { "}
      {""traceId": ""} req.http.x-trace-id {"", "}
      {""spanId": ""} req.http.x-trace-vcl-span-id {"", "}
      if(req.http.x-trace-parent-span-id,
        {""parentSpanId": ""} req.http.x-trace-parent-span-id {"", "},
      "")
      {""name": "Fastly request processing", "}
      {""kind": 1, "}
      {""startTimeUnixNano": "} var.time_start_ns {", "}
      {""endTimeUnixNano": "} var.time_now_ns {", "}
      {""attributes": [ "}
        var.otel_attribs
        {"{ "key": "http.user_agent", "value": { "stringValue": ""} req.http.Use
        {"{ "key": "http.status_code", "value": { "stringValue": ""} resp.status
      {"], "}
      {""status": { "code":"} if (fastly_info.state ~ "ERROR", "2", "0") {" }, "
      {""links": [], "}
      {""droppedLinksCount": 0"}
    {"} ] } ]"}
  {"} ] }"}
;
```

4. Add a call to `telem_start_backend_fetch` and place it in the `vcl_miss` and `vcl_pass` subroutines to see spans that happen in backend systems as children of the Fastly spans.

5. Activate the service.

1. Review the information in our guide to setting up remote log streaming.

2. In the **New Relic OTLP** area, click **Create endpoint**.

3. Fill out the **Create a New Relic OTLP endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **License key / Insert key** field, enter your New Relic license key or Insert API key.
   - From the **Region** controls, select the region to stream logs to.

4. *(Optional)* If you are using New Relic Infinite Tracing, click **Advanced options** and enter your **New Relic Trace Observer URL** in the **Trace Observer URL** field.

5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

Instrumenting your Compute service

Open Telemetry is a vendor-neutral, open-source Observability framework for instrumenting, generating, collecting, and exporting telemetry data such as traces, metrics, and logs. Refer to the Open Telemetry docs and the Fastly Compute documentation for additional information about using Open Telemetry in Compute.

* * *

Log streaming: OpenStack

/en/guides/log-streaming-openstack

Fastly's Real-Time Log Streaming feature can send log files to OpenStack. OpenStack is an open-source platform for cloud-computing that many companies deploy as an infrastructure-as-a-service.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Adding OpenStack as a logging endpoint

Follow these instructions to add OpenStack as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the OpenStack area, click **Create endpoint**.
3. Fill out the **Create an OpenStack endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
   - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
   - In the **Auth URL** field, enter the URL used for OpenStack authentication (e.g., `https://auth.api.rackspacecloud.com/v1.0`).
   - In the **Bucket name** field, enter the name of the OpenStack bucket in which to store the logs.
   - In the **User** field, enter your OpenStack username.
   - In the **Access Key** field, enter your OpenStack access key.

   - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to OpenStack. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the OpenStack area, click **Create endpoint**.

3. Fill out the **Create an OpenStack endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Auth URL** field, enter the URL used for OpenStack authentication (e.g., `https://auth.api.rackspacecloud.com/v1.0`).
    - In the **Bucket name** field, enter the name of the OpenStack bucket in which to store the logs.
    - In the **User** field, enter your OpenStack username.
    - In the **Access Key** field, enter your OpenStack access key.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
    - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
    - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Oracle Cloud Storage

/en/guides/log-streaming-oracle-cloud-storage

Fastly's Real-Time Log Streaming feature can send log files to Oracle Cloud Storage using Oracle Cloud's S3-compatible API connectivity option. Oracle Cloud Storage is a static file storage service used by developers and IT teams.

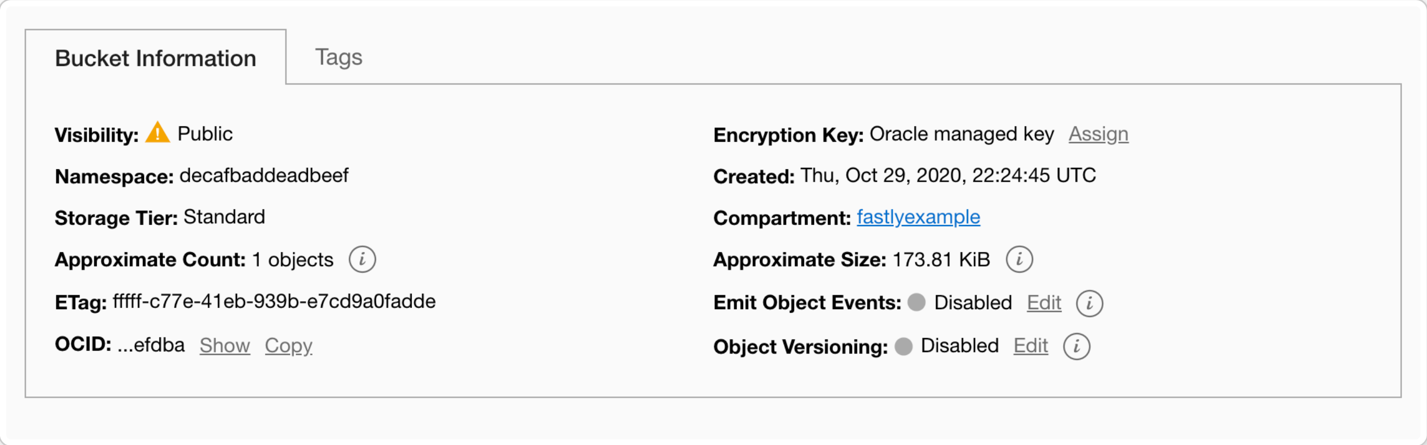
NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Oracle Cloud Storage as a logging endpoint for Fastly services, you need an Oracle **Customer Secret Key**, which consists of an **Access Key** and **Secret Key**. These need read and write permissions on the bucket.

You will also need to know the namespace identifier assigned to your bucket. You can find your namespace by clicking on the bucket and examining the **Bucket Information** tab (`decafbaddeadbeef` in this case).



Visibility: ⚠ Public
Namespace: decafbaddeadbeef
Storage Tier: Standard
Approximate Count: 1 objects
ETag: fffff-c77e-41eb-939b-e7cd9a0fadde
OCID: ...efdba Show Copy
Encryption Key: Oracle managed key Assign
Created: Thu, Oct 29, 2020, 22:24:45 UTC
Compartment: fastlyexample
Approximate Size: 173.81 KiB
Emit Object Events: ● Disabled Edit
Object Versioning: ● Disabled Edit

Adding Oracle Cloud Storage as a logging endpoints

NOTE

This logging endpoint is only available for Fastly's Deliver (VCL-based) services, not for Compute services.

After you've registered for an Oracle Cloud Storage account and created an Access Key, follow these instructions to add Oracle Cloud Storage as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. Click the Amazon Web Services S3 logo.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
   - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
   - In the **Access method** area, select **User Credentials**.
   - In the **Bucket name** field, enter the name of the Oracle Cloud Storage bucket in which to store the logs.

  - In the **Access key** field, enter the access key associated with the Oracle account.
  - In the **Secret key** field, enter the secret key associated with the Oracle account.
  - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. In the **Create a new S3 endpoint** area, click **Advanced options**.
5. Fill out the rest of the **Advanced options** as follows:
  - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
  - In the **Domain** field, enter `<namespace>.compat.objectstorage.<region>.oraclecloud.com`.
  - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
  - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
  - From the **Redundancy level** menu, select a setting. Valid values are **Standard** and **Infrequent Access**. This value defaults to **Standard**.
  - *(Optional)* In the **Server side encryption** area, select an encryption method to protect files that Fastly writes to your Oracle Cloud Storage bucket. Valid values are **None** and **AES-256**. Check out Oracle's guide to this feature, which is functionally identical to Amazon's implementation except for lack of support for a key management service.
6. Click **Create** to create the new logging endpoint.
7. Click **Activate** to deploy your configuration changes.

> NOTE
>
> Although Fastly continuously streams logs into Oracle Cloud, the Oracle Cloud website and API do not make files available for access until after their upload is complete.

Example format

The following is an example format string for sending data to Oracle Cloud Storage. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

* * *

Log streaming: Papertrail

/en/guides/log-streaming-papertrail

Fastly's Real-Time Log Streaming feature can send log files to Papertrail. Papertrail is a web-based log aggregation application used by developers and IT teams. Instructions for setting up remote log streaming via Papertrail are detailed in the Papertrail setup and configuration documentation.

If you're setting up Papertrail for a Compute service, review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

* * *

Log streaming: Scalyr

/en/guides/log-streaming-scalyr

Fastly's Real-Time Log Streaming feature can send log files to Scalyr (now known as DataSet). Scalyr pulls all your server logs and metrics into a centralized, searchable system in real time.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

If you don't already have a Scalyr account, you'll need to register for one. Follow the signup instructions on the Scalyr website.

Once you've signed up, navigate to the **API Keys** area in the **Settings** on your Scalyr dashboard and make note of your Scalyr Write Token. Scalyr uses this to associate data you send them with your account. You'll need this token when you set up your endpoint with Fastly.

If you're adding the Scalyr endpoint via the command line, instead of the web interface, you should also have your Fastly API token and the service ID and version number of the Fastly service for which you'll be enabling Scalyr logging.

Adding Scalyr as a logging endpoint

Follow these instructions to add Scalyr as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the Scalyr area, click **Create endpoint**.
3. Fill out the **Create a Scalyr endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Logfile** field, specify the log file name under which your logs will appear on Scalyr's Overview page. Defaults to `logplex`.
    - In the **Token** field, enter the Scalyr Write Token provided in the Scalyr dashboard.
    - From the **Region** menu, select the region to stream logs to.
4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Scalyr. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the Scalyr area, click **Create endpoint** button.

3. Fill out the **Create a Scalyr endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - *(Optional)* In the **Logfile** field, specify the log file name under which your logs will appear on Scalyr's Overview page. Defaults to `logplex`.
    - In the **Token** field, enter the Scalyr Write Token provided in the Scalyr dashboard.
    - From the **Region** menu, select the region to stream logs to.

4. Click **Create** to create the new logging endpoint.

5. Click **Activate** to deploy your configuration changes.

---

* * *

Log streaming: SFTP

/en/guides/log-streaming-sftp

Fastly's Real-Time Log Streaming feature can send log files to SFTP, a secure file transfer subsystem for the Secure Shell (SSH) protocol. Our SFTP endpoint supports both password-based authentication and SSH public-key authentication, with SSH public-key authentication being preferred. To learn more about SSH public-key authentication, or to learn how to generate public and private key pairs, see this guide.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Adding SFTP as a logging endpoint

Follow these instructions to add SFTP as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. In the SFTP area, click **Create endpoint**.
3. Fill out the **Create an SSH File Transfer Protocol (SFTP) endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Address** field, enter the hostname or IP address of the SFTP server. In the port field after the colon, enter the port number you're using for SFTP (the default is `22`).
    - In the **Path** field, enter the path to use for storing log files. Leaving the default `/` in this field means the files will be saved in the root path. We describe this variable in more detail in our guide on changing where log files are written.

> TIP
>
> If you save logs on the SFTP server, make sure the directory already exists.

    - In the **User** field, enter the username used to authenticate to the SFTP server.
    - In the **Known hosts** field, enter a Host key for each Host you can connect to over SFTP. Each Host key you enter must be on its own line. Known hosts entries should match what's stored in your `known_hosts` file located in your home directory (or the local account settings if you're working with macOS or a Windows operating system). A known hosts entry looks like this:

      `1.2.3.4 ecdsa-sha2-nistp256 aBc123xYz…`

      where the `1.2.3.4` is the SFTP IP address, `ecdsa-sha2-nistp256` is your Host key algorithm, and `aBc123xYz…` is your public key.
    - In the **Secret key** field, enter the SSH secret key used to connect to the server. If both Secret key and Password are entered, the Secret key will be used in preference.
    - In the **Password** field, enter the password used to authenticate to the SFTP server. If both Password and Secret key are entered, the Secret key will be used in preference.

  - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.

4. Click **Advanced options** and fill out the fields as follows:

  - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.

  - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.

  - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.

5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to an SFTP logging endpoint. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.

2. In the SFTP area, click **Create endpoint**.

3. Fill out the **Create an SSH File Transfer Protocol (SFTP) endpoint** fields as follows:

  - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
  - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
  - In the **Address** field, enter the hostname or IP address of the SFTP server. In the port field after the colon, enter the port number you're using for SFTP (the default is `22`).
  - In the **Path** field, enter the path to use for storing log files. Leaving the default `/` in this field means the files will be saved in the root path. We describe this variable in more detail in our guide on changing where log files are written.

> ✓ TIP
>
> If you save logs on the SFTP server, make sure the directory already exists.

  - In the **User** field, enter the username used to authenticate to the SFTP server.
  - In the **Known hosts** field, enter a Host key for each Host you can connect to over SFTP. Each Host key you enter must be on its own line. Known hosts entries should match what’s stored in your `known_hosts` file located in your home directory (or the local account settings if you're working with macOS or a Windows operating system). A known hosts entry looks like this:

    `1.2.3.4 ecdsa-sha2-nistp256 aBc123xYz…`

    where the `1.2.3.4` is the SFTP IP address, `ecdsa-sha2-nistp256` is your Host key algorithm, and `aBc123xYz…` is your public key.
  - In the **Secret key** field, enter the SSH secret key used to connect to the server. If both Secret key and Password are entered, the Secret key will be used in preference.
  - In the **Password** field, enter the password used to authenticate to the SFTP server. If both Password and Secret key are entered, the Secret key will be used in preference.
  - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.

4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

* * *

Log streaming: Shape Log Analysis

/en/guides/log-streaming-shape-log-analysis

Fastly's Real-Time Log Streaming feature can send log files to Shape Security. Shape Log Analysis uses anonymized attack data to analyze HTTP and application logs for insight into fraudulent activity and various types of attack that attempt to bypass security measures protecting origin servers.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Shape Log Analysis as a logging endpoint, send an email to fastly@f5.com with the subject line "Fastly Log Streaming Setup" to request a secure S3 bucket from Shape. In return, you will receive an email with details to help you complete the configuration of the logging endpoint including:

1. a note about the Log format value
2. a Bucket name
3. an Access key
4. a Secret key
5. a Path
6. a Domain

Each item will be specifically numbered in the email and the images in the configuration details below reflect those numbers.

Adding Shape Log Analysis as a logging endpoint

NOTE

This logging endpoint is only available for Fastly's Deliver (VCL-based) services, not for Compute services.

IMPORTANT

Shape Log Analysis setup and configuration uses Fastly's Amazon S3 log streaming endpoint to enable logs for storage and analysis. It places them in a secure S3 bucket managed by Shape that only accepts traffic from Fastly IP addresses.

After you've contacted fastly@f5.com and received the prerequisite information email, complete the following steps:

1. Review the information in our guide to setting up remote log streaming.
2. In the Amazon Web Services S3 area, click **Create endpoint**.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, copy and paste the following log format value to send log data to Shape's secure S3 bucket:

```
{
  "timestamp": "%{begin:%Y-%m-%dT%H:%M:%S%z}t",
  "ts": "%{time.start.sec}V",
  "id.orig_h": "%h", "status_code": "%>s",
  "method": "%m",
  "host": "%{Host}i",
  "uri": "%U%q",
  "accept_encoding": "%{Accept-Encoding}i",
  "request_body_len": "%{req.body_bytes_read}V",
  "response_body_len": "%{resp.body_bytes_written}V",
  "location": "%{Location}i",
  "x_forwarded_for": "%{X-Forwarded-For}i",
  "user_agent": "%{User-Agent}i",
  "referer": "%{Referer}i",
  "accept": "%{Accept}i",
  "accept_language": "%{Accept-Language}i",
  "content_type": "%{Content-Type}o",
  "geo_city": "%{client.geo.city}V",
  "geo_country_code": "%{client.geo.country_code}V",
  "is_tls": %{if(req.is_ssl, "true", "false")}V,
  "tls_version": "%{tls.client.protocol}V",
  "tls_cipher_request": "%{tls.client.cipher}V",
  "tls_cipher_req_hash": "%{tls.client.ciphers_sha}V",
  "tls_extension_identifiers_hash": "%{tls.client.tlsexts_sha}V"
}
```

    - In the **Access method** area, select **User Credentials**.
    - In the **Bucket name** field, enter the name of value #2 “Bucket Name” received in the Shape email response.
    - In the **Access key** field, enter the name of value #3 “Access Key” received in the Shape email response.
    - In the **Secret key** field, enter the name of value #4 “Secret Key” received in the Shape email response.

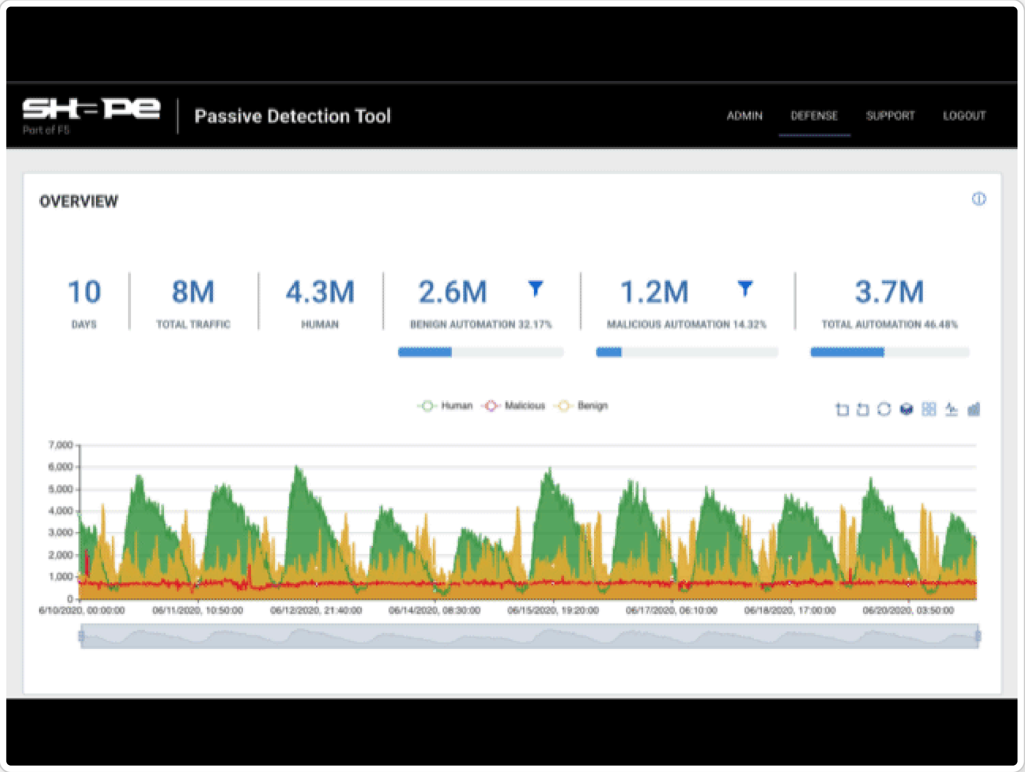
> **NOTE**
>
> Password management software may mistakenly treat the **Secret Key** field as a password field because of the way your web browser works. As such, that software may try to auto-fill this field with

your Fastly account password. If this happens to you, the integration with Fastly services won't work and you will need to enter **Secret Key** manually instead.

4. Click **Advanced options** and fill out the fields as follows:
    - In the **Path** field, enter the name of value #5 “Path” received in the email response.
    - In the **Domain** field, enter the name of value #6 “Domain” received in the email response.
    - In the **Select a log line format** area, select `Blank` to prevent prefixes from being added to log line messages and only report details in JSON log format. Our guide on changing log line formats provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Shape data analysis

Once Fastly logging configuration is complete, logs will be sent to Shape's secure S3 bucket for analysis. Typically, Shape collects approximately two weeks worth of log data to provide an analysis of attack traffic. After analysis is complete, Shape will send you a report with data on topics like Malicious Automation, Attack Surface (URLs), Account Takeover Bots, Suspicious Manual Fraud, and Top Bot Campaigns.


SHAPE
Part of F5
Passive Detection Tool
ADMIN
DEFENSE
SUPPORT
LOGOUT
OVERVIEW
10
DAYS
8M
TOTAL TRAFFIC
4.3M
HUMAN
2.6M
BENIGN AUTOMATION 32.17%
1.2M
MALICIOUS AUTOMATION 14.32%
3.7M
TOTAL AUTOMATION 46.48%
Human
Malicious
Benign
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
6/10/2020, 00:00:00
06/11/2020, 10:50:00
06/12/2020, 21:40:00
06/14/2020, 08:30:00
06/15/2020, 19:20:00
06/17/2020, 06:10:00
06/18/2020, 17:00:00
06/20/2020, 03:50:00

* * *

Log streaming: Splunk

/en/guides/log-streaming-splunk

Fastly's Real-Time Log Streaming feature can send log files to Splunk. Splunk is a web-based log analytics platform used by developers and IT teams.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

To use Splunk as a logging endpoint, you'll need to enable the HTTP Event Collector (HEC), create a token, and enable it. Follow the instructions on Splunk's website:

1. Enable HEC.
2. Create an HEC token.
3. Enable the HEC token.
4. Disable indexer acknowledgment for tokens used by Fastly to stream logs.

You'll need to remember the HEC token and find the URL for your collector. The URL structure depends on the type of Splunk instance you're using. Use the table below to find the URL structure for your Splunk instance.

Type	URL
Self hosted	`https://<hostname>:8088/services/collector/event`
Self-service Splunk Cloud plans	`https://input-<hostname>:8088/services/collector/event`
All other Splunk Cloud plans	`https://http-inputs-<hostname>:8088/services/collector/event`

While logged in to Splunk, you can find the hostname for the URL in your web browser's address bar.

Adding Splunk as a logging endpoint

After you've created a Splunk account and obtained your customer token, follow these instructions to add Splunk as a logging endpoint for Fastly services:

1. Review the information in our guide to setting up remote log streaming.
2. In the Splunk area, click **Create endpoint**.
3. Fill out the **Create a Splunk endpoint** fields as follows:
   - In the **Name** field, enter a human-readable name for the endpoint.
   - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
   - In the **Log format** field, enter an Apache-style string or VCL variables to use for log formatting. You can use our recommended log format.

   - In the **URL** field, enter the URL to send data to (e.g., `https://<splunk_host>:8088/services/collector/event/1.0`).
   - In the **Token** field, enter the token for the HEC.
   - *(Optional)* From the **Use TLS** controls, select whether or not to enable TLS. When you select Yes, additional TLS fields appear.
   - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. If you are using Splunk Enterprise see the Splunk Enterprise section below for more information.
   - In the **TLS CA certificate** field, enter the CA certificate used to verify that the origin's certificate is valid. It must be in PEM format. This is not required if your origin-side TLS certificate is signed by a well-known CA. See the using TLS CA certificates section for more information.
   - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the origin server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client certificate. A TLS client certificate allows your server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the backend server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your server to authenticate that Fastly is performing the connection.
   - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to append to a batch, if non-zero.
   - *(Optional)* In the **Maximum bytes** field, enter the maximum size of the log batch, if non-zero.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Recommended log format

We recommend using the following log format to send data to Splunk.

> NOTE
>
> All JSON sent to the Splunk HEC must have an event field. The event field can be text or nested JSON. There can also be other meta data in the payload. See the Splunk documentation for more information.

```
{
  "time":%{time.start.sec}V,
  "host":"%{Fastly-Orig-Host}i",
  "event":  {
    "service_id":"%{req.service_id}V",
    "time_start":"%{begin:%Y-%m-%dT%H:%M:%S%Z}t",
    "time_end":"%{end:%Y-%m-%dT%H:%M:%S%Z}t",
    "time_elapsed":%D,
    "client_ip":"%h",
  "client_as_name":"%{client.as.name}V",
  "client_as_number":"%{client.as.number}V",
  "client_connection_speed":"%{client.geo.conn_speed}V",
  "request":"%m",
  "protocol":"%H",
  "origin_host":"%v",
  "url":"%{cstr_escape(req.url)}V",
  "is_ipv6":%{if(req.is_ipv6, "true", "false")}V,
  "is_tls":%{if(req.is_ssl, "true", "false")}V,
  "tls_client_protocol":"%{cstr_escape(tls.client.protocol)}V",
  "tls_client_servername":"%{cstr_escape(tls.client.servername)}V",
  "tls_client_cipher":"%{cstr_escape(tls.client.cipher)}V",
  "tls_client_cipher_sha":"%{cstr_escape(tls.client.ciphers_sha )}V",
  "tls_client_tlsexts_sha":"%{cstr_escape(tls.client.tlsexts_sha)}V",
  "is_h2":%{if(fastly_info.is_h2, "true", "false")}V,
  "is_h2_push":%{if(fastly_info.h2.is_push, "true", "false")}V,
  "h2_stream_id":"%{fastly_info.h2.stream_id}V",
  "request_referer":"%{Referer}i",
  "request_user_agent":"%{User-Agent}i",
  "request_accept_content":"%{Accept}i",
  "request_accept_language":"%{Accept-Language}i",
  "request_accept_encoding":"%{Accept-Encoding}i",
  "request_accept_charset":"%{Accept-Charset}i",
  "request_connection":"%{Connection}i",
  "request_dnt":"%{DNT}i",
  "request_forwarded":"%{Forwarded}i",
  "request_via":"%{Via}i",
  "request_cache_control":"%{Cache-Control}i",
  "request_x_requested_with":"%{X-Requested-With}i",
  "request_x_att_device_id":"%{X-ATT-Device-Id}i",
  "request_x_forwarded_for":"%{X-Forwarded-For}i",
  "status":"%s",
  "content_type":"%{Content-Type}o",
  "response_state":"%{fastly_info.state}V",
  "response_age":"%{Age}o",
  "response_cache_control":"%{Cache-Control}o",
  "response_expires":"%{Expires}o",
  "response_last_modified":"%{Last-Modified}o",
  "response_tsv":"%{TSV}o",
  "server_datacenter":"%{server.datacenter}V",
  "server_ip":"%A",
  "geo_city":"%{client.geo.city.utf8}V",
  "geo_country_code":"%{client.geo.country_code}V",
  "geo_continent_code":"%{client.geo.continent_code}V",
  "geo_region":"%{client.geo.region}V",
  "req_header_size":%{req.header_bytes_read}V,
  "req_body_size":%{req.body_bytes_read}V,
  "resp_header_size":%{resp.header_bytes_written}V,
  "resp_body_size":%B,
  "socket_cwnd":%{client.socket.cwnd}V,
  "socket_nexthop":"%{client.socket.nexthop}V",
      "socket_tcpi_rcv_mss":%{client.socket.tcpi_rcv_mss}V,
      "socket_tcpi_snd_mss":%{client.socket.tcpi_snd_mss}V,
      "socket_tcpi_rtt":%{client.socket.tcpi_rtt}V,
      "socket_tcpi_rttvar":%{client.socket.tcpi_rttvar}V,
      "socket_tcpi_rcv_rtt":%{client.socket.tcpi_rcv_rtt}V,
      "socket_tcpi_rcv_space":%{client.socket.tcpi_rcv_space}V,
      "socket_tcpi_last_data_sent":%{client.socket.tcpi_last_data_sent}V,
      "socket_tcpi_total_retrans":%{client.socket.tcpi_total_retrans}V,
      "socket_tcpi_delta_retrans":%{client.socket.tcpi_delta_retrans}V,
      "socket_ploss":%{client.socket.ploss}V
    }
  }
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Splunk area, click **Create endpoint**.
3. Fill out the **Create a Splunk endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **URL** field, enter the URL to send data to (e.g., `https://<splunk_host>:8088/services/collector/event/1.0`).
    - In the **Token** field, enter the token for the HEC.
    - *(Optional)* From the **Use TLS** controls, select whether or not to enable TLS. When you select Yes, additional TLS fields appear.
    - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. If you are using Splunk Enterprise check out the Splunk Enterprise section below for more information.
    - In the **TLS CA certificate** field, enter the CA certificate used to verify that the Splunk server's certificate is valid. It must be in PEM format. This is not required if your Splunk-side TLS certificate is signed by a well-known CA. Check out the using TLS CA certificates section for more information.
    - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate Fastly to the Splunk server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your Splunk server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
    - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate Fastly to the Splunk server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your Splunk server to authenticate that Fastly is performing the connection.
    - *(Optional)* In the **Maximum logs** field, enter the maximum number of logs to append to a batch.
    - *(Optional)* In the **Maximum bytes** field, enter the maximum size of the log batch.

4. Click **Create** to create the new logging endpoint.
5. Click **Activate** to deploy your configuration changes.

Recommended log format

Data sent to Splunk HEC must be serialized in a way conforming to Splunk's expectations.

If your logs are not formatted properly, attempts at processing your logs by your Splunk endpoint may fail. Here's an example format string for sending data to Splunk:

```
{
    "time": 1652331824.730,
    "source": "fastly",
    "index": "main",
    "event": {
        "message": "Something happened",
        "severity": "INFO"
    }
}
```

Using TLS CA certificates

If you've installed your own TLS certificate in Splunk Enterprise or Splunk Cloud, you'll need to provide the corresponding CA certificate.

Splunk Cloud

For Splunk Cloud, the default set up has the following CA certificate:

```
-----BEGIN CERTIFICATE-----
MIIB/DCCAaGgAwIBAgIBADAKBggqhkjOPQQDAjB+MSswKQYDVQQDEyJTcGx1bmsg
Q2xvdWQgQ2VydGlmaWNhdGUgQXV0aG9yaXR5MRYwFAYDVQQHEw1TYW4gRnJhbmNp
c2NvMRMwEQYDVQQKEwpTcGx1bmsgSW5jMQswCQYDVQQIEwJDQTEVMBMGA1UECxMM
U3BsdW5rIENsb3VkMB4XDTE0MTExMDA3MDAxOFoXDTM0MTEwNTA3MDAxOFowfjEr
MCkGA1UEAxMiU3BsdW5rIENsb3VkIENlcnRpZmljYXRlIEF1dGhvcml0eTEWMBQG
A1UEBxMNU2FuIEZyYW5jaXNjbzETMBEGA1UEChMKU3BsdW5rIEluYzELMAkGA1UE
CBMCQ0ExFTATBgNVBAsTDFNwbHVuayBDbG91ZDBZMBMGByqGSM49AgEGCCqGSM49
AwEHA0IABPRRy9i3yQcxgMpvCSsI7Qe6YZMimUHOecPZWaGz5jEfB4+p5wT7dF3e
QrgjDWshVJZvK6KGO7nDh97GnbVXrTCjEDAOMAwGA1UdEwQFMAMBAf8wCgYIKoZI
zj0EAwIDSQAwRgIhALMUgLYPtICN9ci/ZOoXeZxUhn3i4wIo2mPKEWX0IcfpAiEA
8Jid6bzwUqAdDZPSOtaEBXV9uRIrNua0Qxl1S55TlWY=
-----END CERTIFICATE-----
```

Splunk Enterprise

Splunk Enterprise provides a set of default certificates, but we strongly recommend you configure your own certificates for your Fastly logging endpoint rather than relying on the default certificates. The certificates provided by Splunk Enterprise only specify a Common Name (CN), which cannot be used to properly verify the identity of the Splunk host presenting the certificate. Additionally, these certificates are less secure because the same root certificate is available in every Splunk Enterprise download. We encourage you to maintain the best possible security posture by configuring your own certificates rather than relying on the default certificates. The Splunk documentation provides a guide for configuring your own certificates.

* * *

Log streaming: Storj DCS

/en/guides/log-streaming-storj-dcs

Fastly's Real-Time Log Streaming feature can send log files to Storj DCS, a decentralized object storage service that is S3 compatible and end-to-end encrypted by default.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Storj DCS as a logging endpoint for Fastly services, you will need to create a Storj DCS account, project and access credentials, and a bucket that will store the log output.

Adding Storj DCS as a logging endpoint

> NOTE
>
> This logging endpoint is only available for Fastly's Deliver (VCL-based) services, not for Compute services.

Follow these instructions to add Storj DCS as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. Click the Amazon Web Services S3 logo.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Storj DCS bucket in which to store the logs.

   - In the **Access method** area, select **User Credentials**.
   - In the **Access key** field, enter the access key associated with the Storj DCS bucket.
   - In the **Secret key** field, enter the secret key associated with the Storj DCS bucket.
   - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. This value defaults to `3600` seconds.

4. Click **Advanced options** and fill out the fields as follows:
   - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
   - In the **Domain** field, enter the fully qualified hostname of any Storj DCS Gateway.
   - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.
   - In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
   - *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
   - From the **Redundancy level** menu, select a setting. This value defaults to **Standard**.
   - *(Optional)* In the **Server side encryption** area, select an encryption method to protect files that Fastly writes to your Storj DCS bucket. Valid values are **None** and **AES-256**.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Storj DCS. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

* * *

Log streaming: Sumo Logic

/en/guides/log-streaming-sumologic

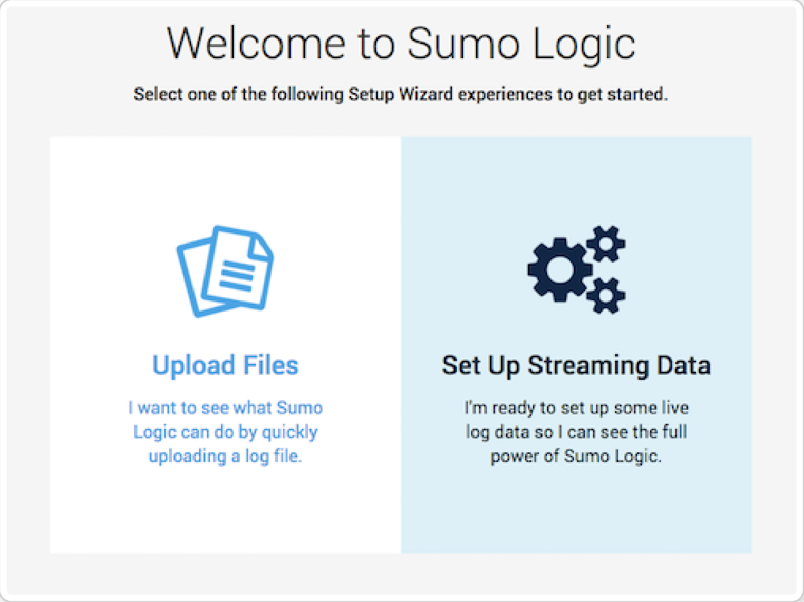
Fastly's Real-Time Log Streaming feature can send log files to Sumo Logic. Sumo Logic is a web-based log analytics platform used by developers and IT teams.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Setting up Sumo Logic

To use Sumo Logic as a logging endpoint, you'll need to create a Sumo Logic account, add a new source, and save the HTTP Source URL. Follow these instructions to add a new source in the Sumo Logic website:

1. The process starts with the Sumo Logic Setup Wizard, which appears immediately after you create your Sumo Logic account. If you already have an account, you can access the wizard by selecting **Setup Wizard** from the **Manage** menu at the top of the Sumo Logic application.

   Welcome to Sumo Logic

   Select one of the following Setup Wizard experiences to get started.

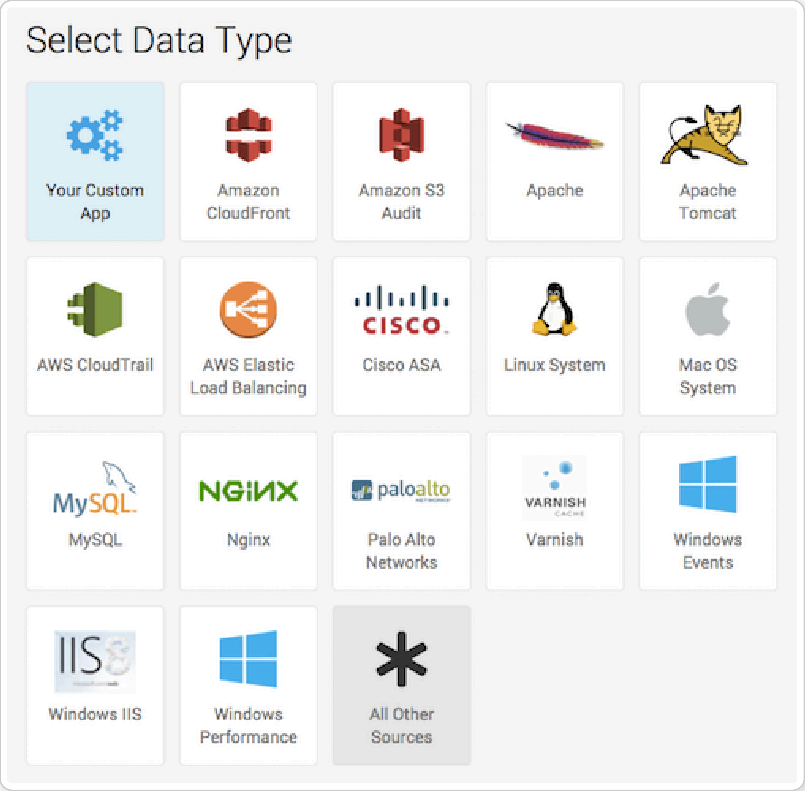
   Upload Files

   I want to see what Sumo Logic can do by quickly uploading a log file.

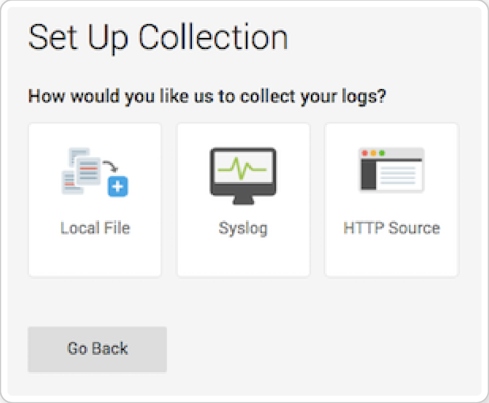
   Set Up Streaming Data

   I'm ready to set up some live log data so I can see the full power of Sumo Logic.

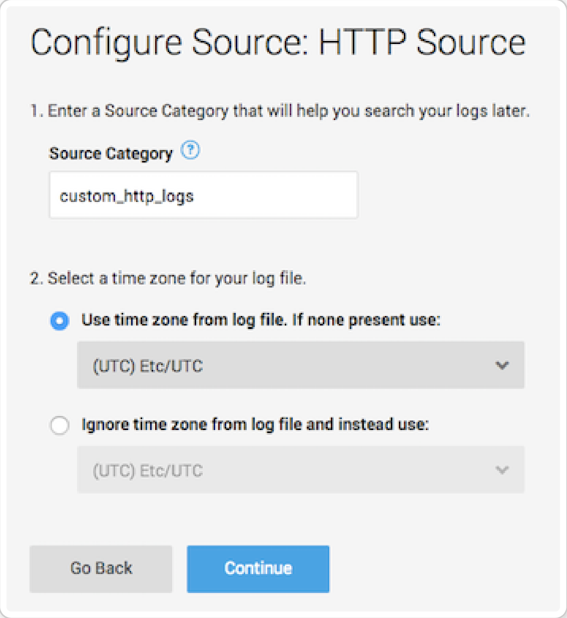
2. Click **Set Up Streaming Data**.


Select Data Type
Your Custom App
Amazon CloudFront
Amazon S3 Audit
Apache
Apache Tomcat
AWS CloudTrail
AWS Elastic Load Balancing
Cisco ASA
Linux System
Mac OS System
MySQL
Nginx
Palo Alto Networks
Varnish
Windows Events
Windows IIS
Windows Performance
All Other Sources

3. Click **All Other Sources**.


Set Up Collection
How would you like us to collect your logs?
Local File
Syslog
HTTP Source
Go Back

4. Click **HTTP Source**.



Configure Source: HTTP Source

1. Enter a Source Category that will help you search your logs later.

Source Category: custom_http_logs

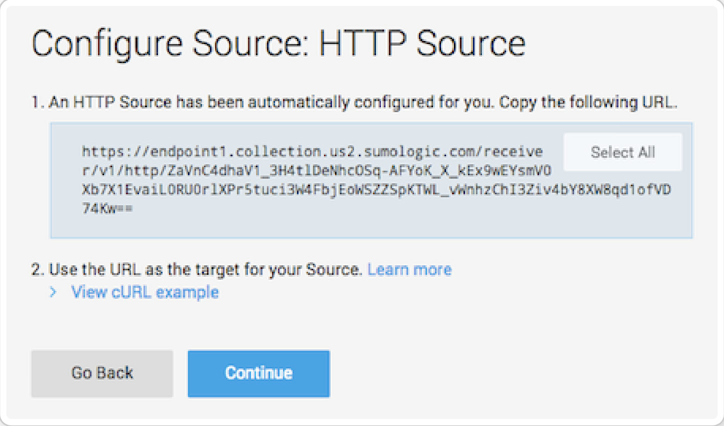
2. Select a time zone for your log file.

● Use time zone from log file. If none present use: (UTC) Etc/UTC

○ Ignore time zone from log file and instead use: (UTC) Etc/UTC

5. In the **Source Category** field, enter a human-readable name for the category (e.g., `fastly_cdn`) and select a time zone for your log file.

6. Click **Continue**. The HTTP Source URL appears.



Configure Source: HTTP Source

1. An HTTP Source has been automatically configured for you. Copy the following URL.

```
https://endpoint1.collection.us2.sumologic.com/receive
r/v1/http/ZaVnC4dhaV1_3H4tlDeNhcOSq-AFYoK_X_kEx9wEYsmVO
Xb7X1EvaiL0RU0rlXPr5tuci3W4FbjEoWSZZSpKTWL_vWnhzChI3Ziv4bY8XW8qd1ofVD
74Kw==
```

2. Use the URL as the target for your Source. Learn more

7. Copy the HTTP Source URL. You will enter this value in the Fastly web interface.

8. Click **Continue**. Sumo Logic will add the new source.

Adding Sumo Logic as a logging endpoint

After you've created a Sumo Logic account and obtained the HTTP Source URL, follow these instructions to add Sumo Logic as a logging endpoint for Fastly services:

1. Review the information in our guide to setting up remote log streaming.
2. In the Sumo Logic area, click **Create endpoint**.
3. Fill out the **Create a Sumo Logic endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log

placement for more information.

  - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
  - In the **Collector URL** field, enter the address of the HTTP Source URL you found in the Sumo Logic website.

4. *(Optional)* To change the log line format, click **Advanced options**. In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to Sumo Logic. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Sumo Logic area, click **Create endpoint**. The Create a Sumo Logic endpoint page appears.
3. Fill out the **Create a Sumo Logic endpoint** fields as follows:
   - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
   - In the **Collector URL** field, enter the address of the HTTP Source URL you found in the Sumo Logic website.
4. *(Optional)* To change the log line format, click **Advanced options**. In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.

5. Click **Create** to create the new logging endpoint.

6. Click **Activate** to deploy your configuration changes.

Troubleshooting

The Sumo Logic logging endpoint is designed for services with sustained levels of traffic. If you aren't seeing any logs in Sumo Logic, try waiting a bit.

* * *

	Log streaming: Syslog
🔗	/en/guides/log-streaming-syslog

Fastly's Real-Time Log Streaming feature can send log files to syslog-based logging software. Syslog is a widely used standard for message logging.

NOTE

Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Adding syslog as a logging endpoint

Follow these instructions to add syslog as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.

2. In the Syslog area, click **Create endpoint**.

3. Fill out the **Create a Syslog endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - In the **Syslog address** field, enter the domain name or IP address and port to which logs should be sent. Be sure this port can receive incoming TCP traffic from Fastly. See the firewall considerations section for more information.
    - *(Optional)* In the **Token** field, enter a string prefix (line prefix) to send in front of each log line.

    IMPORTANT

    To send a token, be sure the log line format is set to a format other than **Blank**.

   - From the **TLS** menu, select **No** to disable encryption for the syslog endpoint, or **Yes** to enable it. When you select Yes, additional TLS fields appear.
   - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the origin server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate to the origin server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client certificate. A TLS client certificate allows your server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.
   - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate to the backend server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

4. *(Optional)* To change the log line format, click **Advanced options**. In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information. This must be compatible with your Syslog configuration.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Example format

The following is an example format string for sending data to syslog. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

Adding separators or static strings

To insert a separator or other arbitrary string into the syslog endpoint format:

1. Create a new header with the following fields:
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter any suitable header name (for example, `http.X-Separator`).
    - In the **Source** field, enter any special character or string you want (for example, `"|"`).
2. Reference the new header variable in the log format box for your specific provider (for example, `req.http.X-Separator`).

1. Review the information in our guide to setting up remote log streaming for Compute. Additionally, our developer documentation provides more information about logging with Compute code written in our supported languages.
2. In the Syslog area, click **Create endpoint**.
3. Fill out the **Create a Syslog endpoint** fields as follows:
    - In the **Name** field, enter the endpoint name you specified in your Compute code. For example, in our Rust code example, the name is `my_endpoint_name`.
    - In the **Syslog address** field, enter the domain name or IP address and port to which logs should be sent. Be sure this port can receive incoming TCP traffic from Fastly. Check out the firewall considerations section for more information.
    - *(Optional)* In the **Token** field, enter a string prefix (line prefix) to send in front of each log line.

      > IMPORTANT
      >
      > To send a token, be sure the log line format is set to a format other than **Blank**.

    - From the **TLS** menu, select **No** to disable encryption for the syslog endpoint, or **Yes** to enable it. When you select Yes, additional TLS fields appear.
    - In the **TLS hostname** field, optionally enter a hostname to verify the logging destination server's certificate. This should be one of the Subject Alternative Name (SAN) fields for the certificate. Common Names (CN) are not supported. This field only appears when you select Yes from the Use TLS menu.
    - *(Optional)* In the **TLS CA certificate** field, copy and paste the certification authority (CA) certificate used to verify that the Syslog server's certificate is valid. The certificate you upload must be in PEM format. Consider uploading the certificate if it's not signed by a well-known certification authority. This value is not required if your TLS certificate is signed by a well-known authority. This field only appears when you select Yes from the Use TLS menu.
    - *(Optional)* In the **TLS client certificate** field, copy and paste the TLS client certificate used to authenticate Fastly to the Syslog server. The TLS client certificate you upload must be in PEM format and must be accompanied by a client key. A TLS client certificate allows your Syslog server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

   - *(Optional)* In the **TLS client key** field, copy and paste the TLS client key used to authenticate Fastly to the Syslog server. The TLS client key you upload must be in PEM format and must be accompanied by a TLS client certificate. A TLS client key allows your Syslog server to authenticate that Fastly is performing the connection. This field only appears when you select Yes from the Use TLS menu.

4. *(Optional)* To change the log line format, click **Advanced options**. In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information. This must be compatible with your Syslog configuration.
5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

Recommended log format

Log messages can take on any format you choose because the log line format selected above will affect the overall line sent to and parsed by Syslog.

Syslog facility and severity

The syslog output includes the following facility and severity values:

```
facility: local0
severity: info
```

Firewall considerations

Syslog has limited security features. For this reason, it's best to create a firewall for your syslog server and only accept TCP traffic on your configured port from our address blocks. Our list of IP address blocks is dynamic, so we recommend programmatically obtaining the list whenever possible.

* * *

Log streaming: Wasabi Hot Cloud Storage

/en/guides/log-streaming-wasabi-hot-cloud-storage

Fastly's Real-Time Log Streaming feature can send log files to Wasabi Hot Cloud Storage using Wasabi's S3-compatible API connectivity option. Wasabi Hot Cloud Storage is a static file storage service used by developers and IT teams.

> NOTE
>
> Fastly does not provide direct support for third-party services. Read Fastly's Terms of Service for more information.

Prerequisites

Before adding Wasabi Hot Cloud Storage as a logging endpoint for Fastly services, we recommend creating an Access Key to which you've given read and write permissions on the bucket.

Adding Wasabi Hot Cloud Storage as a logging endpoint

> NOTE
>
> This logging endpoint is only available for Fastly's Deliver (VCL-based) services, not for Compute services.

After you've registered for an Wasabi Hot Cloud Storage account and created an Access Key, follow these instructions to add Wasabi Hot Cloud Storage as a logging endpoint:

1. Review the information in our guide to setting up remote log streaming.
2. Click the Amazon Web Services S3 logo.
3. Fill out the **Create an Amazon S3 endpoint** fields as follows:
    - In the **Name** field, enter a human-readable name for the endpoint.
    - In the **Placement** area, select where the logging call should be placed in the generated VCL. Valid values are **Format Version Default**, **waf_debug (waf_debug_log)**, and **None**. Read our guide on changing log placement for more information.
    - In the **Log format** field, optionally enter an Apache-style string or VCL variables to use for log formatting. Consult the example format section for details.
    - *(Optional)* In the **Timestamp format** field, enter a timestamp format for log files. The default is an `strftime` compatible string. Our guide on changing where log files are written provides more information.
    - In the **Bucket name** field, enter the name of the Wasabi Hot Cloud Storage bucket in which to store the logs.
    - In the **Access method** area, select **User Credentials**.
    - In the **Access key** field, enter the access key associated with the Wasabi account. See Wasabi's Access Key Guide for more information.
    - In the **Secret key** field, enter the secret key associated with the Wasabi account. See Wasabi's Access Key Guide for more information.
    - *(Optional)* In the **Period** field, enter an interval (in seconds) to control how frequently your log files are rotated. Rotation entails the finalization of one file object and the start of a new one, never removing any previously created file object. This value defaults to `3600` seconds.
4. Click **Advanced options** and fill out the fields as follows:
    - *(Optional)* In the **Path** field, enter the path within the bucket to store the files. The path ends with a trailing slash. If this field is left empty, the files will be saved in the bucket's root path. Our guide on changing where log files are written provides more information.
    - In the **Domain** field, enter `s3.wasabisys.com`.
    - *(Optional)* In the **PGP public key** field, enter a PGP public key that Fastly will use to encrypt your log files before writing them to disk. You will only be able to read the contents by decrypting them with your private key. The PGP key should be in PEM (Privacy-Enhanced Mail) format. Read our guide on log encryption for more information.

- In the **Select a log line format** area, select the log line format for your log messages. Our guide on changing log line formats provides more information.
- *(Optional)* In the **Compression** field, select the compression format you want applied to the log files. Our guide on changing log compression options provides more information.
- The **Redundancy level** option is not useful as Wasabi only provides a single storage class which is most like the standard AWS S3 storage class. Wasabi's Object Storage Class documentation provides more information on using reduced redundancy storage.
- *(Optional)* In the **Server side encryption** area, select an encryption method to protect files that Fastly writes to your Wasabi Hot Cloud Storage bucket. Valid values are **None** and **AES-256**. See Wasabi's guide to this feature which is functionally identical to Amazon's implementation except for lack of support for a key management service.

5. Click **Create** to create the new logging endpoint.
6. Click **Activate** to deploy your configuration changes.

> NOTE
>
> Although Fastly continuously streams logs into Wasabi, the Wasabi website and API do not make files available for access until after their upload is complete.

Example format

The following is an example format string for sending data to Wasabi. Our discussion of format strings provides more information.

```
{
  "timestamp": "%{strftime(\{"%Y-%m-%dT%H:%M:%S%z"\}, time.start)}V",
  "client_ip": "%{req.http.Fastly-Client-IP}V",
  "geo_country": "%{client.geo.country_name}V",
  "geo_city": "%{client.geo.city}V",
  "host": "%{if(req.http.Fastly-Orig-Host, req.http.Fastly-Orig-Host, req.http.Host)}
  "url": "%{json.escape(req.url)}V",
  "request_method": "%{json.escape(req.method)}V",
  "request_protocol": "%{json.escape(req.proto)}V",
  "request_referer": "%{json.escape(req.http.referer)}V",
  "request_user_agent": "%{json.escape(req.http.User-Agent)}V",
  "response_state": "%{json.escape(fastly_info.state)}V",
  "response_status": %{resp.status}V,
  "response_reason": %{if(resp.response, "%22"+json.escape(resp.response)+"%22", "nul
  "response_body_size": %{resp.body_bytes_written}V,
  "fastly_server": "%{json.escape(server.identity)}V",
  "fastly_is_edge": %{if(fastly.ff.visits_this_service == 0, "true", "false")}V
}
```

* * *

Category: Non-Fastly services

These articles describe how non-Fastly services interoperate with Fastly.

Alibaba Object Storage Service

/en/guides/alibaba-object-storage-service

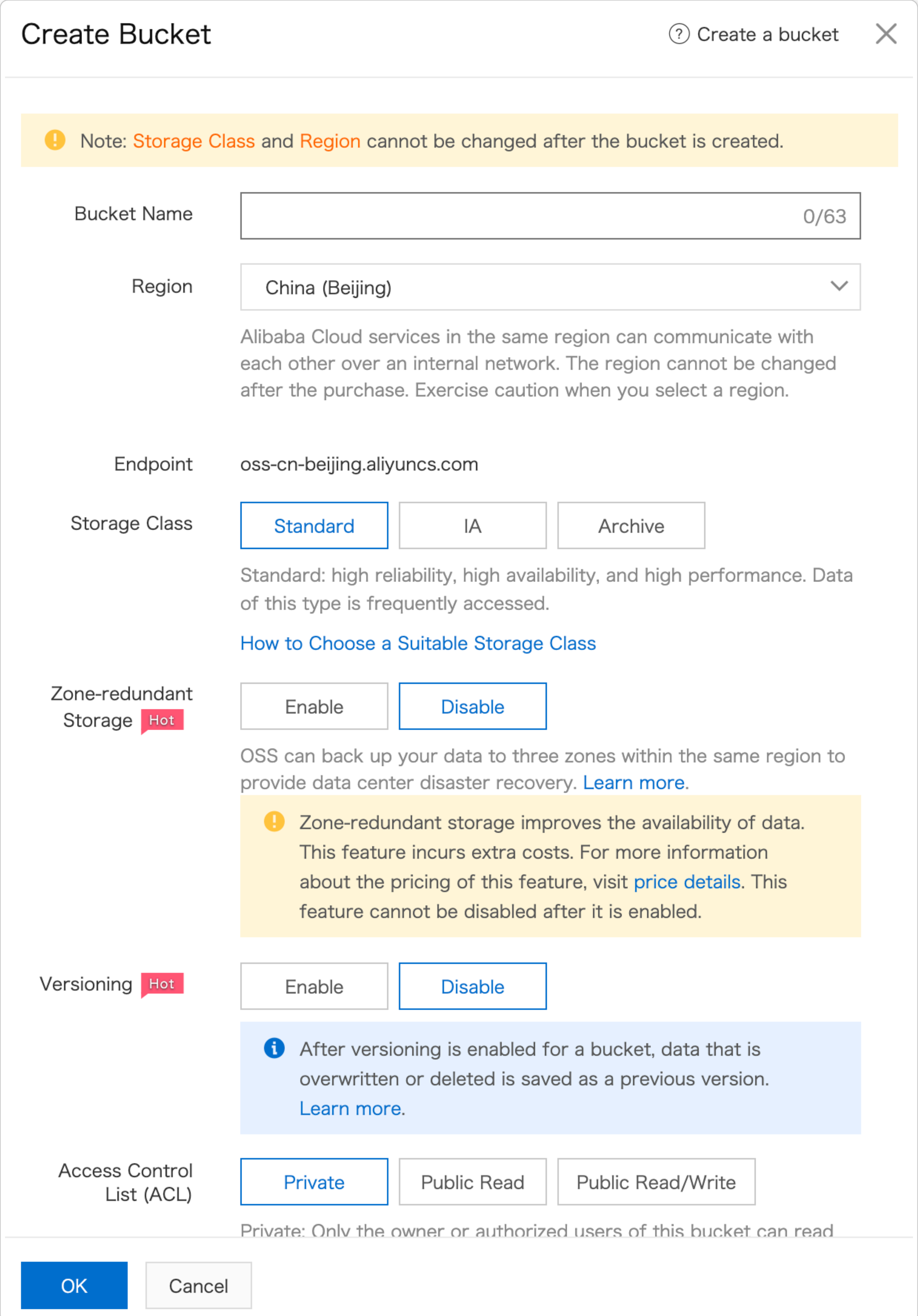
Alibaba Object Storage Service (OSS) can be used as an origin for Fastly for both public and private content.

Using OSS as an origin

To use OSS as an origin, follow the steps below.

Setting up and configuring your OSS account

1. Sign up for Alibaba Object Storage Service.
2. Create a bucket to store your origin's data.



Create Bucket

Note: Storage Class and Region cannot be changed after the bucket is created.

Bucket Name 0/63

Region: China (Beijing)

Alibaba Cloud services in the same region can communicate with each other over an internal network. The region cannot be changed after the purchase. Exercise caution when you select a region.

Endpoint: oss-cn-beijing.aliyuncs.com

Storage Class: Standard

Standard: high reliability, high availability, and high performance. Data of this type is frequently accessed.

How to Choose a Suitable Storage Class

Zone-redundant Storage Hot: Disable

OSS can back up your data to three zones within the same region to provide data center disaster recovery. Learn more.

Zone-redundant storage improves the availability of data. This feature incurs extra costs. For more information about the pricing of this feature, visit price details. This feature cannot be disabled after it is enabled.

Versioning Hot: Disable

After versioning is enabled for a bucket, data that is overwritten or deleted is saved as a previous version. Learn more.

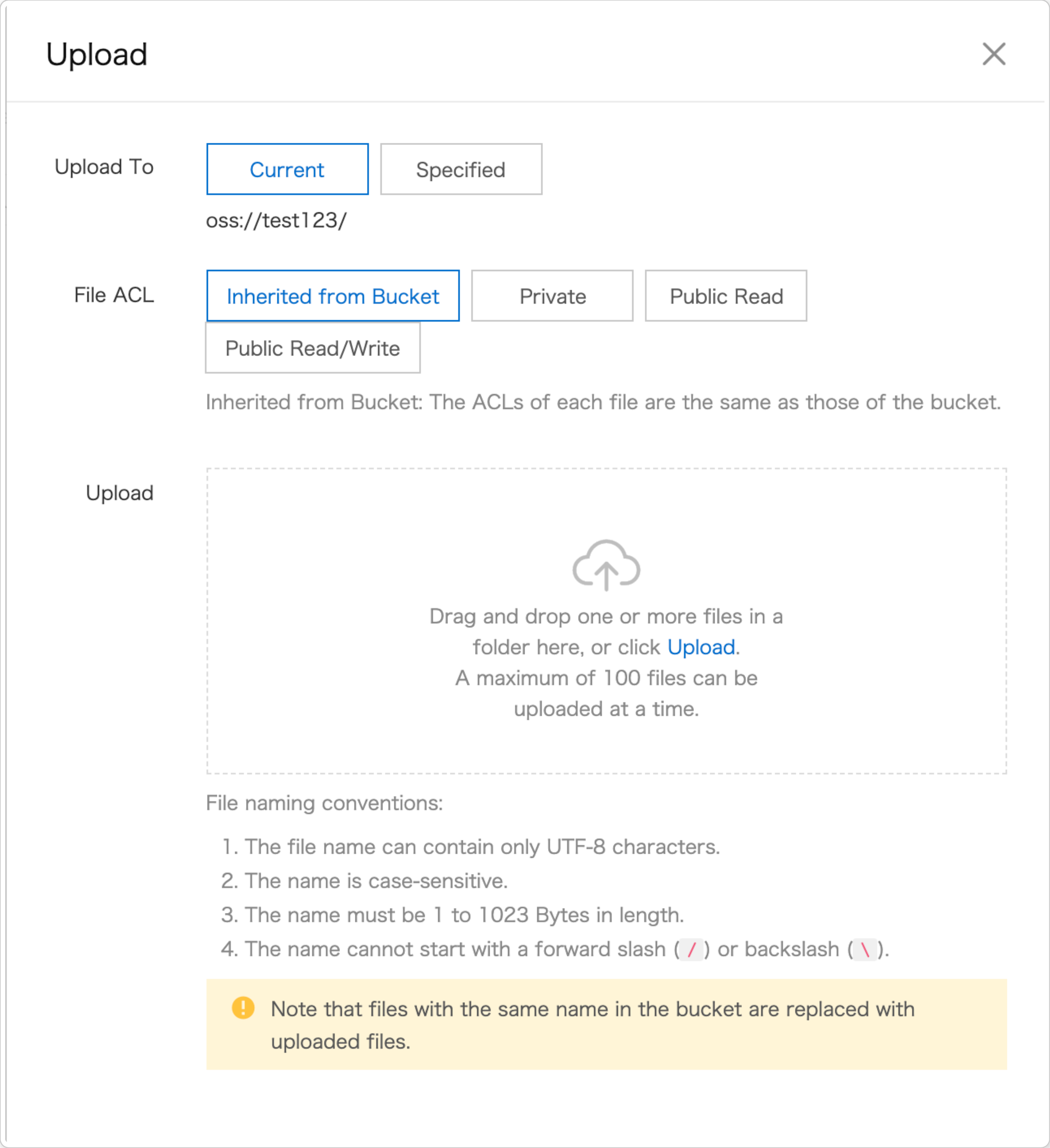
Access Control List (ACL): Private

Private: Only the owner or authorized users of this bucket can read

3. Fill out the **Create Bucket** fields as follows:
    - In the **Bucket Name** field, enter a name for your bucket. Remember the name you enter. You'll need it to connect your bucket to your Fastly service.
    - From the **Region** menu, select a location to store your content. Most customers select a region close to the POP they specify for shielding.
    - From the **Storage Class** options, select **Standard**.
    - From the **Access Control List (ACL)** options, select **Public Read**.
    - *(Optional)* Select other options, such as **Server-side Encryption** and **Scheduled Backup**.
4. Click **OK**.

Uploading files to your bucket

Once you've created your bucket, select it and then navigate to the Files tab to add files to it by clicking **Upload**.


Upload
Upload To
Current
Specified

oss://test123/

File ACL
Inherited from Bucket
Private
Public Read
Public Read/Write

Inherited from Bucket: The ACLs of each file are the same as those of the bucket.

Upload

Drag and drop one or more files in a folder here, or click Upload. A maximum of 100 files can be uploaded at a time.

File naming conventions:

1. The file name can contain only UTF-8 characters.
2. The name is case-sensitive.
3. The name must be 1 to 1023 Bytes in length.
4. The name cannot start with a forward slash (`/`) or backslash (`\`).

Note that files with the same name in the bucket are replaced with uploaded files.

You can make the files externally accessible by selecting the **Public Read** option for the bucket or you can use the **Inherited from Bucket** option next to each of the files.

Setting up Fastly to use OSS as an origin

To add your OSS bucket as an origin, follow the instructions for working with hosts. You'll add specific details about your origin server.

1. On the **Origins** page, click **Create Host** and enter the appropriate address for your Host using the format `<BUCKET>.<REGION>.aliyuncs.com`. For example, if your bucket name is `test123` and your region is Beijing (e.g., `oss-cn-beijing`) your hostname would be `test123.oss-cn-beijing.aliyuncs.com`. You can also find the hostname on the Bucket Overview page in the **Bucket Domain Name** area.
2. Click on the newly created Host to edit it.
3. In the **Name** field, enter a descriptive name for your service (e.g., `Alibaba Object Storage`).
4. If the **Address** field doesn't contain the `<BUCKET>.<REGION>.aliyuncs.com` hostname you provided in the first step, enter it now.
5. Fill out the **Transport Layer Security (TLS)** area fields as follows:
    - Leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
    - Leave the **Verify certificate?** default set to **Yes**.
    - Set the **Certificate hostname** field to the same address that appears in the Address field (e.g., `test123.oss-cn-beijing.aliyuncs.com`).
    - In the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname**. The hostname address you entered during Host creation appears.
6. From the **Shielding** menu below the TLS area, select a Fastly POP near the Alibaba region from the list of shielding locations.
7. In the **Override host** field, enter an appropriate address for your Host (e.g., `test123.oss-cn-beijing.aliyuncs.com`). You entered this information during Host creation.

Review our caveats of shielding and select a shield POP accordingly.

Using OSS with private objects

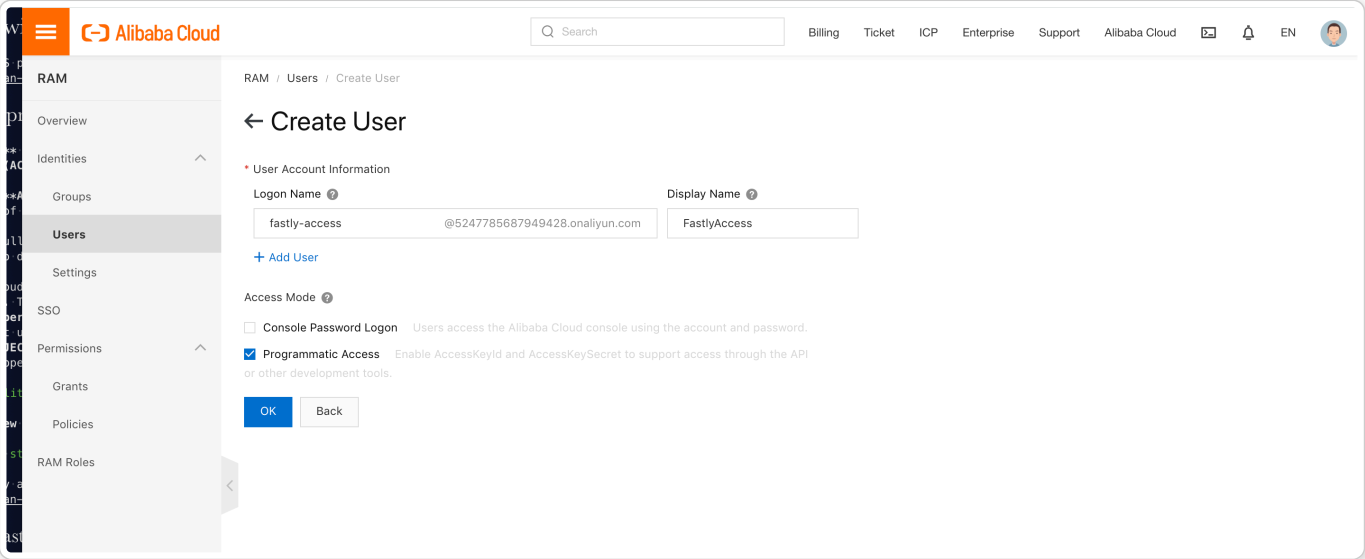
To use Fastly with OSS private objects, be sure you've already made your OSS data available to Fastly by pointing to the right OSS bucket, then follow the steps below.

Setting up a private bucket and sub user

Setting up a private bucket is the same as setting up a public bucket, except you select the **Private** option in the **Access Control List (ACL)** area of the OSS bucket settings.

You'll need an **AccessKey ID** and **Access Key Secret**. These can be linked to your account by clicking on your avatar in the top right corner of the Alibaba Cloud Console, selecting **Access Key**, and then creating a new key. Since this key has full access to the account, we recommend following Alibaba's procedure for creating a sub user. Follow the steps below.

1. Navigate to the Resource Access Management (RAM) page.
2. Click **Users**.
3. Click **Create User**.
4. Enter an appropriate **Logon Name** and **Display Name**.
5. Select the **Programmatic Access** checkbox to enable access through the Alibaba API.



← Create User

* User Account Information

Logon Name: fastly-access @5247785687949428.onaliyun.com

Display Name: FastlyAccess

+ Add User

Access Mode

☐ Console Password Logon Users access the Alibaba Cloud console using the account and password.

☑ Programmatic Access Enable AccessKeyId and AccessKeySecret to support access through the API or other development tools.

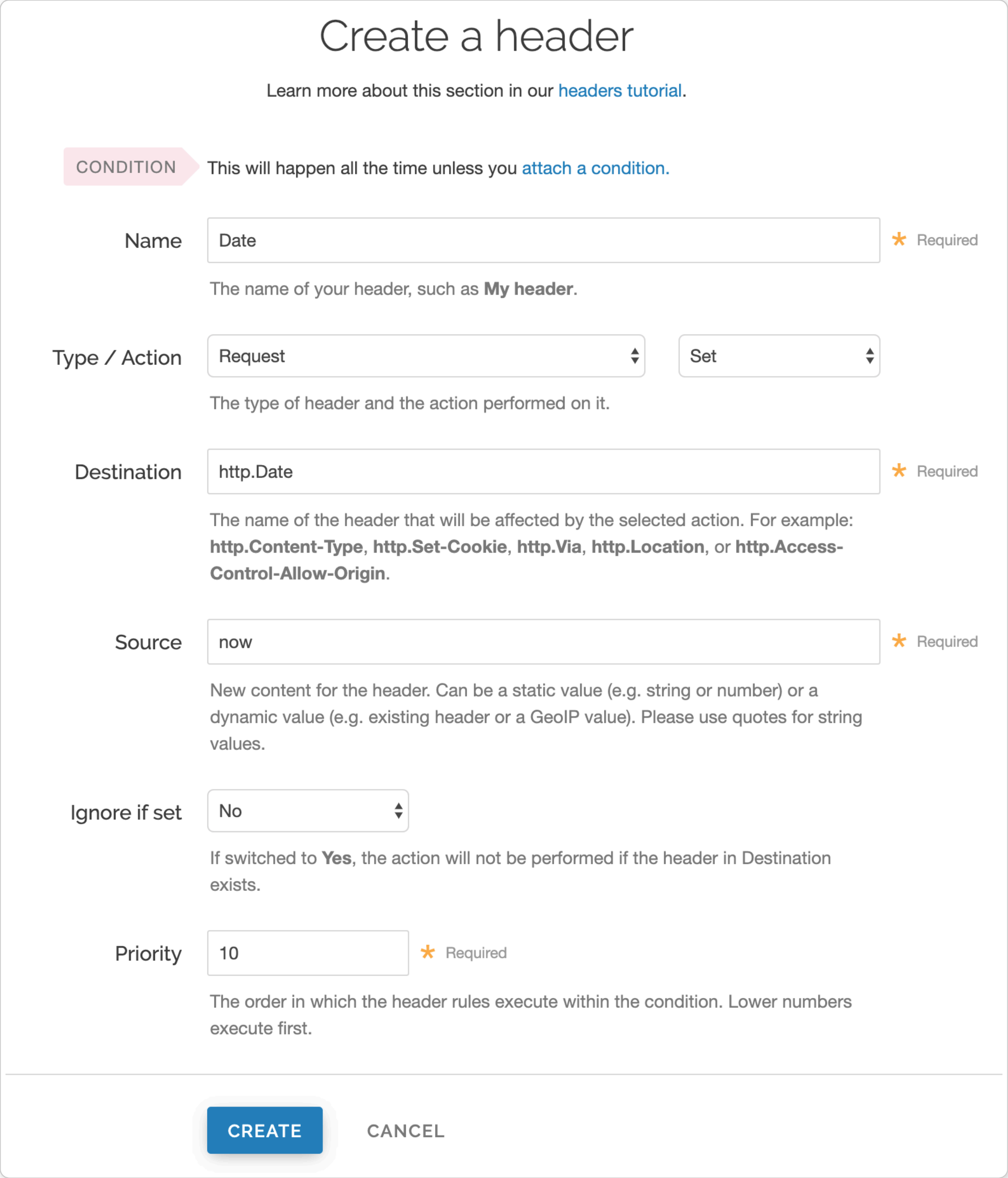
6. Click **OK**.
7. Copy the **AccessKeyId** and **AccessKeySecret**. You'll need these later when you're creating an Authorization header.
8. Go back to the bucket overview, click **Files** and then click **Authorize**. You should see a list of authorized users. If this is a new bucket it should be empty.
9. Click **Authorize**, filling out the fields as follows:
    - From the **Applied To** menu, select the **Whole Bucket** option. You can select **Specified Resources**, but this may lead to unexpected errors later if you don't update the permissions with new files.
    - From the **Accounts** menu, select **RAM Users** and then use the menu to select your newly created RAM user.
    - From the **Authorized Operation** menu, select **Read Only**.
    - You can leave **Condition** blank or customize it using **IP =**, Fastly's IP ranges, or setting **Access Method** to **HTTPS**.

Setting up Fastly to use OSS private content

To use OSS private content with Fastly, you'll need to create two headers: a Date header (required for authorization signature) and a Host header. You'll also need to add some authorization parameters.

Creating a Date header

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Date

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Date

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: now

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a new header** fields as follows:
   - In the **Name** field, enter `Date`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Date`.
   - In the **Source** field, enter `var.ali_expires`.

  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `19`.

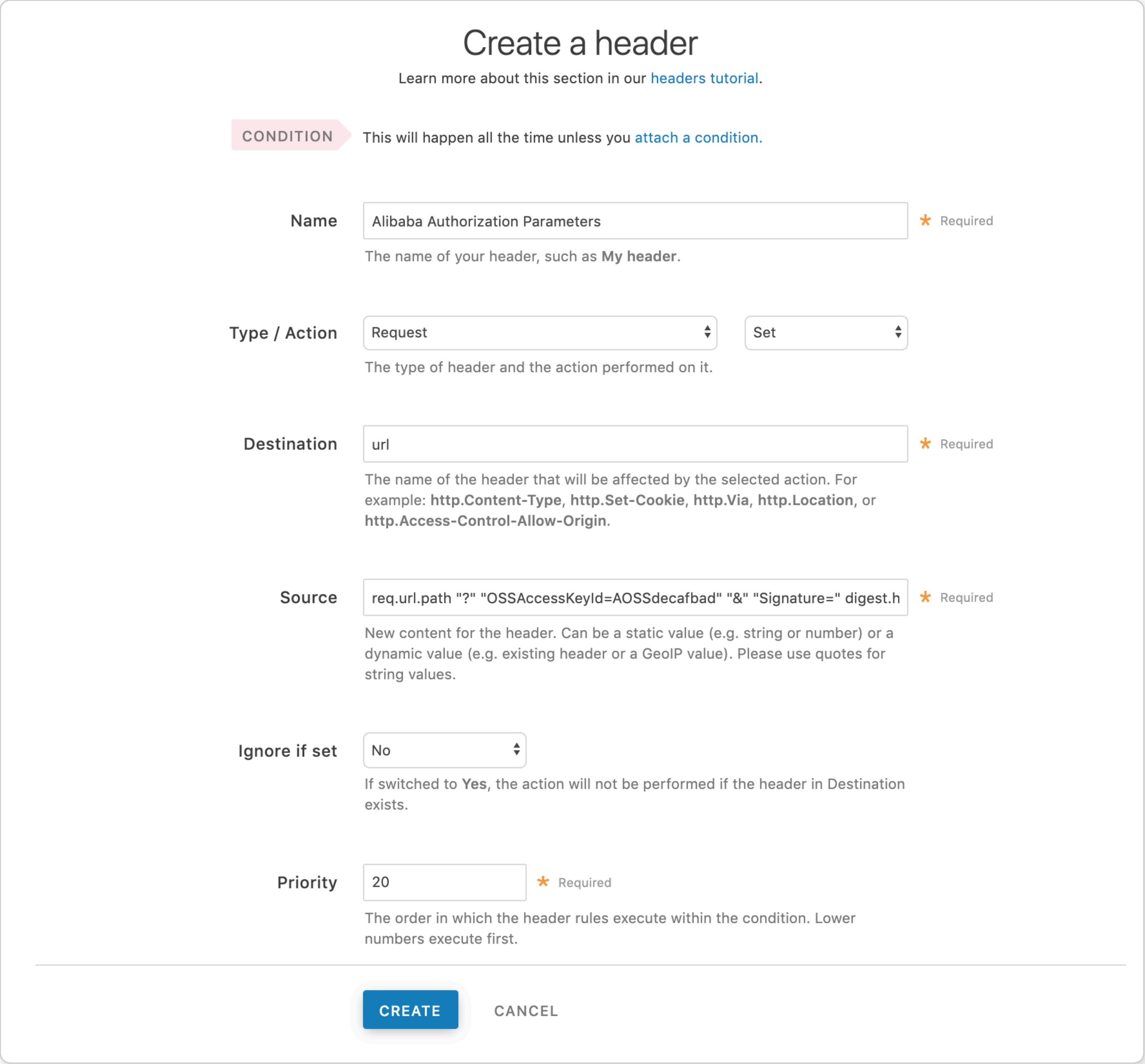
7. Click **Create**. A new Date header appears on the Content page. You will use this later within the signature of the Authorization header.

Creating a Host header

1. Click **Create header**.
2. Fill out the **Create a new header** fields as follows:
   - In the **Name** field, enter `Date`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Host`.
   - In the **Source** field, enter `"<your OSS domain>"`.
   - From the **Ignore if set** menu, select **No**.
   - In the **Priority** field, enter `19`.
3. Click **Create**. A new Host header appears on the Content page.

Creating the Authorization header

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Alibaba Authorization Parameters ★ Required
The name of your header, such as **My header**.

Type / Action: Request / Set
The type of header and the action performed on it.

Destination: url ★ Required
The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: req.url.path "?" "OSSAccessKeyId=AOSSdecafbad" "&" "Signature=" digest.h ★ Required
New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No
If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20 ★ Required
The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Authorization`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `url`.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `20`.
3. In the **Source** field, enter the Authorization header information using the following format:

```
req.url.path "?" "OSSAccessKeyId=<AccessKeyId>" "&" "Signature=" digest.hmac_sha1_ba 64
```

Replace `<AccessKeyId>`, `<AccessKeySecret>`, and `<OSS bucket name>` with the information you gathered before you began. For example:

```
req.url.path "?" "OSSAccessKeyId=AOSSdecafbad" "&" "Signature=" urlencode(digest.hma sh
```

4. Click **Create**. A new Authorization header appears on the Content page.

5. Click **Activate** to deploy your configuration changes.

Setting up Fastly to use OSS private content using VCL Snippets

You can also put the configuration in a VCL Snippet with a priority of `20`.

```
declare local var.ali_bucket STRING;
declare local var.ali_region STRING;
declare local var.ali_access_key_id STRING;
declare local var.ali_access_key_secret STRING;
declare local var.ali_expires INTEGER;
declare local var.ali_canon STRING;
declare local var.ali_sig STRING;_

set var.ali_bucket = "test123";
set var.ali_region = "oss-cn-beijing";
set var.ali_access_key_id = "decafbad";
set var.ali_access_key_secret = "deadbeef";
set var.ali_expires  = std.atoi(now.sec);
set var.ali_expires += 60;


set req.http.Host = var.ali_bucket "." + var.ali_region + ".aliyuncs.com";
set req.http.Date = var.ali_expires;
set var.ali_canon = if(req.method == "HEAD", "GET", req.method) LF LF LF
                    req.http.Date LF "/" var.ali_bucket req.url.path;
set var.ali_sig   = digest.hmac_sha1_base64(var.alibaba_access_key_secret, var.ali_cano

set req.url       = req.url.path;
set req.url       = querystring.set(req.url, "OSSAccessKeyId", var.alibaba_access_key_i
set req.url       = querystring.set(req.url, "Signature",  var.ali_sig);
set req.url       = querystring.set(req.url, "Expires",  var.ali_expires);
```

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Amazon S3

/en/guides/amazon-s3

Amazon S3 public and private buckets can be used as origins with Fastly.

Using Amazon S3 as an origin

To make your S3 data buckets available through Fastly, follow the steps below.

Creating a new service

Follow the instructions for creating a new service.

1. When you create the new domain and the new Host:
    - In the **Domain Name** field on the **Create a domain** page, enter the hostname you want to use as the URL (e.g., `cdn.example.com`).
    - In the **Hosts** field on the **Origins** page, enter the appropriate address for your Host using the format `<BUCKET>.s3.<REGION>.amazonaws.com`. Use the table in the Amazon S3 endpoints of the AWS general reference documentation as a guide. For example, if your bucket name is `fastlytestbucket` and your region is `us-east-2`, your hostname would be `fastlytestbucket.s3.us-east-2.amazonaws.com`.

    > ✓ TIP
    >
    > Most customers select a region close to the interconnect location they specify for shielding.

2. When you edit the Host details on the **Edit this host page**:
    - In the **Name** field, enter any descriptive name for your service if you haven't already done so.
    - In the **Address** field, ensure you've entered the appropriate address for your region (e.g., `fastlytestbucket.s3.us-east-2.amazonaws.com`). You entered this information during Host creation.
3. When you edit the Transport Layer Security (TLS) area information for your Host:
    - Leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.

        > ✓ TIP
        >
        > If you're using Amazon S3 to host a static website, **Enable TLS?** should be set to **No** instead of relying on the default. Amazon doesn't support TLS connections to S3 buckets with the static website hosting feature enabled. You can still use one of Fastly's TLS service options to secure connections between Fastly and clients.

    - Under the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname**. The address you entered during Host creation appears.
    - In the **Certificate hostname** field, enter `fastlytestbucket.s3.us-east-2.amazonaws.com`.
4. In the **Override host** field, enter an appropriate address for your Host (e.g., `fastlytestbucket.s3.us-east-2.amazonaws.com`). You entered this information during Host creation. If you're using an Amazon S3 private bucket leave this field blank.

Validating the DNS mapping

By default, we create a DNS mapping called **yourdomain.global.prod.fastly.net**. In the example above, it would be `cdn.example.com.global.prod.fastly.net`. Enter this URL in a browser to confirm it's working.

Creating a DNS alias

Create a DNS alias for the domain name you specified (e.g., CNAME `cdn.example.com` to `global-nossl.fastly.net`).

Verifying your results

Fastly will cache any content without an explicit `Cache-Control` header for 1 hour. You can verify whether you are sending any cache headers using curl. For example:

```
$ curl -I opscode-full-stack.s3.amazonaws.com

HTTP/1.1 200 OK
x-amz-id-2: ZpzRp7IWc6MJ8NtDEFGH12QBdk2CM1+RzVOngQbhMp2f2ZyalkFsZd4qPaLMkSlh
x-amz-request-id: ABV5032583242618
Date: Fri, 18 Mar 2012 17:15:38 GMT
Content-Type: application/xml
Transfer-Encoding: chunked
Server: AmazonS3
```

In this example, no Cache-Control headers are set so the default Time to Live (TTL) will be applied.

Enhanced cache control

If you need more control over how different types of assets are cached (e.g., JavaScript files, images), check out our documentation on cache freshness.

Using an Amazon S3 private bucket

To use an Amazon S3 private bucket with Fastly, you must implement version 4 of Amazon's header-based authentication. You can do this using custom VCL. Start by obtaining the following information from AWS:

Item	Description
Bucket name	The name of your AWS S3 bucket. When you download items from your bucket, this is the string listed in the URL path or hostname of each object.
Region	The AWS region code of the location where your bucket resides (e.g., `us-east-1`).
Access key	The AWS access key string for an IAM account that has at least read permission on the bucket.
Secret key	The AWS secret access key paired with the access key above.

Once you have this information, you can configure your Fastly service to authenticate against your S3 bucket using header authentication by calculating the appropriate header value in VCL.

IMPORTANT

Consider leaving the **Override host** field for the origin blank in your service settings. This setting will override the Host header from the snippets shown here and may invalidate the signature that authenticates the information being sent.

Start by creating a regular VCL snippet. Give it a meaningful name, such as `AWS protected origin`. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code (be sure to change specific values as noted to ones relevant to your own AWS bucket):

```
declare local var.awsAccessKey STRING;
declare local var.awsSecretKey STRING;
declare local var.awsS3Bucket STRING;
declare local var.awsRegion STRING;
declare local var.canonicalHeaders STRING;
declare local var.signedHeaders STRING;
declare local var.canonicalRequest STRING;
declare local var.canonicalQuery STRING;
declare local var.stringToSign STRING;
declare local var.dateStamp STRING;
declare local var.signature STRING;
declare local var.scope STRING;

set var.awsAccessKey = "YOUR_AWS_ACCESS_KEY";   # Change this value to your own data
set var.awsSecretKey = "YOUR_AWS_SECRET_KEY";   # Change this value to your own data
set var.awsS3Bucket = "YOUR_AWS_BUCKET_NAME";   # Change this value to your own data
set var.awsRegion = "YOUR_AWS_BUCKET_REGION";   # Change this value to your own data

if (req.method == "GET" && !req.backend.is_shield) {

  set bereq.http.x-amz-content-sha256 = digest.hash_sha256("");
  set bereq.http.x-amz-date = strftime({"%Y%m%dT%H%M%SZ"}, now);
  set bereq.http.host = var.awsS3Bucket ".s3." var.awsRegion ".amazonaws.com";
  set bereq.url = querystring.remove(bereq.url);
  set bereq.url = regsuball(urlencode(urldecode(bereq.url.path)), {"%2F"}, "/");
  set var.dateStamp = strftime({"%Y%m%d"}, now);
  set var.canonicalHeaders = ""
    "host:" bereq.http.host LF
    "x-amz-content-sha256:" bereq.http.x-amz-content-sha256 LF
    "x-amz-date:" bereq.http.x-amz-date LF
  ;
  set var.canonicalQuery = "";
  set var.signedHeaders = "host;x-amz-content-sha256;x-amz-date";
  set var.canonicalRequest = ""
    "GET" LF
    bereq.url.path LF
    var.canonicalQuery LF
    var.canonicalHeaders LF
    var.signedHeaders LF
    digest.hash_sha256("")
  ;

  set var.scope = var.dateStamp "/" var.awsRegion "/s3/aws4_request";

  set var.stringToSign = ""
    "AWS4-HMAC-SHA256" LF
    bereq.http.x-amz-date LF
    var.scope LF
    regsub(digest.hash_sha256(var.canonicalRequest),"^0x", "")
  ;

  set var.signature = digest.awsv4_hmac(
    var.awsSecretKey,
    var.dateStamp,
    var.awsRegion,
    "s3",
    var.stringToSign
  );

  set bereq.http.Authorization = "AWS4-HMAC-SHA256 "
    "Credential=" var.awsAccessKey "/" var.scope ", "
    "SignedHeaders=" var.signedHeaders ", "
    "Signature=" + regsub(var.signature,"^0x", "")
  ;
  unset bereq.http.Accept;
  unset bereq.http.Accept-Language;
  unset bereq.http.User-Agent;
  unset bereq.http.Fastly-Client-IP;
}
```

You may also remove the headers that AWS adds to the response. Do this by creating another VCL snippet. Give it a meaningful name, such as `Strip AWS response headers`. When you create the snippet, select **within subroutine** to specify its placement and choose **fetch** as the subroutine type. Then, place the following code in the **VCL** field:

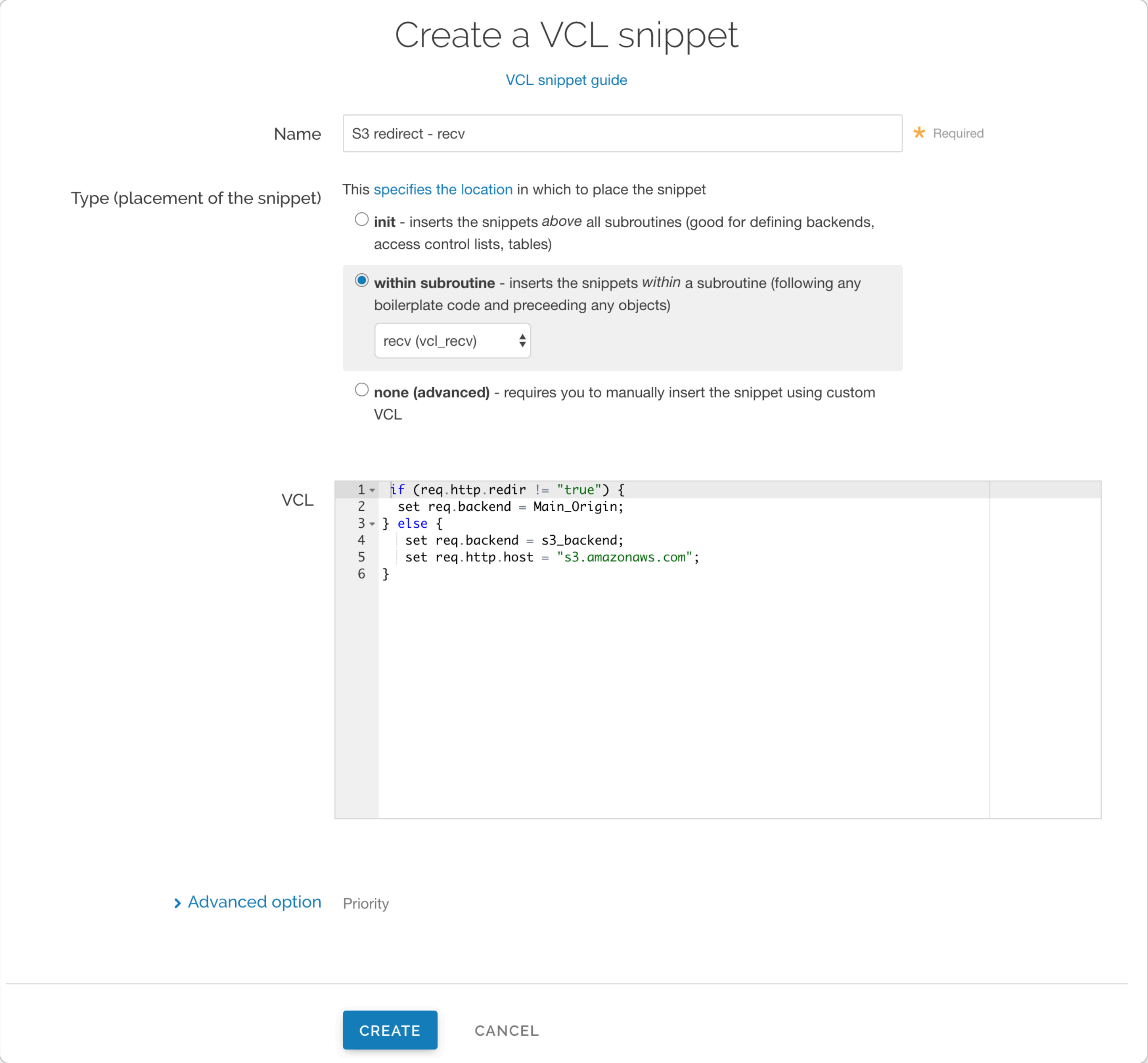
```
unset beresp.http.x-amz-id-2;
unset beresp.http.x-amz-request-id;
unset beresp.http.x-amz-delete-marker;
unset beresp.http.x-amz-version-id;
```

Following redirects to S3 objects and caching S3 responses

Using VCL Snippets, Fastly can follow redirects to S3 objects and cache the response.

To configure Fastly to follow redirects to S3 objects, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **VCL Snippets**.
5. Click **Create snippet**.



Create a VCL snippet

Name: S3 redirect - recv

Type (placement of the snippet)

This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

recv (vcl_recv)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL

```
if (req.http.redir != "true") {
  set req.backend = Main_Origin;
} else {
   set req.backend = s3_backend;
   set req.http.host = "s3.amazonaws.com";
}
```

6. In the **Name** field, enter an appropriate name (e.g., `S3 redirect - recv`).

7. From the **Type (placement of the snippet)** controls, select **within subroutine**.

8. From the **Select subroutine** menu, select **recv (`vcl_recv`)**.

9. In the **VCL** field, add the following condition:

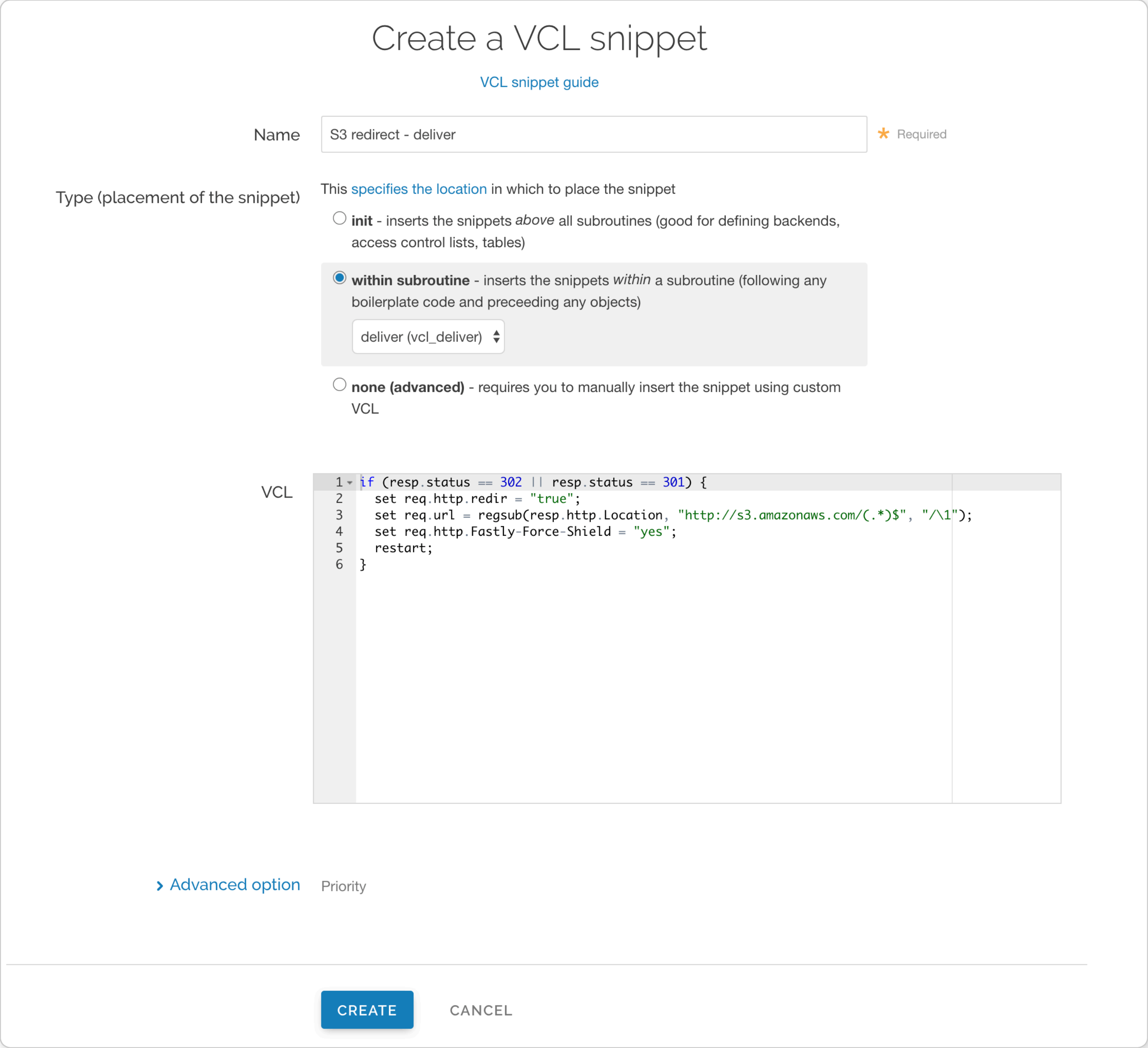
```
if (req.http.redir != "true") {
  set req.backend = Main_Origin;
} else {
  set req.backend = s3_backend;
  set req.http.host = "s3.amazonaws.com";
}
```

IMPORTANT

In the condition above, be sure to replace the `Main_Origin` and `s3_backend` placeholders with the actual names of your backends in the service to which you're applying these redirects. You can find the exact names by navigating to the **Deliver** page and clicking **Show VCL**, which appears just beneath your service name while viewing that service.

10. Click **Create** to create the snippet.

11. Click **Create snippet** again.



Create a VCL snippet

VCL snippet guide

Name: S3 redirect - deliver

* Required

Type (placement of the snippet)

This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceeding any objects)

deliver (vcl_deliver)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

VCL

```
if (resp.status == 302 || resp.status == 301) {
  set req.http.redir = "true";
  set req.url = regsub(resp.http.Location, "http://s3.amazonaws.com/(.*)$", "/\1");
  set req.http.Fastly-Force-Shield = "yes";
  restart;
}
```

12. In the **Name** field, enter an appropriate name (e.g., `S3 redirect - deliver`).

13. From the **Type (placement of the snippet)** controls, select **within subroutine**.

14. From the **Select subroutine** menu, select **deliver (**`vcl_deliver`**)**.

15. In the **VCL** field, add the following condition:

```
if (resp.status == 302 || resp.status == 301) {
    set req.http.redir = "true";
    set req.url = regsub(resp.http.Location, "http://s3.amazonaws.com/(.*)$", "/\1");
    set req.http.Fastly-Force-Shield = "yes";
    restart;
  }
```

16. Click **Create** to create the snippet.

17. Click **Activate** to deploy your configuration changes.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Backblaze B2 Cloud Storage

/en/guides/backblaze-b2-cloud-storage

Backblaze B2 Cloud Storage (B2) public and private buckets can be used as origins with Fastly.

TIP

Backblaze offers an integration discount that eliminates egress costs to Fastly when using Backblaze B2 Cloud Storage as an origin. In addition, Backblaze also offers a migration program designed to offset many of the data transfer costs associated with switching from another cloud provider to Backblaze. To ensure your migration has minimal downtime, contact support.

Before you begin

Before you begin the setup and configuration steps required to use B2 as an origin, keep in mind the following:

- **You must have a valid Backblaze account.** Before you can create a new bucket and upload files to it for Fastly to use, you must first create a Backblaze account at the Backblaze website.
- **Backblaze provides two ways to set up and configure B2.** B2 can be set up and configured using either the Backblaze web interface or the B2 command line tool. Either creation method works for *public* buckets. To use *private* buckets, however, you must use the B2 command line tool. For additional details, including instructions on how to install the command line tool, read Backblaze's B2 documentation.
- **Backblaze provides two APIs for integrating with Backblaze B2 Cloud Storage.** You can use the B2 Cloud Storage API or the S3 Compatible API to make your B2 data buckets available through Fastly. The S3 Compatible API allows existing S3 integrations and SDKs to integrate with B2. Buckets and their specific application keys created prior to May 4th, 2020, however, cannot be used with the S3 Compatible API. For more information, read Backblaze's article on Getting Started with the S3 Compatible API.

Using Backblaze B2 as an origin

To use B2 as an origin, follow the steps below.

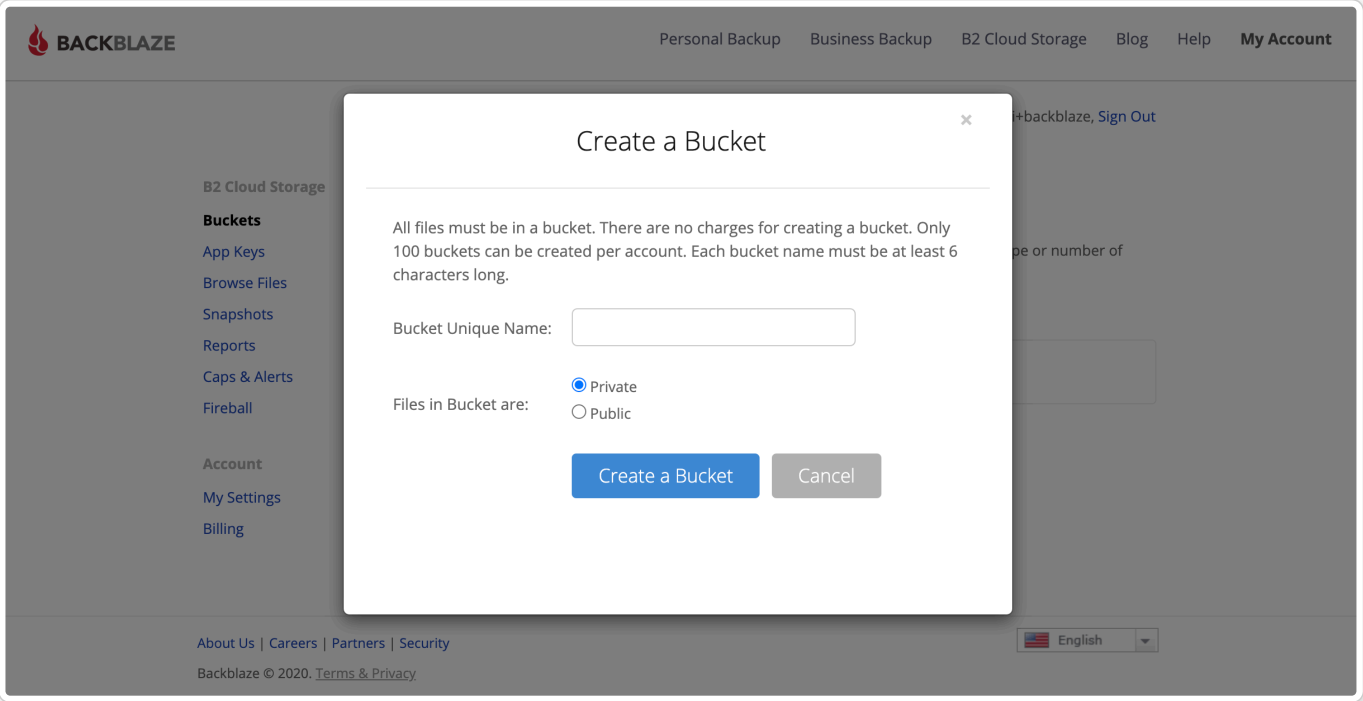
Creating a new bucket

Data in B2 is stored in buckets. Follow these steps to create a new bucket via the B2 web interface.

TIP

The Backblaze Guide provides details on how to create a bucket using the command line tool.

1. Log in to your Backblaze account. Your Backblaze account settings page appears.
2. Click **Buckets**.
3. Click **Create a Bucket**.



Create a Bucket

All files must be in a bucket. There are no charges for creating a bucket. Only 100 buckets can be created per account. Each bucket name must be at least 6 characters long.

Bucket Unique Name:

Files in Bucket are: ● Private ○ Public

BACKBLAZE
Personal Backup
Business Backup
B2 Cloud Storage
Blog
Help
My Account
+backblaze, Sign Out
B2 Cloud Storage
Buckets
App Keys
Browse Files
Snapshots
Reports
Caps & Alerts
Fireball
Account
My Settings
Billing
pe or number of
×
Create a Bucket
Cancel
About Us | Careers | Partners | Security
Backblaze © 2020. Terms & Privacy
English

4. In the **Bucket Unique Name** field, enter a unique bucket name. Each bucket name must be at least 6 alphanumeric characters and can only use hyphens (`-`) as separators, not spaces.
5. Click **Create a Bucket**. The new bucket appears in the list of buckets on the B2 Cloud Storage Buckets page.
6. Upload a file to the new bucket you just created.

NOTE

Buckets created prior to May 4th, 2020 cannot be used with the S3 Compatible API. If you do not have any S3 Compatible buckets, Backblaze recommends creating a new bucket.

Uploading files to a new bucket

Once you've created a new bucket in which to store your data, follow these steps to upload files to it via the B2 web interface.

TIP

The Backblaze Guide provides details on how to upload files using the command line tool.

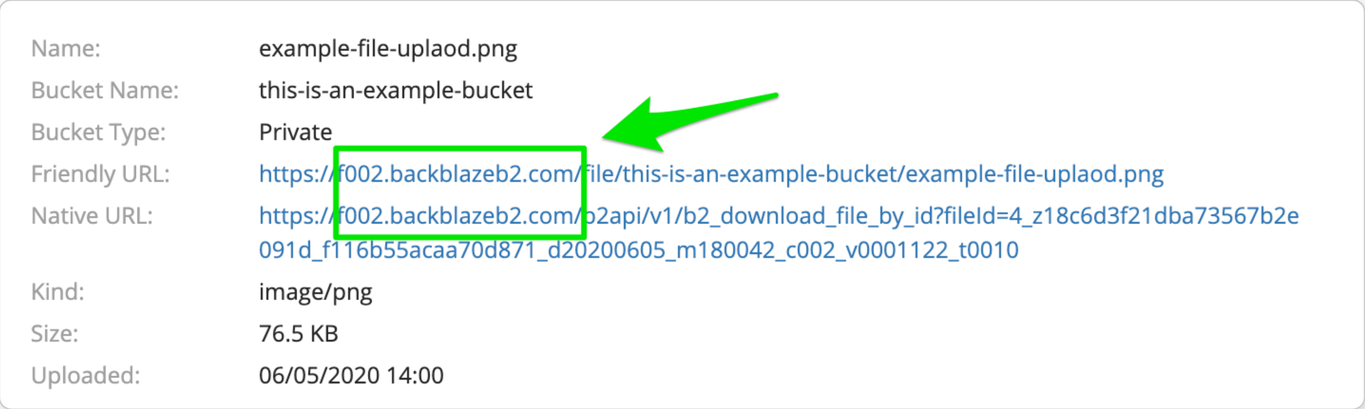
1. Click **Buckets** in the B2 web interface. The B2 Cloud Storage Buckets page appears.
2. Find the bucket details for the bucket you just created.
3. Click **Upload/Download**.
4. Click **Upload**.
5. Either drag and drop any file into the window or click to use the file selection tools to find a file to be uploaded. The name and type of file at this stage doesn't matter. Any file will work. Once uploaded, the name of the file appears in the list of files for the bucket.
6. Find your bucket's assigned hostname so you can set up a Fastly service that interacts with B2.

Finding your bucket's assigned hostname

To set up a Fastly service that interacts with your B2, you will need to know the hostname Backblaze assigned to the bucket you created and uploaded files to.

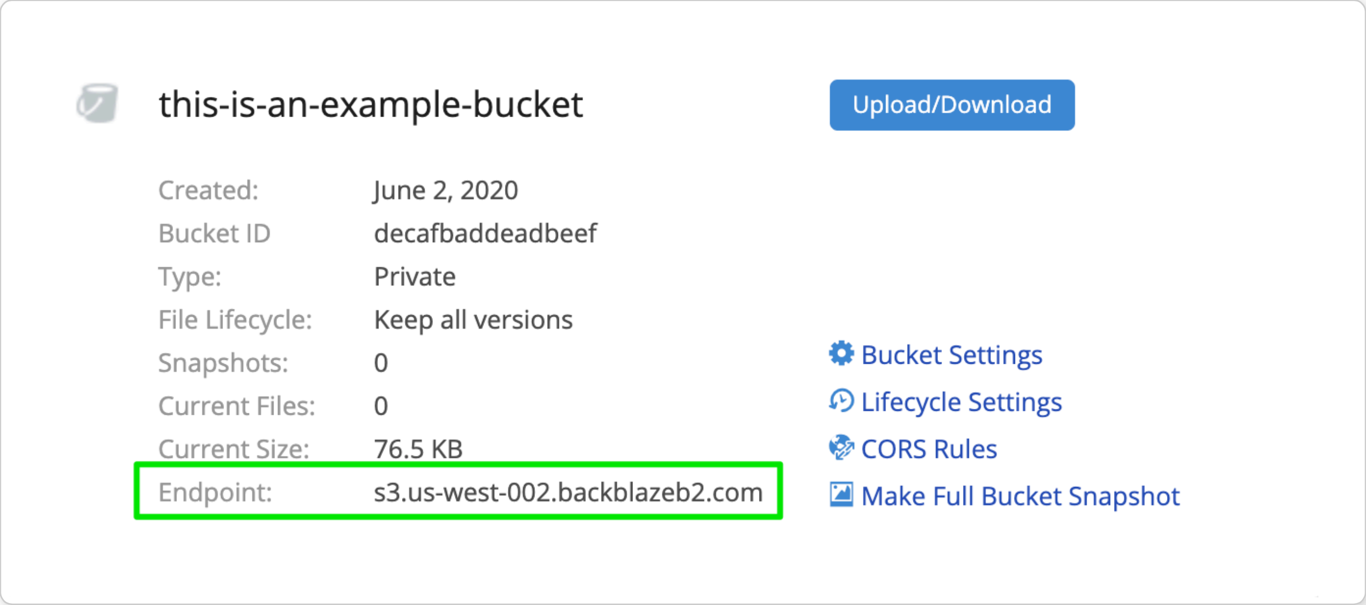
Find your hostname in one of the following ways:

- **Via the B2 web interface when you're using the standard B2 Cloud Storage API.** Click the name of the file you just uploaded and examine the **Friendly URL** and **Native URL** fields in the Details window that appears. The hostname is the text after the `https://` designator in each line that matches exactly.



Name:	example-file-uplaod.png
Bucket Name:	this-is-an-example-bucket
Bucket Type:	Private
Friendly URL:	https://f002.backblazeb2.com/file/this-is-an-example-bucket/example-file-uplaod.png
Native URL:	https://f002.backblazeb2.com/b2api/v1/b2_download_file_by_id?fileId=4_z18c6d3f21dba73567b2e091d_f116b55acaa70d871_d20200605_m180042_c002_v0001122_t0010
Kind:	image/png
Size:	76.5 KB
Uploaded:	06/05/2020 14:00

- **Via the command line and the B2 Cloud Storage API.** Run the `b2 get-account-info` command on the command line and use the hostname from the `downloadUrl` attribute.
- **Via the B2 web interface when you're using the S3 Compatible API.** Click **Buckets** and find the bucket details for the bucket you just created. The hostname is the text in the Endpoint field.



this-is-an-example-bucket

Created: June 2, 2020
Bucket ID decafbaddeadbeef
Type: Private
File Lifecycle: Keep all versions
Snapshots: 0
Current Files: 0
Current Size: 76.5 KB
Endpoint: s3.us-west-002.backblazeb2.com

Creating a Backblaze application key for private buckets

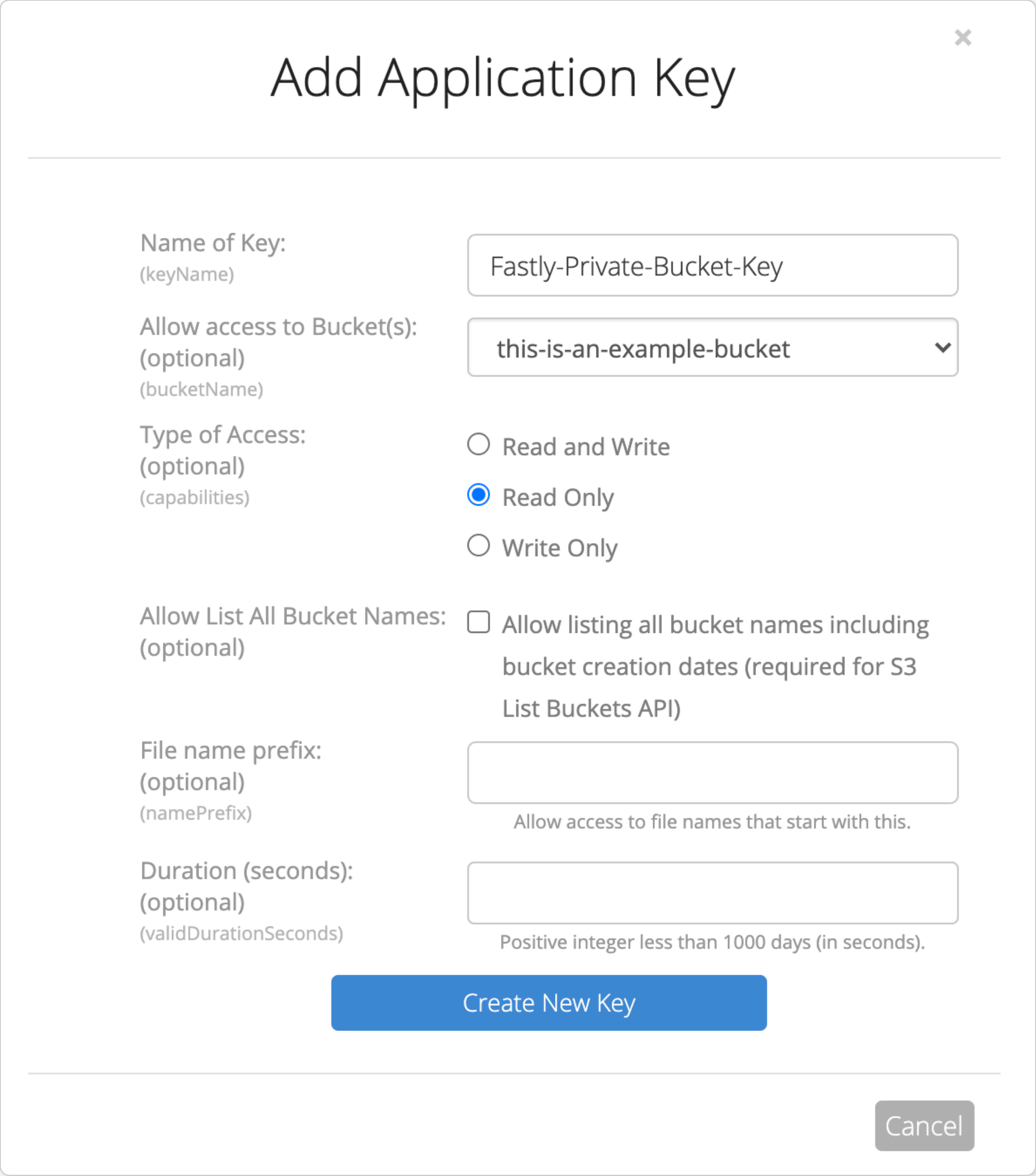
Your Backblaze master application key controls access to all buckets and files on your Backblaze account. If you plan to use a Backblaze B2 *private* bucket with Fastly, you should create an application key specific to the bucket.

> NOTE
>
> The Backblaze documentation provides more information about application keys. When creating application keys for your private bucket, we recommend using the least amount of privileges possible. You can optionally set the key to expire after a certain number of seconds (up to a maximum of 1000 days or 86,400,000 seconds). If you choose an expiration, however, you'll need to periodically create a new application key and then update your Fastly configuration accordingly each time.

Via the web interface

To create an application key via the B2 web interface:

1. Click **App Keys**.
2. Click **Add a New Application Key**.



Add Application Key

Name of Key: (keyName) Fastly-Private-Bucket-Key

Allow access to Bucket(s): (optional) (bucketName) this-is-an-example-bucket

Type of Access: (optional) (capabilities)
○ Read and Write
● Read Only
○ Write Only

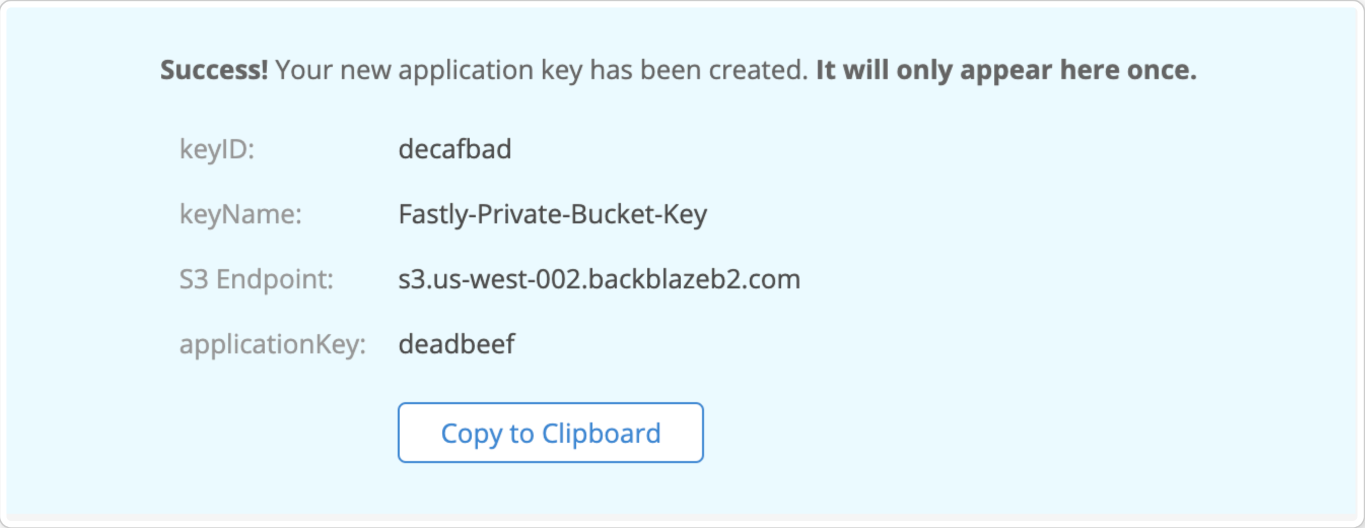
Allow List All Bucket Names: (optional) ☐ Allow listing all bucket names including bucket creation dates (required for S3 List Buckets API)

File name prefix: (optional) (namePrefix)
Allow access to file names that start with this.

Duration (seconds): (optional) (validDurationSeconds)
Positive integer less than 1000 days (in seconds).

3. Fill out the fields of the **Add Application Key** controls as follows:
    - In the **Name of Key** field, enter the name of your private bucket key. Key names are alphanumeric and can only use hyphens (`-`) as separators, not spaces.
    - From the **Allow access to Bucket(s)** menu, select the name of your private bucket.
    - From the **Type of Access** controls, select **Read Only**.
    - Leave the remaining optional controls and fields blank.

4. Click **Create New Key**.



Success! Your new application key has been created. **It will only appear here once.**

keyID: decafbad

keyName: Fastly-Private-Bucket-Key

S3 Endpoint: s3.us-west-002.backblazeb2.com

applicationKey: deadbeef

5. Immediately note the **keyID** and the **applicationKey** from the success message. You'll use this information when you implement header-based authentication with private objects.

Via the command line

To create an application key from the command line, run the `create-key` command as follows:

```
$ b2 create-key --bucket <bucketName> <keyName> shareFiles,listBuckets
```

where `<bucketName> <keyName>` represents the name of the bucket and key you created. For example:

```
$ b2 create-key --bucket this-is-an-example-bucket Fastly-Private-Bucket-Key shareFiles,
```

The **keyID** and the **applicationKey** are the two values returned.

NOTE

Application keys created prior to May 4th, 2020 cannot be used with the S3 Compatible API.

Creating a new service

To create a new Fastly service, you must first create a new domain and then create a new host and edit it to accept traffic for B2. Instructions to do this appear in our guide to creating a new service. While completing these instructions, keep the following in mind:

1. When you create the new host, enter the B2 bucket's hostname in the **Hosts** field on the **Origins** page.
2. When you edit the host details on the **Edit this host page**, confirm the Transport Layer Security (TLS) area information for your host. Specifically, make sure you:
   - secure the connection between Fastly and your origin.
   - enter your bucket's hostname in the **Certificate hostname** field.
   - select the checkbox to match the SNI hostname to the Certificate hostname (it appears under the SNI hostname field).

3. *(Optional)* Also when you edit the host, enable shielding by choosing the appropriate shielding location from the **Shielding** menu. When using B2 Cloud Storage, this means you must choose a shielding location closest to the most appropriate Backblaze data center. For the data centers closest to:
    - Sacramento, California (in the US West region), choose **San Jose (SJC)** from the **Shielding** menu.
    - Phoenix, Arizona (in the US West region), choose **Palo Alto (PAO)** from the **Shielding** menu.
    - Amsterdam, Netherlands (in the EU central region), choose **Amsterdam (AMS)** from the **Shielding** menu.
4. Decide whether or not you should specify an override host:
    - If you're using the S3 Compatible API, skip this step and don't specify an override host.
    - If you're not using the S3 Compatible API, in the **Override host** field, enter an appropriate address for your host (e.g., `s3-uswest-000.backblazeb2.com` or `f000.backblazeb2.com`).

Using the S3 Compatible API

Using the S3 Compatible API with public objects

To use the S3 Compatible API with public objects, you will need to make sure the `Host` header contains the name of your B2 Bucket. There are two ways to do this, both of which require you to get your region name which will be the 2nd part of your S3 Endpoint. So if your S3 Endpoint is `s3.us-west-000.backblazeb2.com`, this means your region will be `us-west-000`.

1. In the **Origin** you created set the **Override host** field to `<bucket>.s3.<region>.backblazeb2.com` (e.g., `testing.s3.uswest-000.backblazeb2.com`).
2. Create a VCL Snippet. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code. Be sure to change specific values as noted to ones relevant to your own B2 bucket - in this case `var.b2Bucket` would be `"testing"` and `var.b2Region` would be `"uswest-000"`.

```
declare local var.b2Bucket STRING;
declare local var.b2Region STRING;
set var.b2Bucket = "YOUR_B2_BUCKET_NAME";   # Change this value to your own data
set var.b2Region = "YOUR_B2_BUCKET_REGION"; # Change this value to your own data

if (req.method == "GET" && !req.backend.is_shield) {
  set bereq.http.host = var.b2Bucket ".s3." var.b2Region ".backblazeb2.com";
}
```

Using the S3 Compatible API with private objects

To use a Backblaze B2 private bucket with Fastly, you must implement version 4 of Amazon's header-based authentication. You can do this using custom VCL.

Start by obtaining the following information from Backblaze (see Creating a Backblaze application key for private buckets):

Item	Description
Bucket name	The name of your Backblaze B2 bucket. When you download items from your bucket, this is the string listed in the URL path or hostname of each object.

Item	Description
Region	The Backblaze region code of the location where your bucket resides (e.g., `uswest-000`).
Access key	The Backblaze keyID for the App Key that has at least read permission on the bucket.
Secret key	The Backblaze applicationKey paired with the access key above.

Once you have this information, you can configure your Fastly service to authenticate against your B2 bucket using header authentication by calculating the appropriate header value in VCL.

Start by creating a regular VCL snippet. Give it a meaningful name, such as `AWS protected origin`. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code (be sure to change specific values as noted to ones relevant to your own AWS bucket):

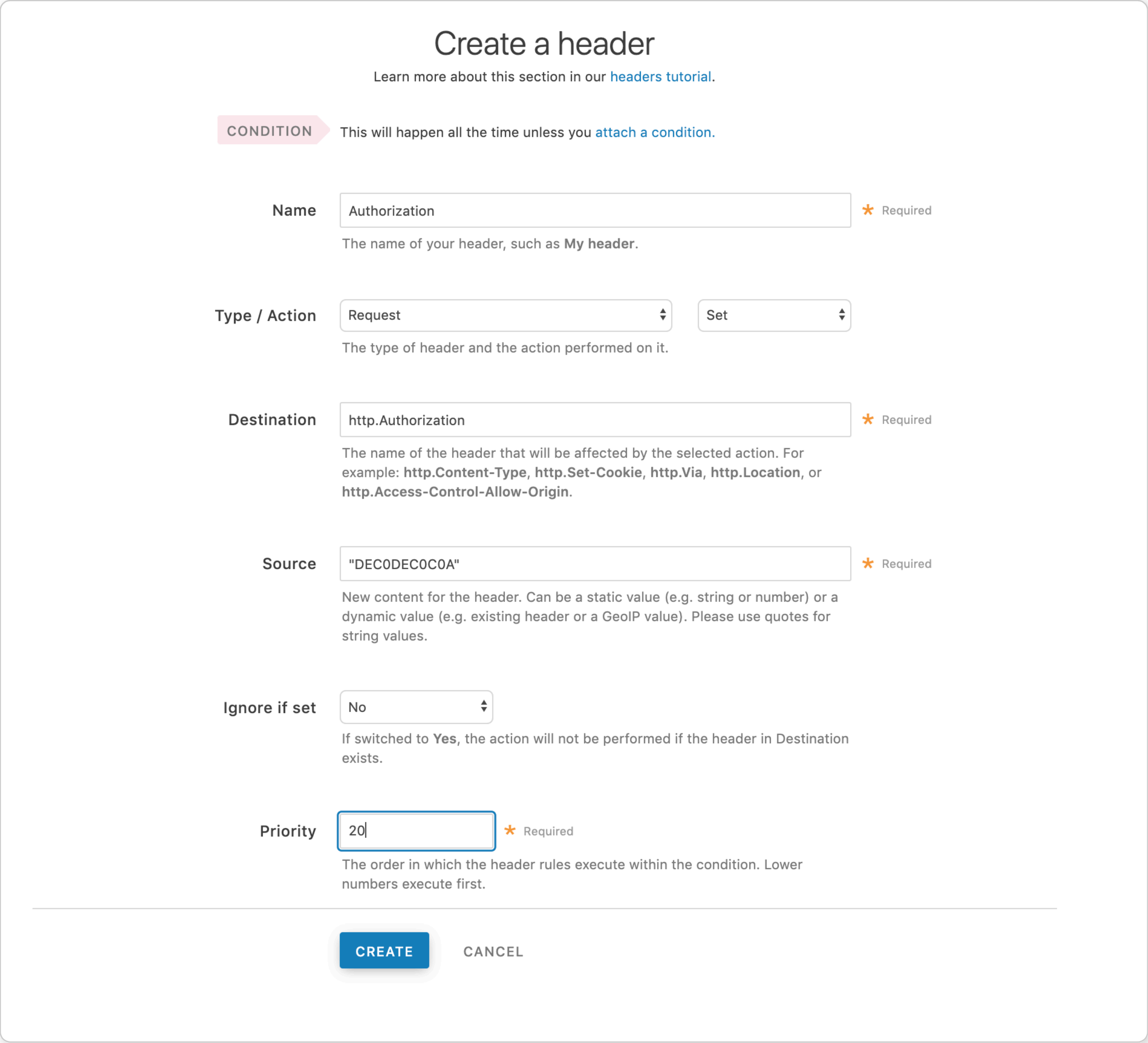
```
declare local var.b2AccessKey STRING;
declare local var.b2SecretKey STRING;
declare local var.b2Bucket STRING;
declare local var.b2Region STRING;
declare local var.canonicalHeaders STRING;
declare local var.signedHeaders STRING;
declare local var.canonicalRequest STRING;
declare local var.canonicalQuery STRING;
declare local var.stringToSign STRING;
declare local var.dateStamp STRING;
declare local var.signature STRING;
declare local var.scope STRING;

set var.b2AccessKey = "YOUR_B2_ACCESS_KEY";   # Change this value to your own data
set var.b2SecretKey = "YOUR_B2_SECRET_KEY";   # Change this value to your own data
set var.b2Bucket = "YOUR_B2_BUCKET_NAME";     # Change this value to your own data
set var.b2Region = "YOUR_B2_BUCKET_REGION";   # Change this value to your own data

if (req.method == "GET" && !req.backend.is_shield) {

  set bereq.http.x-amz-content-sha256 = digest.hash_sha256("");
  set bereq.http.x-amz-date = strftime({"%Y%m%dT%H%M%SZ"}, now);
  set bereq.http.host = var.b2Bucket ".s3." var.b2Region ".backblazeb2.com";
  set bereq.url = querystring.remove(bereq.url);
  set bereq.url = regsuball(urlencode(urldecode(bereq.url.path)), {"%2F"}, "/");
  set var.dateStamp = strftime({"%Y%m%d"}, now);
  set var.canonicalHeaders = ""
    "host:" bereq.http.host LF
    "x-amz-content-sha256:" bereq.http.x-amz-content-sha256 LF
    "x-amz-date:" bereq.http.x-amz-date LF
  ;
  set var.canonicalQuery = "";
  set var.signedHeaders = "host;x-amz-content-sha256;x-amz-date";
  set var.canonicalRequest = ""
    "GET" LF
    bereq.url.path LF
    var.canonicalQuery LF
    var.canonicalHeaders LF
    var.signedHeaders LF
    digest.hash_sha256("")
  ;

  set var.scope = var.dateStamp "/" var.b2Region "/s3/aws4_request";

  set var.stringToSign = ""
    "AWS4-HMAC-SHA256" LF
    bereq.http.x-amz-date LF
    var.scope LF
    regsub(digest.hash_sha256(var.canonicalRequest),"^0x", "")
  ;

  set var.signature = digest.awsv4_hmac(
    var.b2SecretKey,
    var.dateStamp,
    var.b2Region,
    "s3",
    var.stringToSign
  );

  set bereq.http.Authorization = "AWS4-HMAC-SHA256 "
    "Credential=" var.b2AccessKey "/" var.scope ", "
    "SignedHeaders=" var.signedHeaders ", "
    "Signature=" + regsub(var.signature,"^0x", "")
  ;
  unset bereq.http.Accept;
  unset bereq.http.Accept-Language;
  unset bereq.http.User-Agent;
  unset bereq.http.Fastly-Client-IP;
}
```

Using the B2 API

Public Objects

You'll need to make sure the URL contains your bucket name. You can do this using a **Header** object as follows:

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Authorization ★ Required
The name of your header, such as **My header**.

Type / Action: Request | Set
The type of header and the action performed on it.

Destination: http.Authorization ★ Required
The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "DEC0DEC0C0A" ★ Required
New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No
If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20 ★ Required
The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Rewrite B2 URL`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `url`.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `20`.
3. In the **Source** field, enter `"/file/<Bucket Name>" req.url`.
4. Click **Create**. A new Authorization header appears on the Content page.
5. Click **Activate** to deploy your configuration changes.

Alternatively create a VCL Snippet. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code. Be sure to change the variable to the name of your own B2 bucket.

```
declare local var.b2Bucket STRING;
set var.b2Bucket = "YOUR_B2_BUCKET_NAME";   # Change this value to your own data

if (req.method == "GET" && !req.backend.is_shield) {
  set bereq.url = "/file/" var.b2Bucket bereq.url;
}
```

Private Objects

1. To use a Backblaze B2 private bucket with Fastly, you must obtain an `Authorization Token`. This must be obtained via the command line.
2. You'll now need to authorize the command line tool with the application key you obtained.

   ```
   $ b2 authorize-account <keyID> <applicationKey>
   ```

3. You will now need to get an authorization token for the private bucket.

   ```
   $ b2 get-download-auth <bucket>
   ```

   e.g.,

   ```
   $ b2 get-download-auth testing
   ```

   This will create a token that is valid for 86400 seconds (i.e 1 day), the default. You can optionally change the expiration time from anywhere between 1s and 604,800 seconds (i.e 1 week).

   ```
   $ b2 get-download-auth --duration 604800 testing
   ```

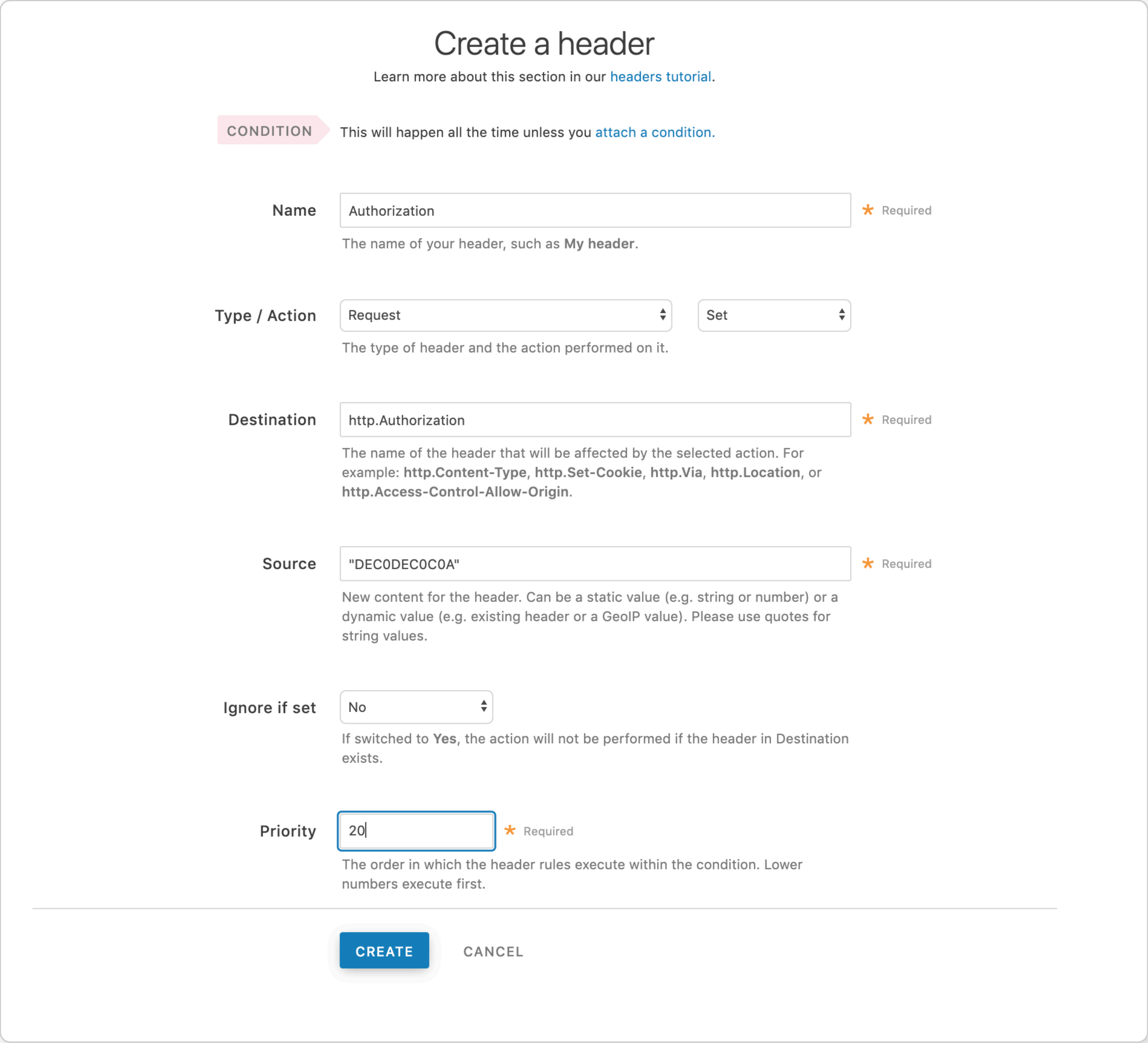
   Take note of the generated token.

> NOTE
>
> You will need to regenerate an authorization token and update your Fastly configuration before the end of the expiration time. A good way to do this would be through Fastly's versionless dictionaries.

Passing a generated token to Backblaze

There are two ways you can pass the generated token to Backblaze. The first is using an `Authorization` header. This is the recommended method.

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

This will happen all the time unless you attach a condition.

Name: Authorization ★ Required

The name of your header, such as **My header**.

Type / Action: Request | Set

The type of header and the action performed on it.

Destination: http.Authorization ★ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "DEC0DEC0C0A" ★ Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

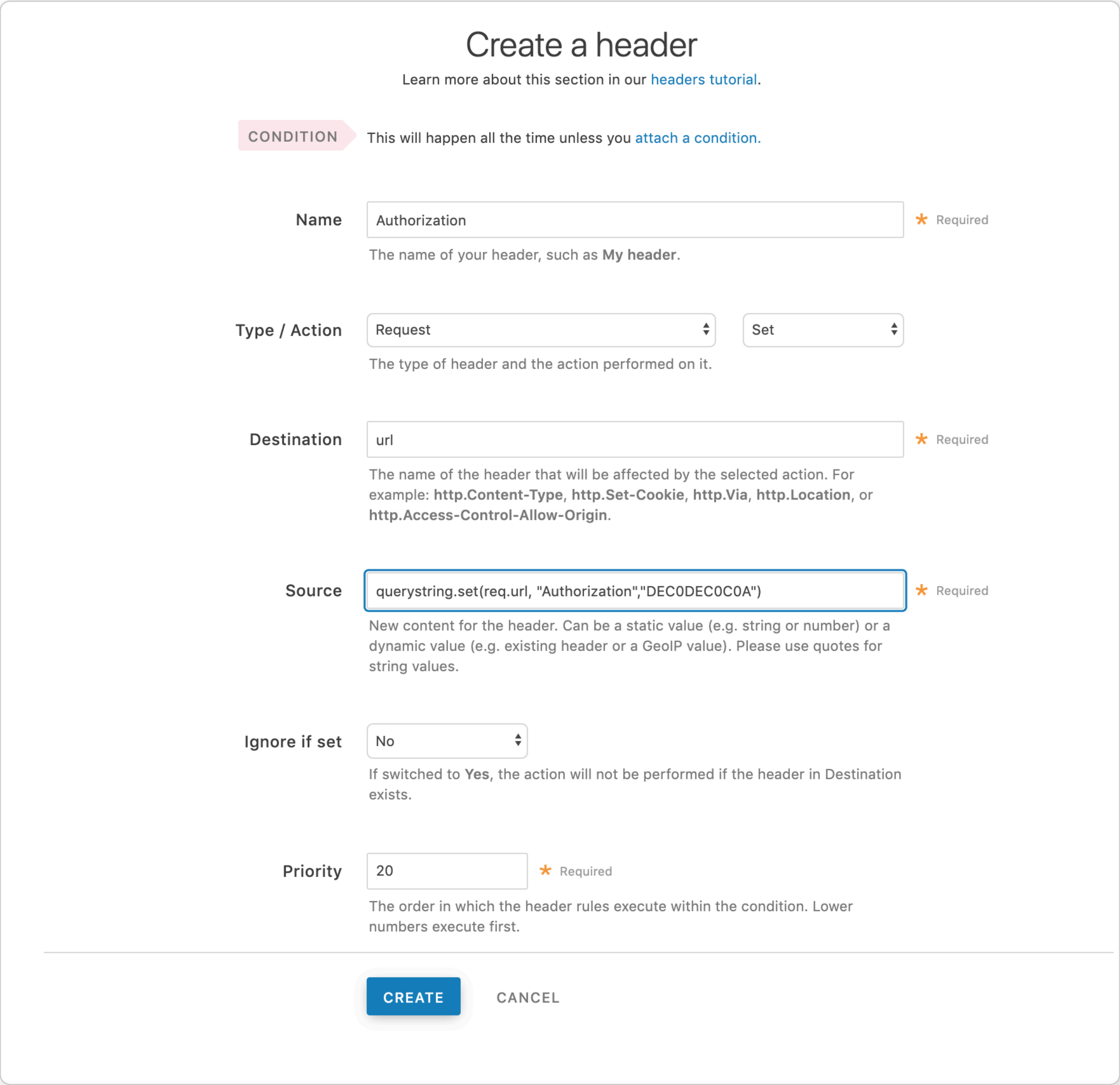
Priority: 20 ★ Required

The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Authorization`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Authorization`.
   - From the **Ignore if set** menu, select **No**.
   - In the **Priority** field, enter `20`.
3. In the **Source** field, enter the **Authorization Token** generated in the command line tool, surrounded by quotes. For example, if the token generated was `DEC0DEC0C0A`, then the **Source** field would be `"DEC0DEC0C0A"`.
4. Click **Create**. A new Authorization header appears on the Content page.
5. Click **Activate** to deploy your configuration changes.

Alternatively, the second way is to pass an `Authorization` query parameter.

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Authorization

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: url

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: querystring.set(req.url, "Authorization","DEC0DEC0C0A")

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20

The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Authorization`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `url`.
   - From the **Ignore if set** menu, select **No**.
   - In the **Priority** field, enter `20`.
3. In the **Source** field, enter the header authorization information using the following format:

```
querystring.set(req.url, "Authorization", "<Authorization Token>")
```

Using the previous example, that would be:

```
querystring.set(req.url, "Authorization", "DEC0DEC0C0A")
```

4. Click **Create**. A new Authorization header appears on the Content page.
5. Click **Activate** to deploy your configuration changes.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Data transfer with Backblaze B2

/en/guides/data-transfer-with-backblaze-b2

Fastly has partnered with Backblaze to provide an integration between Fastly and Backblaze B2 Cloud Storage services. Specifically, there are no egress costs from Backblaze to Fastly when you configure a Backblaze B2 service as your Fastly origin.

To offset many of the data transfer costs associated with switching from another cloud provider, Backblaze offers a migration program. Backblaze will cover migration fees for customers migrating over 50 TB of data from the US, Canada, and Europe regions of other Cloud Providers, and storing it with them for at least 12 months. For customers migrating less than 50 TB of data, Backblaze offers discounted transfer rates.

Read more about the migration program and review estimated cost saving calculations in their guide to Migrate to B2 Cloud Storage. To ensure your migration has minimal downtime, contact support.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Developer guide: Backends

/en/guides/developer-guide-backends

* * *

Developer guide: CI plugins

🔗 /en/guides/developer-guide-ci-plugins

* * *

Developer guide: CMS plugins

🔗 /en/guides/developer-guide-cms-plugins

* * *

Developer guide: Terraform

🔗 /en/guides/developer-guide-terraform

* * *

Developer guide: Third-party logging

🔗 /en/guides/developer-guide-third-party-logging

* * *

DigitalOcean Spaces

🔗 /en/guides/digitalocean-spaces

DigitalOcean Spaces public and private Spaces can be used as origins with Fastly.

Using DigitalOcean Spaces as an origin

To make your DigitalOcean Spaces available through Fastly, follow the steps below.

Creating a new service

Follow the instructions for creating a new service.

1. When you create the new domain and the new host:
    - In the **Domain Name** field on the **Create a domain** page, enter the hostname you want to use as the URL (e.g., `cdn.example.com`).
    - In the **Hosts** field on the **Origins** page, enter the appropriate address for your Host using the format `<SPACE>.<REGION>.digitaloceanspaces.com`. For example, if your space name is `test123` and your region is `nyc3`, your hostname would be `test123.nyc3.digitaloceanspaces.com`.
2. When you edit the host details on the **Edit this host page**:
    - In the **Name** field, enter any descriptive name for your service if you haven't already done so.
    - In the **Address** field, ensure you've entered the appropriate address for your Host (e.g., `test123.nyc3.digitaloceanspaces.com`). You entered this information during Host creation.
3. When you edit the Transport Layer Security (TLS) area information for your host:
    - Leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
    - In the **Certificate hostname** field, enter the same address that appears in the Address field (e.g., `test123.nyc3.digitaloceanspaces.com`).
    - Under the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname.** The address you entered during Host creation appears.
4. In the **Override host** field, enter an appropriate address for your Host (e.g., `test123.nyc3.digitaloceanspaces.com`). You entered this information during Host creation.

Testing your results

By default, we create DNS mapping called `yourdomain.global.prod.fastly.net`. In the example above, it would be `cdn.example.com.global.prod.fastly.net`. Create a DNS alias for the domain name you specified (e.g., CNAME `cdn.example.com` to `global-nossl.fastly.net`).

Fastly will cache any content without an explicit `Cache-Control` header for 1 hour. You can verify whether you are sending any cache headers using curl. For example:

```
$ curl -I opscode-full-stack.nyc3.digitaloceanspaces.com

HTTP/1.1 200 OK
x-amz-id-2: ZpzRp7IWc6MJ8NtDEFGH12QBdk2CM1+RzVOngQbhMp2f2ZyalkFsZd4qPaLMkSlh
x-amz-request-id: ABV5032583242618
Date: Fri, 18 Mar 2012 17:15:38 GMT
Content-Type: application/xml
Transfer-Encoding: chunked
```

In this example, no Cache-Control headers are set so default TTL will be applied.

Enhanced cache control

If you need more control over how different types of assets are cached (e.g., JavaScript files, images), refer to our cache freshness documentation.

Using private DigitalOcean Spaces

To use a private DigitalOcean Space with Fastly, follow the instructions below.

Before you begin

Be sure you've already made your Spaces data available to Fastly by pointing to the right Space and setting your origin to port 443. This needs to be done before authenticating.

Be sure you've got the access key, secret key, and Space name on hand. The DigitalOcean Spaces Authorization header takes the following form:

```
Authorization: AWS `AWSAccessKeyId`:`Signature`
```

From the DigitalOcean website you will need the following information:

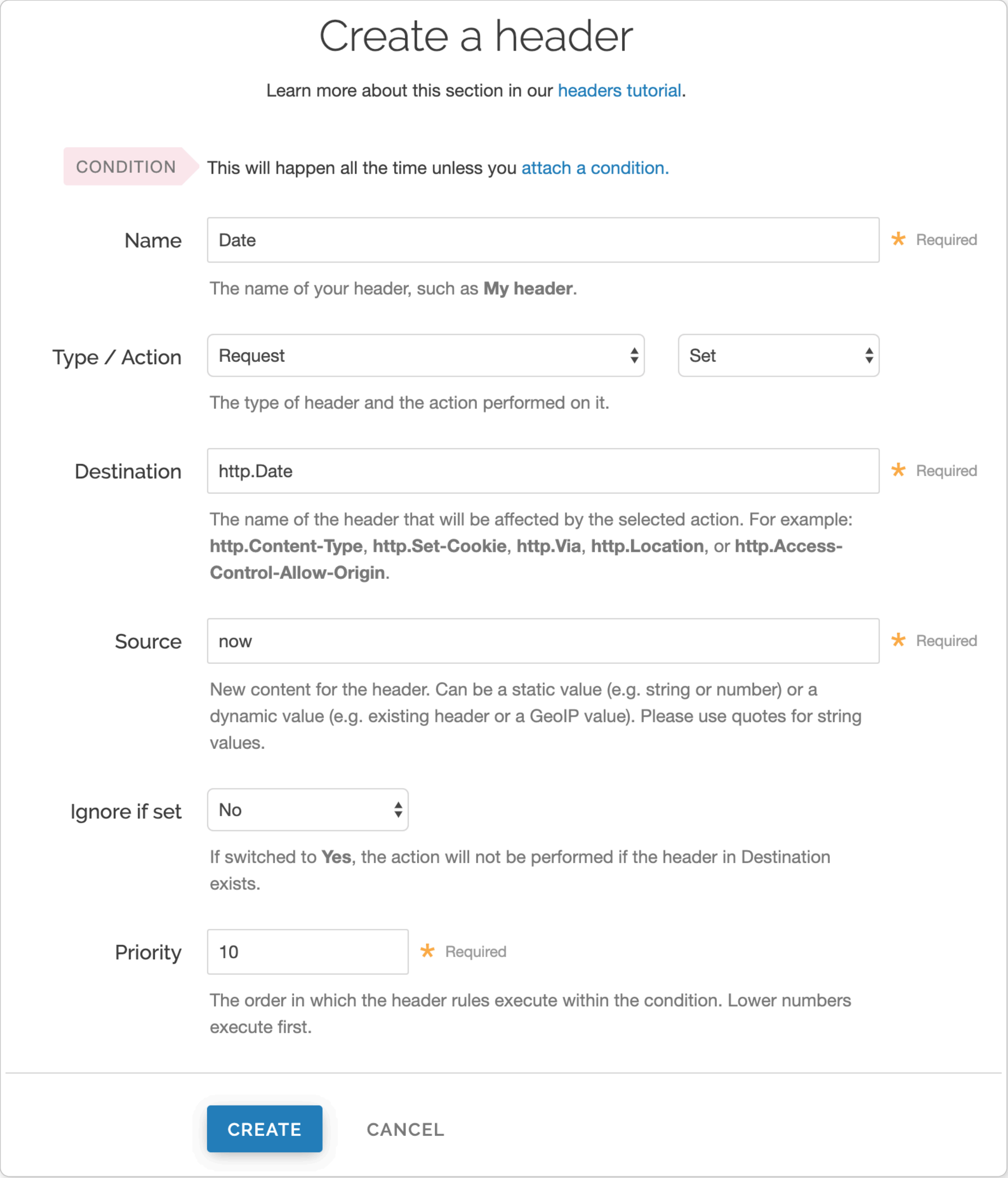
- the **access key** and **secret key**
- your **Space** name

Setting up Fastly to use a private DigitalOcean Space

In order to use a private DigitalOcean Space with Fastly, create two headers, a Date header (for use with the authorization Signature) and an Authorization header.

Create a Date header

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Date — Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Date — Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: now — Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 — Required

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Date`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Date`.
   - In the **Source** field, enter `now`.

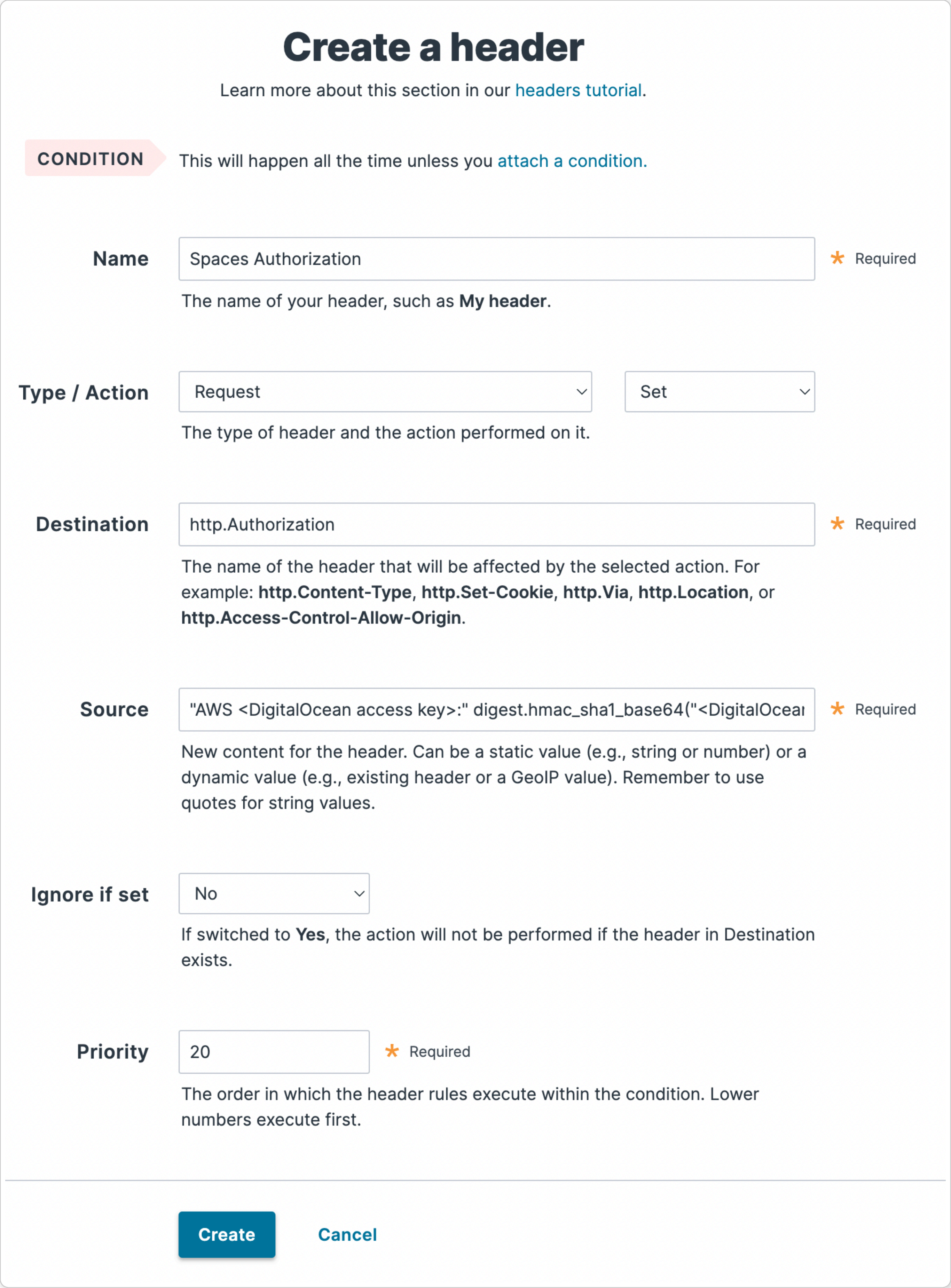
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**. A new Date header appears on the Content page. You will use this later within the Signature of the Authorization header.

Create an Authorization header

Next, create the Authorization header with the specifications listed below.

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name Spaces Authorization * Required

The name of your header, such as **My header**.

Type / Action Request | Set

The type of header and the action performed on it.

Destination http.Authorization * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source "AWS <DigitalOcean access key>:" digest.hmac_sha1_base64("<DigitalOcea * Required

New content for the header. Can be a static value (e.g., string or number) or a dynamic value (e.g., existing header or a GeoIP value). Remember to use quotes for string values.

Ignore if set No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority 20 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:

- In the **Name** field, enter `Spaces Authorization`.
- From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
- In the **Destination** field, enter `http.Authorization`.
- From the **Ignore if set** menu, select **No**.
- In the **Priority** field, enter `20`.

3. In the **Source** field, enter the header authorization information using the following format:

```
"AWS <DigitalOcean access key>:" digest.hmac_sha1_base64("<DigitalOcean secret key>' if
```

replacing `<DigitalOcean access key>`, `<DigitalOcean secret key ID>`, and `<Space name>` with the information you gathered before you began. For example:

```
"AWS JKCAUEFV2ONFFOFMSSLA:" digest.hmac_sha1_base64("P2WPSu68Bfl89j72vT+bXYZB7SjlOw wh
```

4. Click **Create**. The new Authorization header appears on the Content page.

A detailed look at the Source field

So what's going on in the Source field of the Authorization header? Here's the basic format:

`AWS<Access Key><Signature Function><key><message>`

It tells us the following:

Element	Description
`AWS`	A constant placed before the access key. It's always AWS.
`access key`	The access key from your DigitalOcean account. We used `JKCAUEFV2ONFFOFMSSLA` in this example.
`signature function`	The algorithm used to validate the key and message of the signature. We used `digest.hmac_sha1_base64(<key>, <message>)` in this example.
`key`	The secret key from your DigitalOcean account. We used `P2WPSu68BfI89j72vT+bXYZB7SjIOwhT4whqt27` in this example.
`message`	The UTF-8 encoding of the StringToSign. Check out the table below for a break down of each portion of the message.

The message that's part of the Source field in the Authorization header takes on this basic format:

`<HTTP-verb></n><Content-MD5>/n<Content-Type></n><Date></n><CanonicalizedAmzHeader></n><CanonicalizedResource>`

It tells us the following:

Element	Description
`HTTP-verb`	The REST verb. We use `req.method` in this example. We rewrite HEAD to GET because Varnish does this internally before sending requests to origin.
`/n`	A newline indicator constant. It's always /n.
`Content-MD5`	The content-md5 header value, used as a message integrity check. It's often left blank. We use `LF` (line feed) in this example.
`Content-Type`	The content-type header value, used to specify the MIME-type. It's often left blank. We use `LF`in this example.
`Date`	The date and time stamp. We use `req.http.Date` (which we created first as a separate header in the steps above).
`CanonicalizedAmzHeader`	The x-amz headers, which customize your Spaces implementation. It's often left blank. We use `LF` in this example.
`CanonicalizedResource`	Your DigitalOcean Space name. We use `"/test123"` in this example.

Following redirects to Spaces objects and caching Spaces responses

With custom VCL, Fastly can follow redirects to Spaces objects and cache the Spaces response as well as the 301 or 302 response separately.

Be sure to read our instructions about mixing and matching Fastly VCL with custom VCL. It's important to include the entire VCL boilerplate if you do not intend to override the Fastly default settings.

To configure Fastly to follow redirects to Spaces objects, insert the following VCL in your custom VCL:

Within `vcl_recv`

```
sub vcl_recv {

  if (req.http.redir != "true") {
    set req.backend = Main_Origin;
  } else {
    set req.backend = spaces_backend;
    set req.http.host = "nyc3.digitaloceanspaces.com";
  }

#FASTLY recv

  if (req.method != "HEAD" && req.method != "GET" && req.method != "FASTLYPURGE") {
    return(pass);
  }

  return(lookup);

}
```

Within `vcl_deliver`

```
sub vcl_deliver {

  if (resp.status == 302 || resp.status == 301) {
    set req.http.redir = "true";
    set req.url = regsub(resp.http.Location, "http://nyc3.digitaloceanspaces.com/(.*)$"
    set req.http.Fastly-Force-Shield = "yes";
    restart;
  }

#FASTLY deliver

  return(deliver);
}
```

Be sure to set the `Main_Origin` and `spaces_backend` to the actual name of your backends in the service to which you're applying these redirects. You can find the exact names by reviewing your VCL. Simply click on the VCL button at the top of the page while viewing the service.

Once you added these VCL snippets to your custom VCL, upload the VCL file and then activate the new version of your service to apply the changes.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Discounted egress from Google

/en/guides/discounted-egress-from-google

Fastly has partnered with Google to provide an integration between Fastly and Google services. Specifically, the integration allows you to connect Google's Cloud Platform service directly to Fastly's content delivery network services via private network interconnections (direct PNIs), thus speeding up your content delivery and optimizing backend workload.

When you sign up for Fastly services and configure a Google Cloud Platform service as your origin server, you designate a specific point of presence (POP) to serve as an Origin Shield that handles cached content from their servers.

Requests from Fastly POPs to these specific interconnect locations are routed over Fastly's network, leveraging optimized TCP connection handling, quick-start, and opened connections to enable fast response times between POPs and through to the end-user. Fastly ensures requests go directly to the Origin Shield instead of the origin servers. Only requests that the entire network has never handled will go back to the Google Cloud Platform service.

WARNING

We encourage you to read Google's CDN interconnect pricing. Despite this connection to Fastly's services being in place, in certain circumstances your data may egress from Google over the public internet rather than via the private network interconnection. In such cases, your traffic to the public internet will be metered according to your commercial arrangement with Google.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

	Google Cloud Storage
🔗	/en/guides/google-cloud-storage

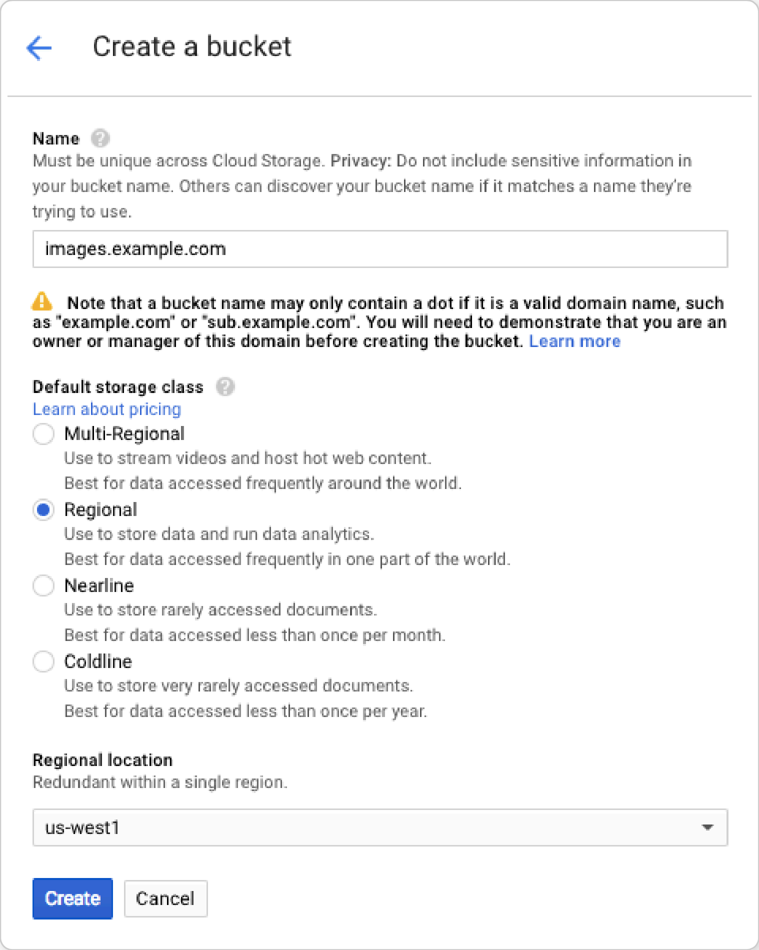
Google Cloud Storage (GCS) can be used as an origin with your Fastly services once you set up and configure your GCS account and link it to a Fastly service. It can also be configured to use private content. This speeds up your content delivery and reduces your origin's workload and response times with the dedicated links between Google and Fastly's POPs.

Using GCS as an origin server

To make your GCS data available through Fastly, follow the steps below.

Setting up and configuring your GCS account

1. Sign up for Google Cloud Storage.
2. Create a bucket to store your origin's data.



Create a bucket

Name

Must be unique across Cloud Storage. **Privacy:** Do not include sensitive information in your bucket name. Others can discover your bucket name if it matches a name they're trying to use.

images.example.com

⚠ Note that a bucket name may only contain a dot if it is a valid domain name, such as "example.com" or "sub.example.com". You will need to demonstrate that you are an owner or manager of this domain before creating the bucket. Learn more

Default storage class

Learn about pricing

○ Multi-Regional
Use to stream videos and host hot web content.
Best for data accessed frequently around the world.

● Regional
Use to store data and run data analytics.
Best for data accessed frequently in one part of the world.

○ Nearline
Use to store rarely accessed documents.
Best for data accessed less than once per month.

○ Coldline
Use to store very rarely accessed documents.
Best for data accessed less than once per year.

Regional location

Redundant within a single region.

us-west1

3. Fill out the **Create a bucket** fields as follows:
    - In the **Name** field, enter a name for your bucket (e.g., `mybucket`). You can also create a domain-named bucket (e.g., `images.example.com`), but you'll be required to verify your domain ownership using Google's Search Console, if you have not already done so. See the instructions on Google's website. Remember the name you type. You'll need it to connect your GCS bucket to your Fastly service.
    - In the **Default storage class** area, select **Regional**.
    - From the **Regional location** menu, select a location to store your content. Most customers select a region close to the interconnect location they specify for shielding.
4. Click **Create**.

You should now add objects to your bucket and make them externally accessible by selecting the **Public link** checkbox next to each of the objects.

Adding your GCS bucket as an origin server

To add your GCS bucket as an origin server, follow the instructions for working with hosts. You'll add specific details about your origin server.

1. In the **Hosts** field on the **Origins** page, enter the appropriate address for your Host using the format `<BUCKET>.storage.googleapis.com`. For example, if your bucket name is `test123`, your hostname would be `test123.storage.googleapis.com`.
2. For the initial **Edit this host** fields:
    - In the **Name** field, enter any descriptive name for your service (e.g., `Google Cloud Storage`).

   - In the **Address** field, enter the appropriate address for your Host using the format `<BUCKET>.storage.googleapis.com`. For example, if your bucket name is `mybucket`, your hostname would be `mybucket.storage.googleapis.com`.

3. When you edit the Transport Layer Security (TLS) area information for your host:
   - Leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
   - In the **Certificate hostname** field, enter `storage.googleapis.com`.
   - Under the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname**. The hostname address you entered during Host creation appears.
4. From the **Shielding** menu below the TLS area, select an interconnect location from the list of shielding locations.
5. In the **Override host** field, enter an appropriate address for your Host (e.g., `test123.storage.googleapis.com`). You entered this information during Host creation.

Interconnect locations

Interconnect locations allow you to establish direct links with Google's network edge when you choose your shielding location. By selecting one of the locations listed in our documentation, your traffic will be carried across our interconnections with Google and you will be eligible to receive Google's CDN partner pricing discount. Most customers select the interconnect closest to their GCS bucket's region. Review our caveats of shielding and select an interconnect accordingly.

Setting the Cache-Control header for your GCS bucket

By default, GCS performs its own caching for publicly readable objects, which may complicate efforts to purge cache. To avoid potential problems, we recommend using the gsutil command line utility to set the Cache-Control header for one or more objects in your GCS bucket:

```
$ gsutil setmeta -h "Cache-Control: no-store, max-age=86400" gs://<bucket>/*.html
```

Replace `<bucket>` in the example above with your GCS bucket's name. Note that `no-store` instructs GCS not to cache your content, while `max-age=86400` instructs Fastly to cache your content for one day. See Google's documentation on the `setmeta` command for more information.

Changing the default TTL for your GCS bucket

If you want to change the default TTL for your GCS bucket, if at all, keep the following in mind:

- Your GCS account controls the default TTL for your GCS content. GCS currently sets the default TTL to 3600 seconds. Changing the default TTL will not override the default setting in your GCS account.
- To override the default TTL set by GCS from within the Fastly web interface, create a new cache setting and enter the TTL there.
- To override the default TTL in GCS, download the gsutil tool and then change the Cache-Control headers to delete the default TTL or change it to an appropriate setting.

X-Http-Method-Override header behavior

GCS provides a unique functionality that allows clients to add a `X-Http-Method-Override` request header to override the request method being sent in the HTTP messages. For instance, a GET request with the `X-Http-`

`Method-Override: HEAD` request header is treated as a HEAD request by GCS and returns a HEAD response (200 status code with an empty body).

This can cause unintended caching behavior, which is a security risk. For example, if an `X-Http-Method-Override` request header is received and an unexpected response is cached. In order to minimize this risk, we strongly recommend you unset the `X-Http-Method-Override` header in the `vcl_recv` subroutine as shown below:

```
unset req.http.X-Http-Method-Override;
```

Using GCS with private objects

To use Fastly with GCS private objects, be sure you've already made your GCS data available to Fastly by pointing to the right GCS bucket, then follow the steps below.

Setting up interoperable access

By default, GCS authenticates requests using OAuth2, which Fastly does not support. To access private objects on GCS, your project must have HMAC authentication enabled and interoperable storage access keys (an *Access Key* and *Secret* pair) created. Do this by following the steps below.

> TIP
>
> To limit access to your Google Cloud account, consider creating a service account and then creating HMAC keys for that service account. For more information, see Google's documentation on managing HMAC keys for service accounts.

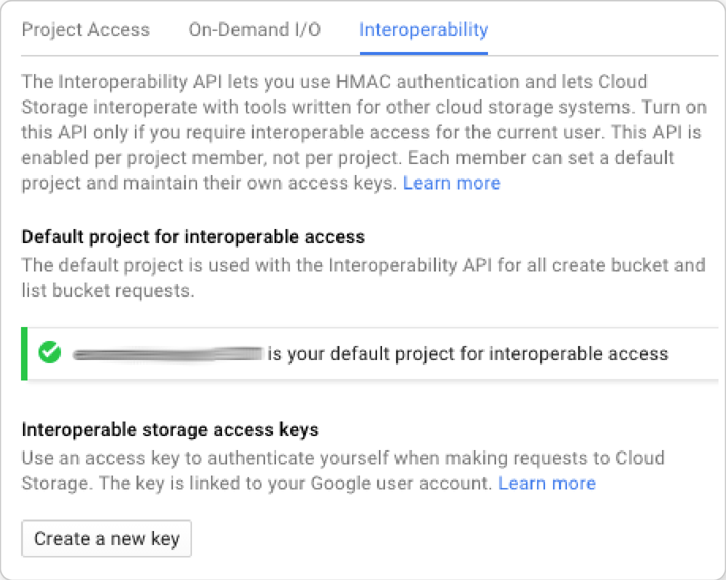
1. Open the Google Cloud Platform console and select the appropriate project.

2. Click **Settings**.

3. Click **Interoperability**.

4. If you have not set up interoperability before, click **Enable interoperability access**.

5. Click **Make** `<PROJECT-ID>` **your default project** for interoperable access. If that project already serves as the default project, that information appears instead.



The Interoperability API lets you use HMAC authentication and lets Cloud Storage interoperate with tools written for other cloud storage systems. Turn on this API only if you require interoperable access for the current user. This API is enabled per project member, not per project. Each member can set a default project and maintain their own access keys. Learn more

Default project for interoperable access

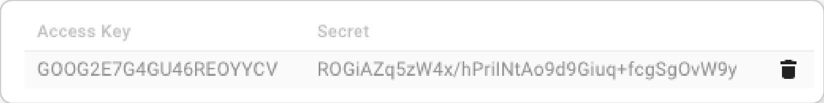
The default project is used with the Interoperability API for all create bucket and list bucket requests.

✓ [redacted] is your default project for interoperable access

Interoperable storage access keys

Use an access key to authenticate yourself when making requests to Cloud Storage. The key is linked to your Google user account. Learn more

6. Click **Create a new key**. An access key and secret code appear.



Access Key	Secret
GOOG2E7G4GU46REOYYCV	ROGiAZq5zW4x/hPrilNtAo9d9Giuq+fcgSgOvW9y

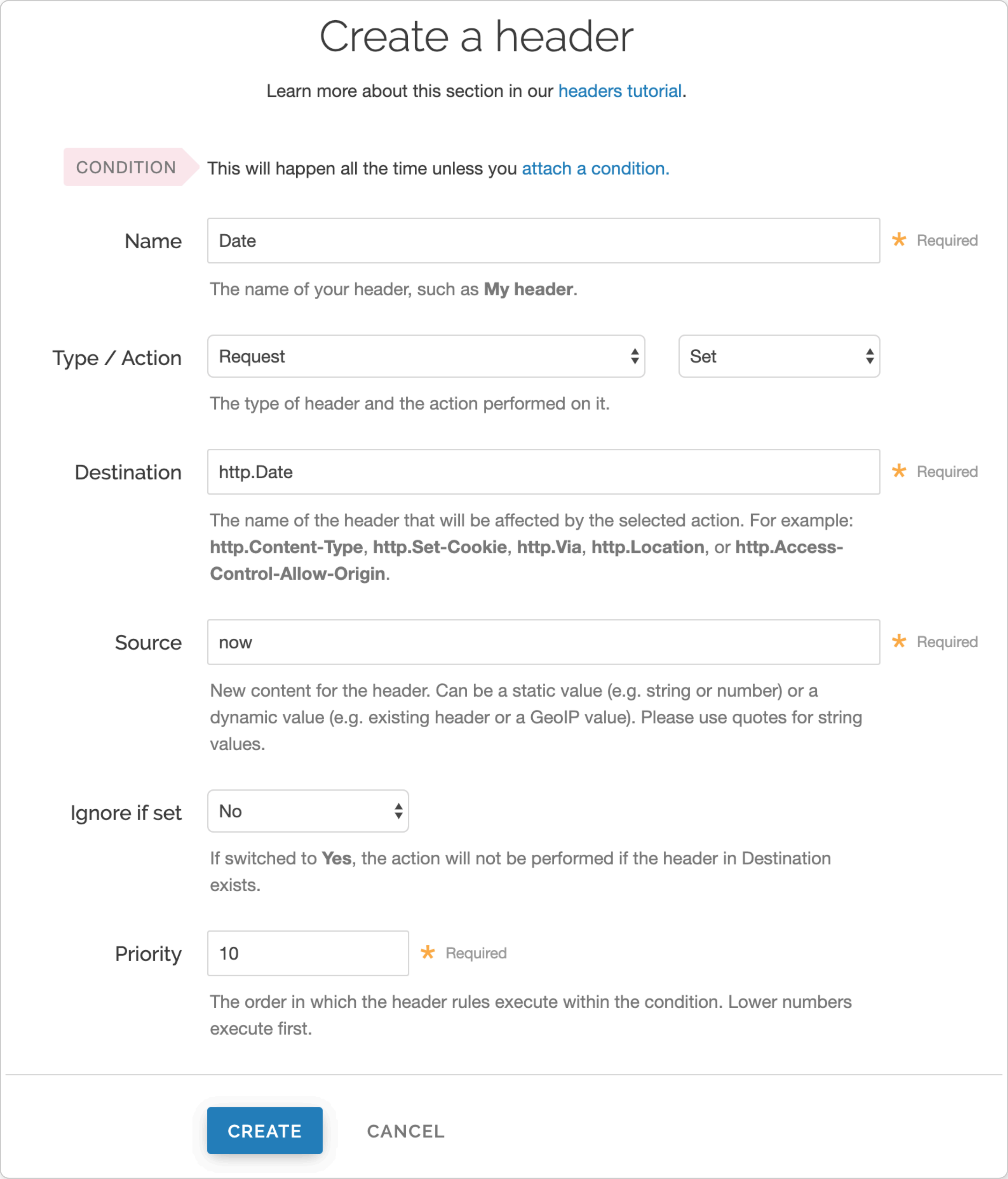
7. Save the access key and secret code that appear. You'll need these later when you're creating an authorization header.

Setting up Fastly to use GCS private content

To use GCS private content with Fastly, create two headers, a Date header (required Authorization Signature) and an Authorization header.

Creating a Date header

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Date ✱ Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Date ✱ Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: now ✱ Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 ✱ Required

The order in which the header rules execute within the condition. Lower numbers execute first.

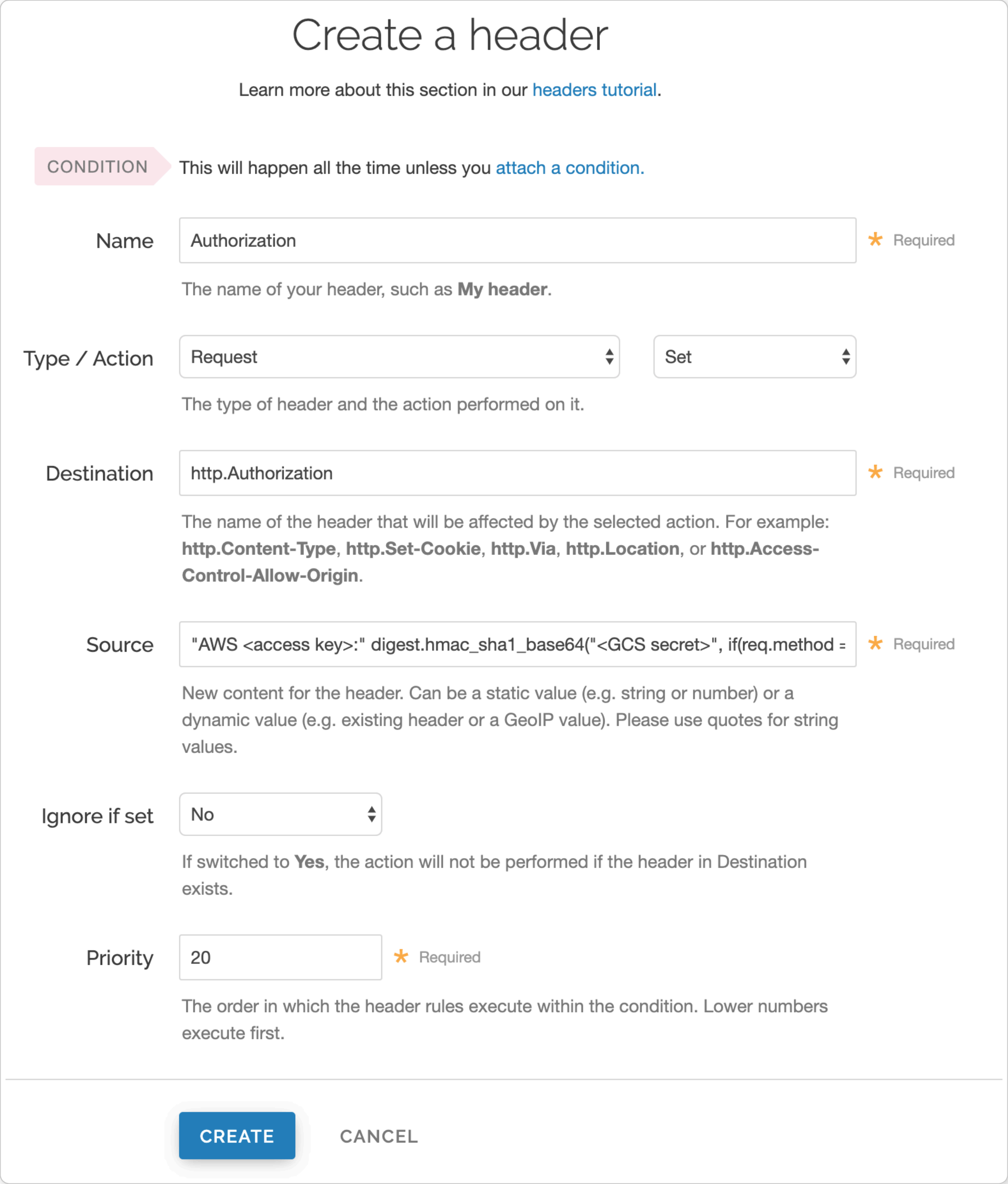
6. Fill out the **Create a new header** fields as follows:
   - In the **Name** field, enter `Date`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Date`.
   - In the **Source** field, enter `now`.

  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**. A new Date header appears on the Content page. You will use this later within the Signature of the Authorization header.

Creating an Authorization header

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Authorization * Required

The name of your header, such as **My header**.

Type / Action: Request | Set

The type of header and the action performed on it.

Destination: http.Authorization * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "AWS <access key>:" digest.hmac_sha1_base64("<GCS secret>", if(req.method = * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Authorization`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `http.Authorization`.
    - From the **Ignore if set** menu, select **No**.

  - In the **Priority** field, enter `20`.

3. In the **Source** field, enter the header authorization information using the following format:

   ```
   "AWS <access key>:" digest.hmac_sha1_base64("<GCS secret>", if(req.method == "HEAD", GE
   ```

   replacing `<access key>`, `<GCS secret>`, and `<GCS bucket name>` with the information you gathered before you began. For example:

   ```
   "AWS GOOGQORE5WOJJHLXH6OD:" digest.hmac_sha1_base64("oQb0hdmaxFOc5UmC6F833Cde0+ghRSg 7C
   ```

4. Click **Create**. A new Authorization header appears on the Content page.

5. Click **Activate** to deploy your configuration changes.

A detailed look at the Source field

So what's going on in the Source field of the Authorization header? Here's the basic format:

`AWS<access key><signature function><key><message>`

It tells us the following:

Element	Description
`AWS`	A constant placed before the access key. It's always AWS.
`access key`	The access key ID from your GCS developer's account. We used `GOOGQORE5WOJJHLXH6OD` in this example.
`signature function`	The algorithm used to validate the key and message of the signature. We used `digest.hmac_sha1_base64(<key>, <message>)` in this example.
`key`	The secret key ID from your GCS developer's account. We used `oQb0hdmaxFOc5UmC6F833Cde0+ghRSgsr7CCnX62` in this example.
`message`	The UTF-8 encoding of the StringToSign. See the table below for a break down of each portion of the message.

The message that's part of the Source field in the Authorization header takes on this basic format:

`<HTTP-verb><\n><Content-MD5>\n<Content-Type><\n><Date><\n><CanonicalExtensionHeaders><\n><CanonicalizedResource>`

It tells us the following:

Element	Description
`HTTP-verb`	The REST verb. We use `req.method` in this example.
`\n`	A newline indicator constant. It's always \n.
`Content-MD5`	The content-md5 header value, used as a message integrity check. It's often left blank. We use `LF` (line feed) in this example.

Element	Description
`Content-Type`	The content-type header value, used to specify the MIME-type. It's often left blank. We use `LF` in this example.
`Date`	The date and time stamp. We use `req.http.Date` (which we created first as a separate header in the steps above).
`CanonicalExtensionHeaders`	The x-amz- or x-goog- headers, which customize your GCS implementation. It's often left blank. We use `LF` in this example.
`CanonicalizedResource`	Your GCS resource path name. We're concatenating GCS bucket name `"/test123"` with object path `req.url.path` in this example.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

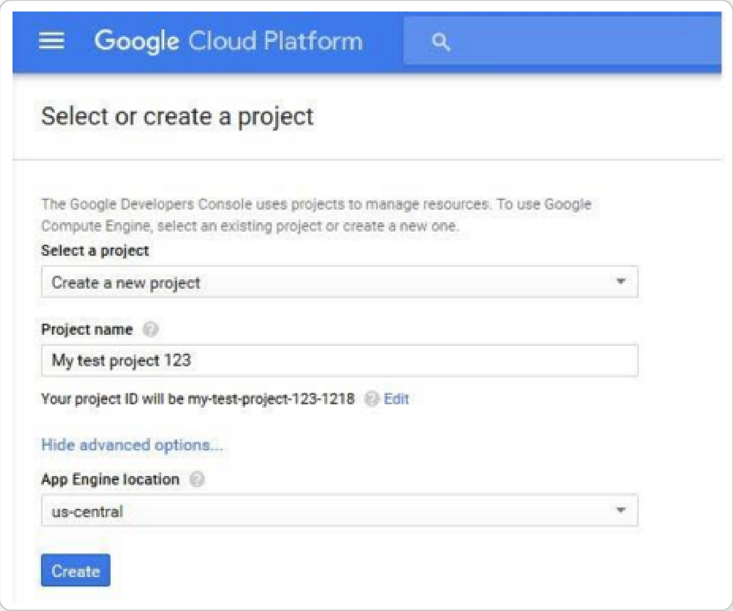
Google Compute Engine

🔗 /en/guides/google-compute-engine

Google Compute Engine (GCE) lets you create and run a virtual machine (VM) on the Google infrastructure. The VM can be used as an origin with your Fastly service once you set up and configure your VM instance and link your instance to a Fastly service.

Creating and setting up your GCE instance

1. Sign up for Google Compute Engine and start the basic set up. If you are already signed up and at your dashboard, click the **Get started** link in the Try Compute Engine area.
2. Create or select a project to hold your origin's data.



Select or create a project

The Google Developers Console uses projects to manage resources. To use Google Compute Engine, select an existing project or create a new one.

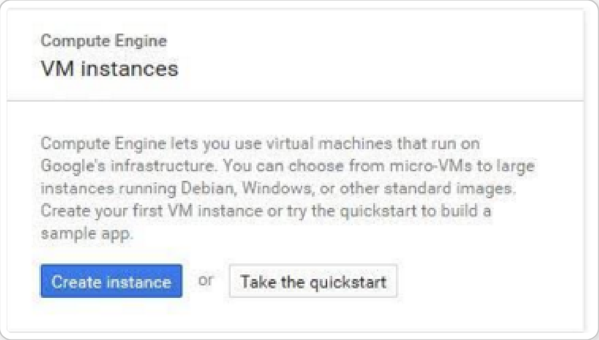
Select a project: Create a new project

Project name: My test project 123

Your project ID will be my-test-project-123-1218 Edit

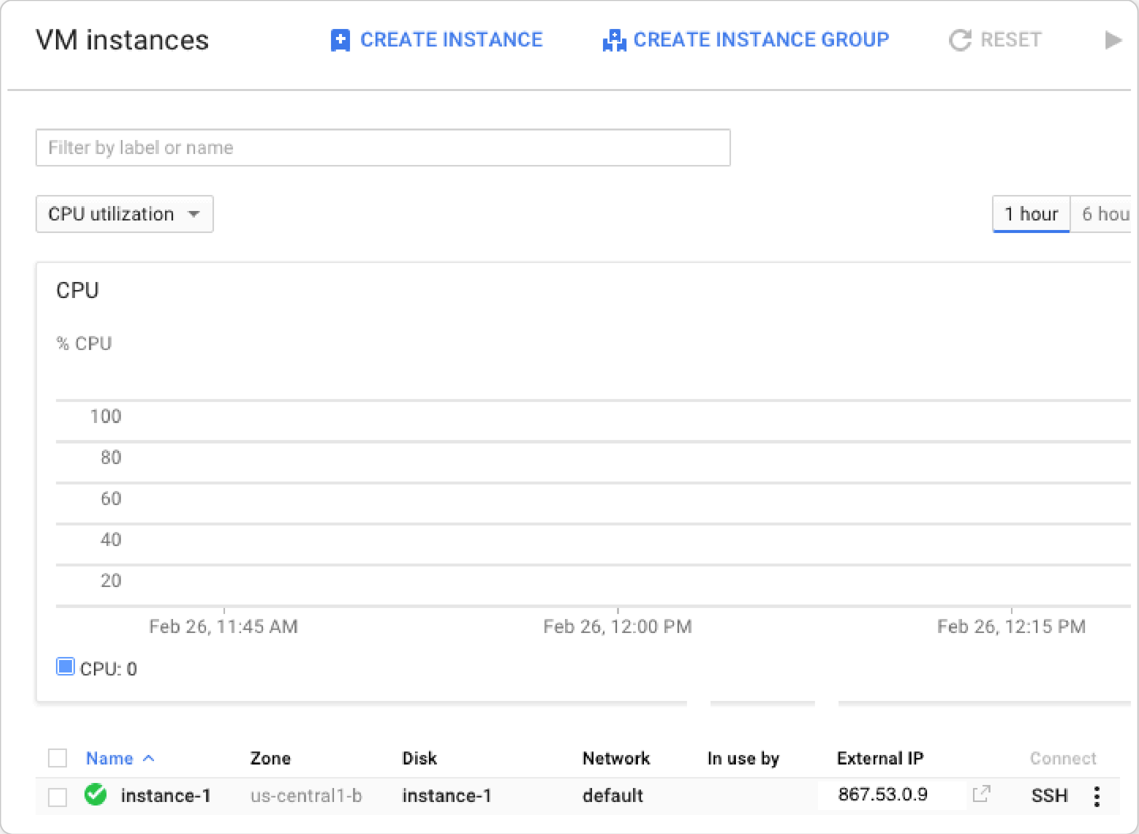
App Engine location: us-central

3. Click **Create instance** to set up your VM. You can set up your instance using either Windows or Linux.



Compute Engine lets you use virtual machines that run on Google's infrastructure. You can choose from micro-VMs to large instances running Debian, Windows, or other standard images. Create your first VM instance or try the quickstart to build a sample app.

4. Fill in the necessary fields and click **Create**. When making your firewall selection, select either **Allow HTTPS traffic** (port 443) or **Allow HTTP traffic** (port 80); you will use one of those ports when you create your new origin in your Fastly service. If you select HTTPS traffic, you need to configure the VM to respond on port 443 with a valid TLS certificate.



☐	Name	Zone	Disk	Network	In use by	External IP	Connect
☐	✓ instance-1	us-central1-b	instance-1	default		867.53.0.9	SSH

5. Make note of the following information for when you create your new origin in your Fastly service:
    - The instance's IP address (located in the External IP column at the bottom of the page). You'll use this in the **Address** field when you create your new origin.
    - The zone you are using (located in the Zone column at the bottom of the page). You'll use this to guide your selection of an appropriate shielding location for your origin.

Creating a new origin in your Fastly service for your GCE account

Link your GCE account to a Fastly service following the steps below.

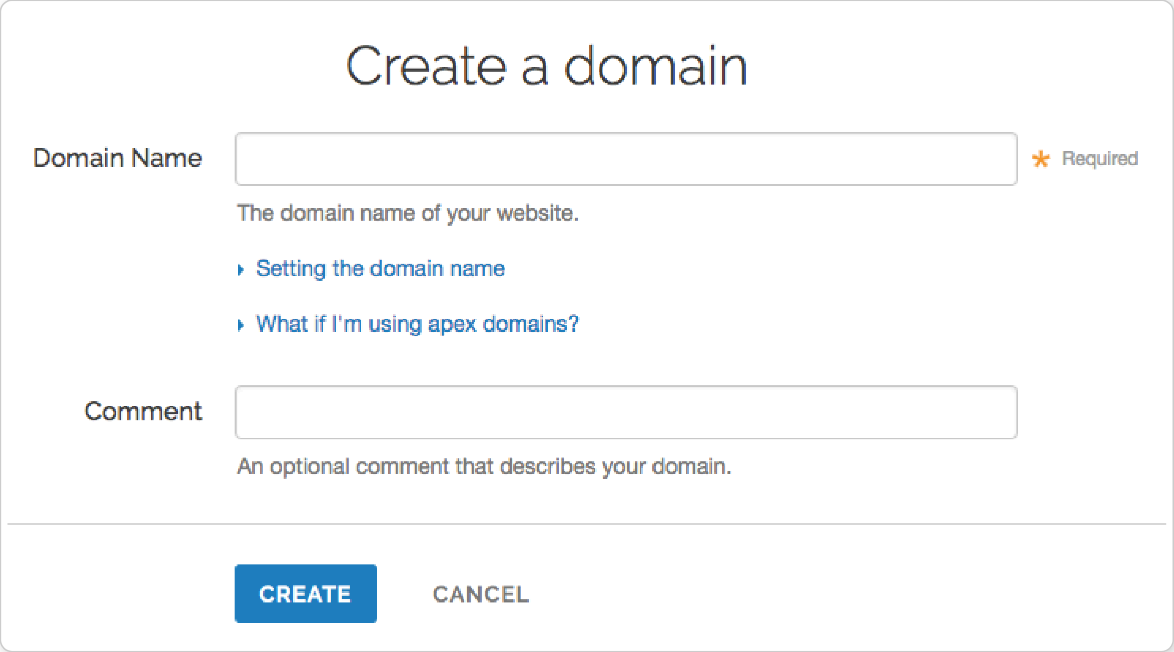
1. Log in to the Fastly web interface.
2. Create a new service if you don't already have one set up.
3. Follow the instructions for working with hosts. You'll add specific details about your origin server when you fill out the **Create a host** fields:
    - In the **Name** field, enter the name of your server (for example, `Google Compute Engine`).
    - In the **Address** field, enter the IP address of your server. This should match the port that you selected in the GCE interface.
    - From the **Shielding** menu, select an interconnect location from the list of shielding locations.

Interconnect locations

Interconnect locations allow you to establish direct links with Google's network edge when you choose your shielding location. By selecting one of the locations listed in our documentation, you will be eligible to receive discounted pricing from Google CDN Interconnect for traffic traveling from Google Cloud Platform to Fastly's network. Most customers select the interconnect closest to their origin. Review our caveats of shielding and select an interconnect accordingly.

Creating new domains for GCE to respond to

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Create domain**.



Create a domain

Domain Name ☐ * Required

The domain name of your website.

Comment ☐

An optional comment that describes your domain.

5. In the **Domain Name** field, enter the name that users will enter in their browsers to access your site.
6. *(Optional)* In the **Comment** field, enter a comment that describes your domain.
7. Click **Create**. The new domain appears on the Domains page.
8. Click **Activate** to deploy your configuration changes.

Creating a CNAME record

You can now test your configuration. In the example above, your domain would appear as `www.example.com.global-nossl.fastly.net`. After you test and you're satisfied with the results, create a CNAME record for your domain (e.g., `www.example.com`) pointing to `global-nossl.fastly.net`.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

HUMAN Bot Defender

/en/guides/human-bot-defender

IMPORTANT

This information is part of a limited availability release. For additional details, read our product and feature lifecycle descriptions.

Fastly provides direct integration between HUMAN Bot Defender (formerly PerimeterX Bot Defender) and Fastly edge servers. By placing a snippet of JavaScript (or HTML5) on your site and custom VCL directly into your Fastly service configuration, this integration allows you to gather behavioral data and statistics that may help you do things like detect invalid traffic and mitigate automated web attacks.

How to get started

Integration with HUMAN Bot Defender requires an account with HUMAN. Once you have this account set up, contact your Fastly account manager or email sales@fastly.com to begin the integration process with Fastly. We'll work with you to configure your service to include the required code to enforce bot mitigation policies.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Microsoft Azure Blob Storage

/en/guides/microsoft-azure-blob-storage

Microsoft Azure Blob Storage public and private containers can be used as origins with Fastly.

TIP

With properly configured services in place, shared Fastly and Microsoft customers will benefit from Fastly's integration with Azure's ExpressRoute Direct Local, which results in Fastly including your outbound data transfer costs from Azure in your standard Fastly pricing. See our guide to outbound data transfers from Azure for more details.

Using Azure Blob Storage as an origin

When using Azure Blob Storage as an origin, we recommend using the most recent version of Azure storage services. Certain incompatibilities may occur if using too old a version or if the version is not specified by the request header. To enforce a version to use when not specified by the header, set the `DefaultServiceVersion` in your Blob storage service.

Alternatively, place a VCL snippet to `vcl_miss` and `vcl_pass` to set or enforce the version. For example, if the version is `2020-06-12`, the VCL snippet would look like the following:

```
{{if (req.backend.is_origin) { }}
{{ set bereq.http.x-ms-version = "2020-06-12"; }}
}
```

Once you've configured your Azure Blob Storage stores are ready to make them available through Fastly, follow the steps below.

Creating a new service

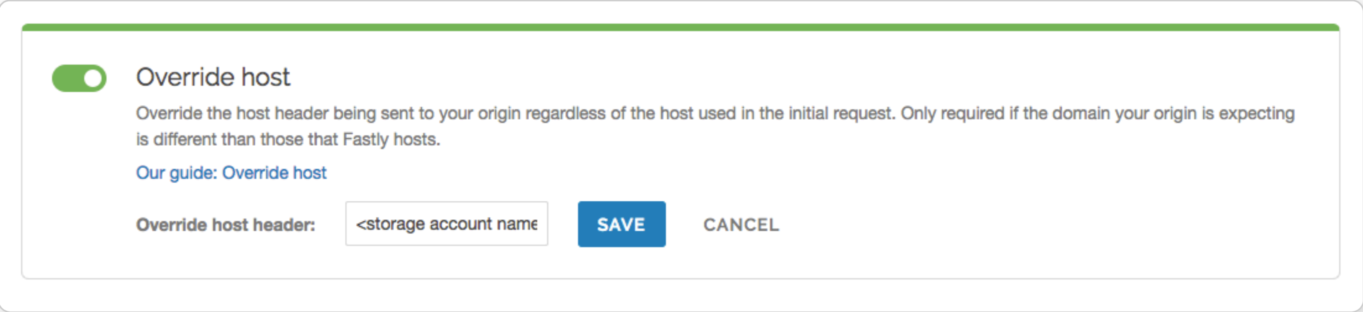
Follow the instructions for creating a new service. You'll add specific details about your origin when you fill out the **Create a new service** fields:

- In the **Name** field, enter any descriptive name for your service.
- In the **Domain** field, enter the hostname you want to use as the URL (e.g., `cdn.example.com`).
- In the **Address** field, enter `<storage account name>.blob.core.windows.net`.
- In the **Transport Layer Security (TLS)** area, leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
- In the **Transport Layer Security (TLS)** area, enter `<storage account name>.blob.core.windows.net` in the **Certificate hostname** field.

Setting the default Host and correct path

Once the new service is created, set the default Host to `azure` and then add your container path to the URL by following the steps below.

1. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
2. Click **Edit configuration** and then select the option to clone the active version.
3. Click **Settings**.
4. Click the **Override host** switch.



Override host

Override the host header being sent to your origin regardless of the host used in the initial request. Only required if the domain your origin is expecting is different than those that Fastly hosts.

Our guide: Override host

Override host header: <storage account name

5. Enter the hostname of your Azure Blob Storage account. For example, `<storage account name>.blob.core.windows.net`.

6. Click **Save**.
7. Click **Content**.
8. Click **Create header**.
9. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Modify URL`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `url`.
    - In the **Source** field, enter `"/<your container name>" req.url`.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `10`.
10. Click **Create**.
11. Click **Activate** to deploy your configuration changes.

Testing your results

By default, we create DNS mapping called **yourdomain.global.prod.fastly.net**. In the example above, it would be `cdn.example.com.global.prod.fastly.net`. Create a DNS alias for the domain name you specified (e.g., CNAME `cdn.example.com` to `global-nossl.fastly.net`).

Fastly will cache any content without an explicit `Cache-Control` header for 1 hour. You can verify whether you are sending any cache headers using curl. For example:

```
$ curl -I opscode-full-stack.blob.core.windows.net

HTTP/1.1 200 OK
Date: Fri, 04 May 2018 21:23:07 GMT
Content-Type: application/xml
Transfer-Encoding: chunked
Server: Blob Service Version 1.0 Microsoft-HTTPAPI/2.0
```

In this example, no Cache-Control headers are set so the default TTL will be applied.

Using an Azure Blob Storage private container

To use an Azure Blob Storage private container with Fastly, follow the instructions below.

Before you begin

Be sure you've already made your Azure Blob Storage containers available to Fastly by pointing to the right container and setting your origin to port 443. This needs to be done before authenticating.

To complete the setup, you'll also need your Azure Storage Account shared key and storage account name to construct the Azure Blob Storage Authorization header, which takes the following form:

```
Authorization: SharedKey `Account name`:`Signature`
```

Finally, you'll also need to know your Blob Storage container name.

Setting up Fastly to use an Azure Blob Storage private container with a Shared Key

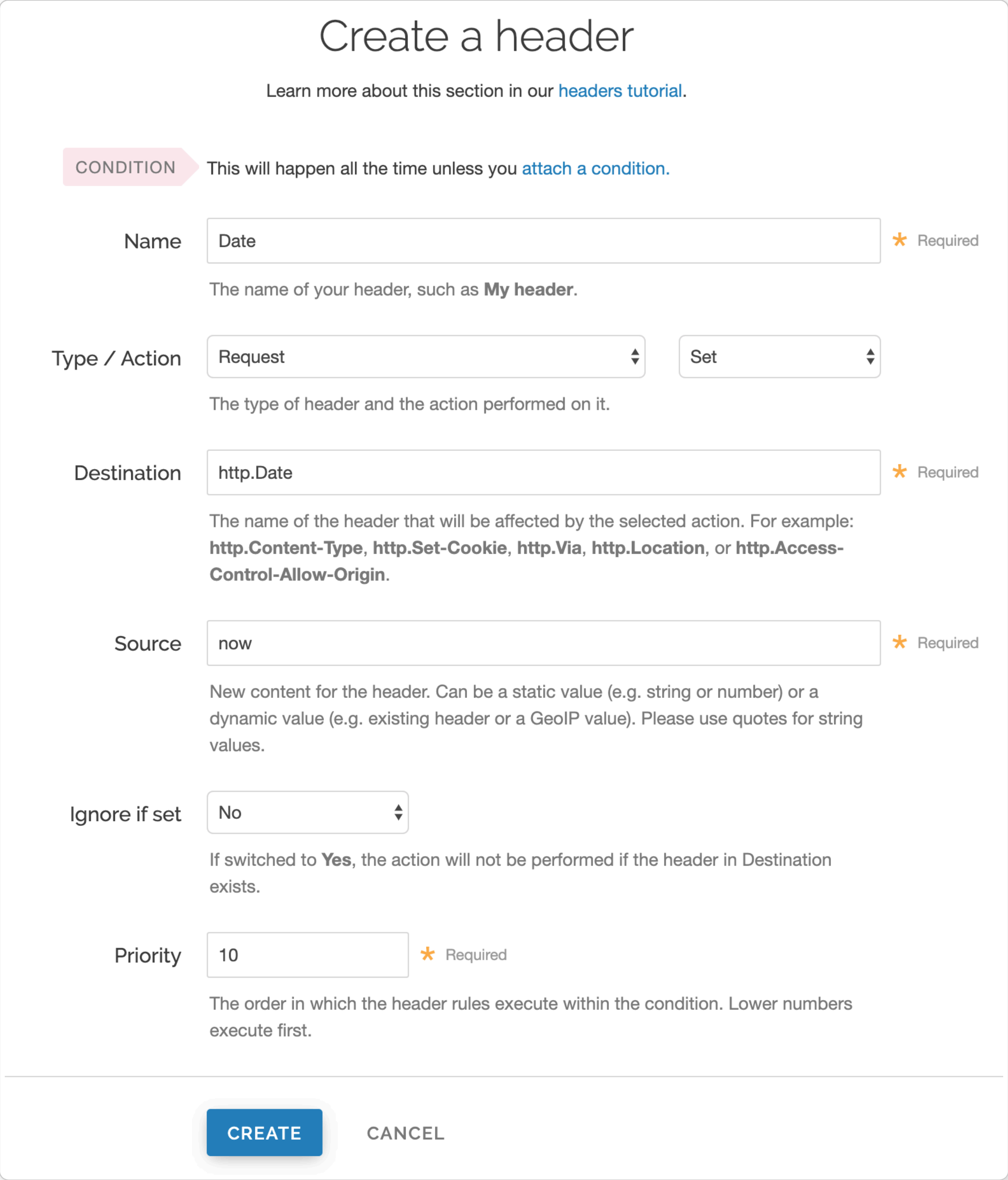
> WARNING
>
> Your account's Shared Key does not have detailed access control. Anyone with access to your Shared Key can read and write to your container. Consider using a Shared Access Signature (SAS) instead.

To access an Azure Blob Storage private container with Fastly using a Shared Key, first familiarize yourself with Microsoft's documentation on authorizing with Shared Key. Then, create two headers: a Date header (for use with the authorization Signature) and an Authorization header.

Create a Date header

Create the Date header using the steps below.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Date (Required)

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Date (Required)

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: now (Required)

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 10 (Required)

The order in which the header rules execute within the condition. Lower numbers execute first.

6. Fill out the **Create a header** fields as follows:
   - In the **Name** field, enter `Date`.
   - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
   - In the **Destination** field, enter `http.Date`.
   - In the **Source** field, enter `now`.

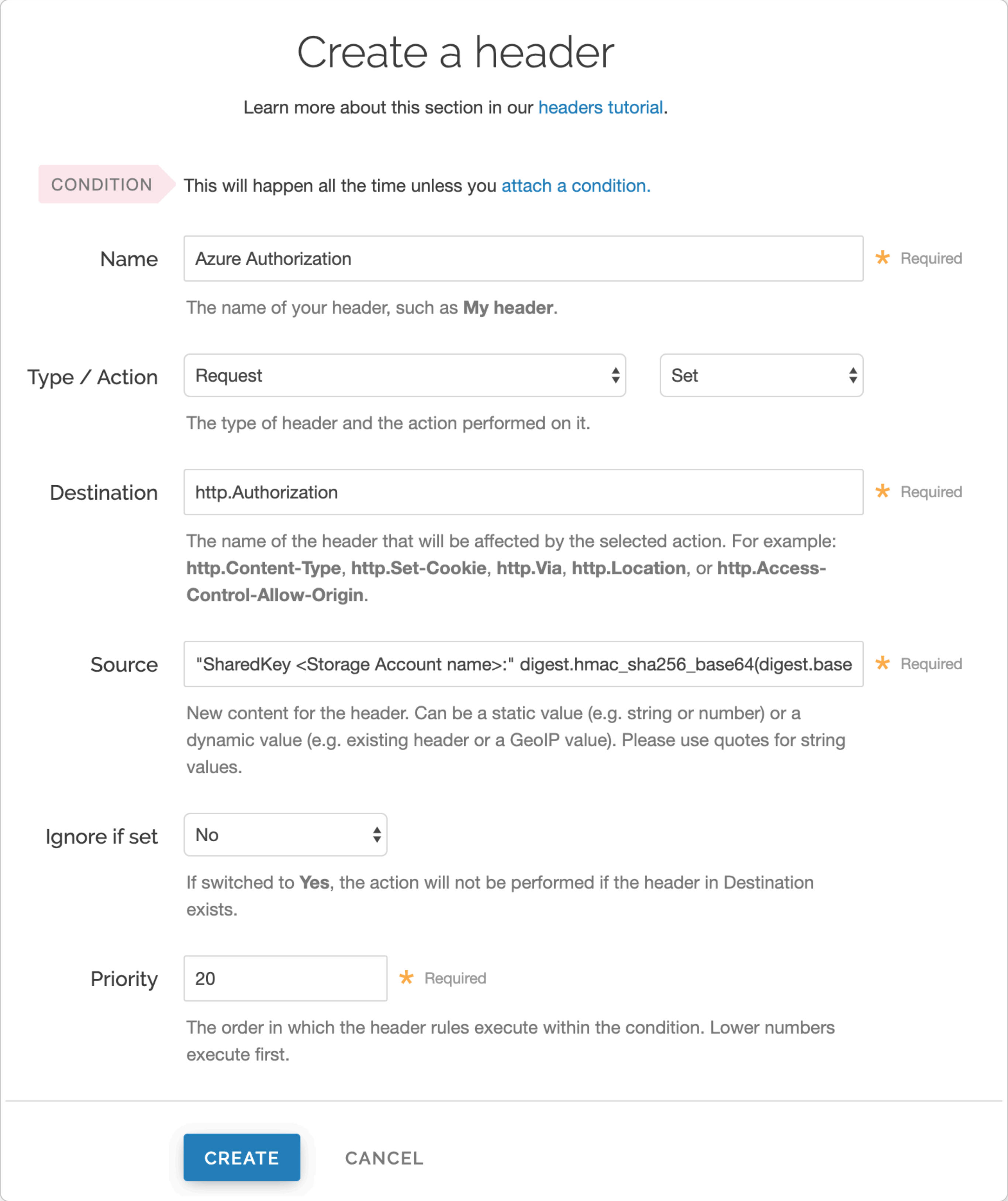
  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `10`.

7. Click **Create**. A new Date header appears on the Content page. You will use this later within the Signature of the Authorization header.

Create an Authorization header

Next, create the Authorization header with the specifications listed below.

1. Click **Create header** again to create another new header.



Create a header

Learn more about this section in our headers tutorial.

CONDITION This will happen all the time unless you attach a condition.

Name: Azure Authorization * Required

The name of your header, such as **My header**.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: http.Authorization * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "SharedKey <Storage Account name>:" digest.hmac_sha256_base64(digest.base * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Azure Authorization`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `http.Authorization`.

  - From the **Ignore if set** menu, select **No**.
  - In the **Priority** field, enter `20`.

3. In the **Source** field, enter the header authorization information using the following format:

   ```
   "SharedKey <Storage Account name>:" digest.hmac_sha256_base64(digest.base64_decode(' zu
   ```

   replacing `<Storage Account name>` and `<Azure Storage Account shared key>` with the information you gathered before you began. For example:

   ```
   "SharedKey test123:" digest.hmac_sha256_base64(digest.base64_decode("UDJXUFN1NjhCZmv No
   ```

   We provide a detailed look at the Source field parameters below.

4. Click **Create**. The new Authorization header appears on the Content page.

5. Click **Activate** to deploy your configuration changes.

A detailed look at the Source field

So what's going on in the Source field of the Authorization header? Here's the basic format:

`SharedKey<storage account name><Signature Function><key><message>`

It tells us the following:

Element	Description
`SharedKey`	A constant placed before the storage account name. It's always SharedKey.
`storage account name`	The name of your Azure Storage Account. We used `test123` in this example.
`signature function`	The algorithm used to validate the key and message of the signature. We used `digest.hmac_sha256_base64(<key>, <message>)` in this example.
`key`	The Azure Storage Account shared key from your Azure Storage developer's account. We used `UDJXUFN1NjhCZmw4OWo3MnZUK2JYWVpCN1NqbE93aFQ0d2hxdDI3` in this example. It must be Base64 decoded.
`message`	The UTF-8 encoding of the StringToSign. See the table below for a break down of each portion of the message.

The message that's part of the Source field in the Authorization header takes on this basic format:

`<HTTP-verb></n><Content-MD5>/n<Content-Type></n><Date></n><CanonicalizedAmzHeader></n><CanonicalizedResource>`

It tells us the following:

Element	Description
`HTTP-verb`	The REST verb. We use `req.method` in this example. We rewrite HEAD to GET because Varnish does this internally before sending requests to origin.
`\n`	A newline indicator constant. It's always \n.
`Content-MD5`	The content-md5 header value, used as a message integrity check. It's often left blank. We use `LF` (line feed) in this example.
`Content-Type`	The content-type header value, used to specify the MIME-type. It's often left blank. We use `LF` in this example.
`Date`	The date and time stamp. We use `req.http.Date` (which we created first as a separate header in the steps above).
`CanonicalizedHeaders`	The x-ms headers, which customize your Azure Blob Storage implementation. It's often left blank. We use `LF` in this example.
`CanonicalizedResource`	Your Storage Account Name. We use `"/test123"` in this example.

Setting up Fastly to use an Azure Blob Storage private container with a Shared Access Signature (SAS)

To access an Azure Blob Storage private container with Fastly using a Service Shared Access Signature (SAS), read Microsoft's Delegate access with a shared access signature page. Then, obtain the SAS and sign the access URL.

TIP

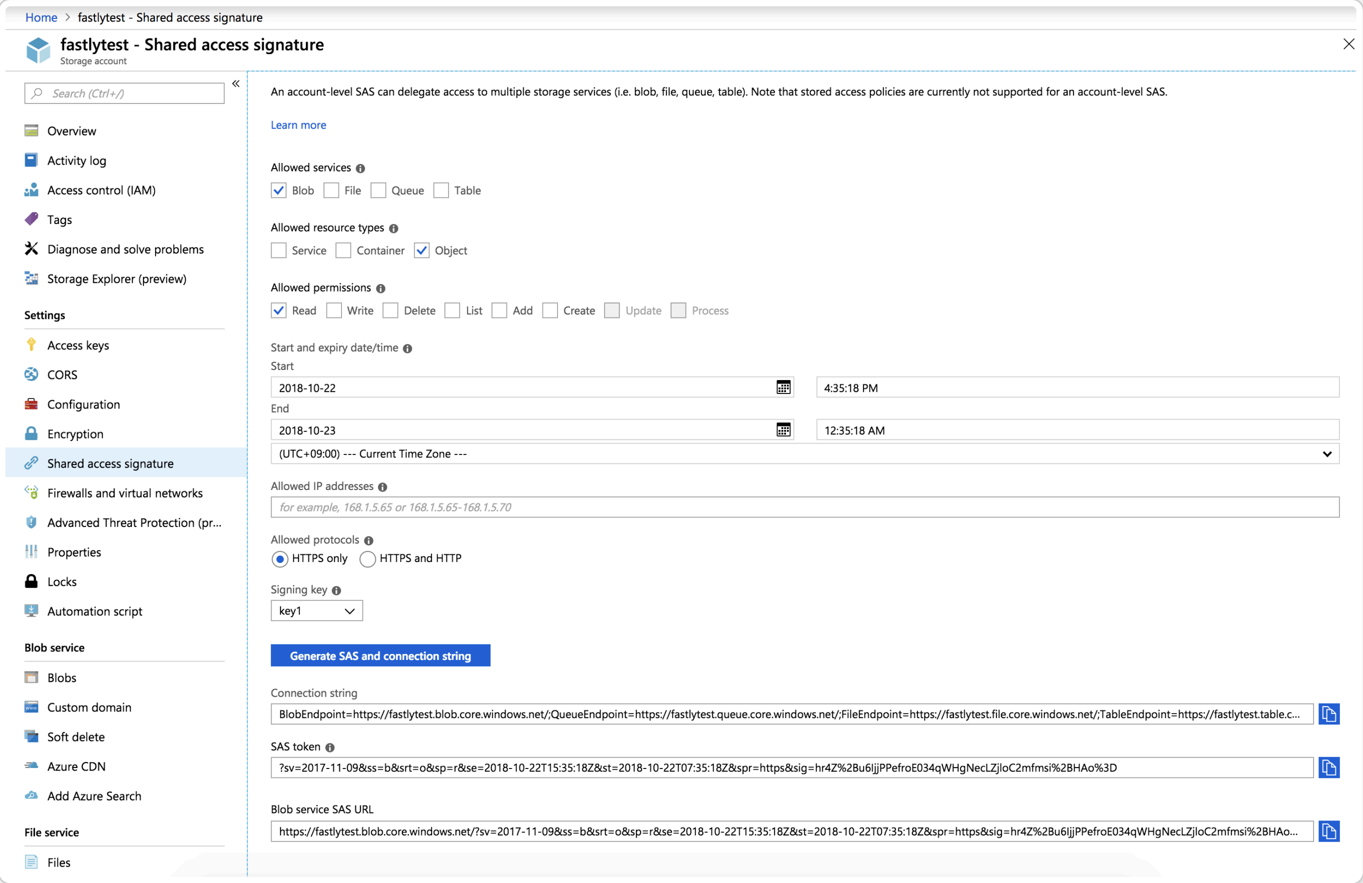
Using a Service Shared Access Signature gives you more detailed control over:

- The interval during which the SAS is valid, including the start time and the expiry time.
- The permissions granted by the SAS. For example, a SAS for a blob might grant read and write permissions to that blob, but not delete permissions.
- An optional IP address or range of IP addresses from which Azure Storage will accept the SAS. For example, you might specify a range of IP addresses belonging to your organization.
- The protocol over which Azure Storage will accept the SAS. You can use this optional parameter to restrict access to clients using HTTPS.

Obtaining the Shared Access Signature

Obtain the SAS using the steps below.

1. In the Azure portal, navigate to your storage account
2. Under settings navigate to **Shared access signature**.


Home > fastlytest - Shared access signature
fastlytest - Shared access signature
Storage account
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Storage Explorer (preview)
Settings
Access keys
CORS
Configuration
Encryption
Shared access signature
Firewalls and virtual networks
Advanced Threat Protection (pr...
Properties
Locks
Automation script
Blob service
Blobs
Custom domain
Soft delete
Azure CDN
Add Azure Search
File service
Files
Learn more
Generate SAS and connection string

An account-level SAS can delegate access to multiple storage services (i.e. blob, file, queue, table). Note that stored access policies are currently not supported for an account-level SAS.

Allowed services: ☑ Blob ☐ File ☐ Queue ☐ Table

Allowed resource types: ☐ Service ☐ Container ☑ Object

Allowed permissions: ☑ Read ☐ Write ☐ Delete ☐ List ☐ Add ☐ Create ☐ Update ☐ Process

Start and expiry date/time

Start: 2018-10-22 4:35:18 PM

End: 2018-10-23 12:35:18 AM

(UTC+09:00) --- Current Time Zone ---

Allowed IP addresses: *for example, 168.1.5.65 or 168.1.5.65-168.1.5.70*

Allowed protocols: ● HTTPS only ○ HTTPS and HTTP

Signing key: key1

Connection string

```
BlobEndpoint=https://fastlytest.blob.core.windows.net/;QueueEndpoint=https://fastlytest.queue.core.windows.net/;FileEndpoint=https://fastlytest.file.core.windows.net/;TableEndpoint=https://fastlytest.table.c...
```

SAS token

```
?sv=2017-11-09&ss=b&srt=o&sp=r&se=2018-10-22T15:35:18Z&st=2018-10-22T07:35:18Z&spr=https&sig=hr4Z%2Bu6ljjPPefroE034qWHgNecLZjloC2mfmsi%2BHAo%3D
```

Blob service SAS URL

```
https://fastlytest.blob.core.windows.net/?sv=2017-11-09&ss=b&srt=o&sp=r&se=2018-10-22T15:35:18Z&st=2018-10-22T07:35:18Z&spr=https&sig=hr4Z%2Bu6ljjPPefroE034qWHgNecLZjloC2mfmsi%2BHAo...
```

3. From the **Allowed services** controls, select **Blob**.
4. From the **Allowed resource types** controls, select **Object**.
5. From the **Allowed permissions** controls, select **Read**.
6. Leave the **Start time** set to the current date and time.
7. Set the **End time** as far in the future as you are comfortable (see note below).
8. Ensure the **Allowed protocols** remain set to **HTTPS only**.
9. Click **Generate SAS and connection string**. The generated information appears.
10. Copy and save the contents of the **SAS token** field. It will look something like:

```
?sv=2017-11-09&ss=b&srt=o&sp=r&se=2019-10-22T15:41:23Z&st=2018-10-22T07:41:23Z&spr=h  ps
```

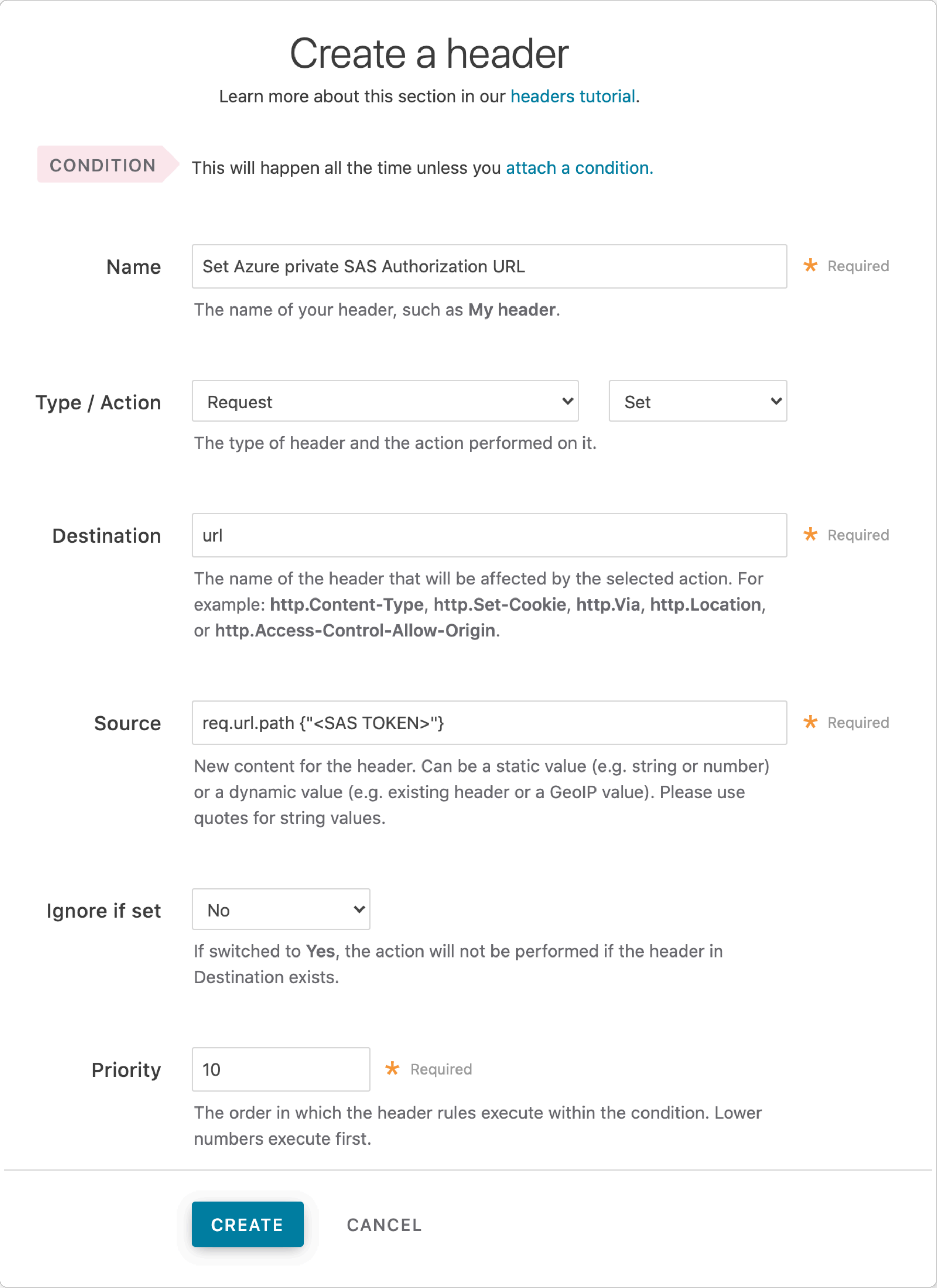
We provide a detailed look at the Shared Access Signature parameters below.

Signing the URL

Next, sign the access URL by creating an authorization header following the steps below.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.

3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Content**.
5. Click **Create header**.



Create a header
Learn more about this section in our headers tutorial.
CONDITION
This will happen all the time unless you attach a condition.

Name: Set Azure private SAS Authorization URL

* Required
The name of your header, such as My header.

Type / Action: Request / Set

The type of header and the action performed on it.

Destination: url

* Required
The name of the header that will be affected by the selected action. For example: http.Content-Type, http.Set-Cookie, http.Via, http.Location, or http.Access-Control-Allow-Origin.

Source: req.url.path {"<SAS TOKEN>"}

* Required
New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to Yes, the action will not be performed if the header in Destination exists.

Priority: 10

* Required
The order in which the header rules execute within the condition. Lower numbers execute first.
CREATE
CANCEL

6. Fill out the **Create a header** fields as follows:

- In the **Name** field, enter a meaningful name such as `Set Azure private SAS Authorization URL`.
- From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
- In the **Destination** field, enter `url`.
- In the **Source** field, enter `req.url.path {"<SAS TOKEN>"}` replacing `{"<SAS TOKEN>"}` with the token you obtained from the Azure Portal.
- From the **Ignore if set** menu, select **No**.
- In the **Priority** field, enter `10`.

7. Click **Create**.
8. Click **Activate** to deploy your configuration changes.

A detailed look at the Shared Access Signature parameters

Microsoft's Create a service SAS page provides more details on shared access signatures and how they are constructed.

Element	Description
`sv`	The `signedversion` field. This is required and should be whatever the Azure portal provided.
`ss`	The `signedservice` field. This is required and should be `b` for "blob storage."
`srt`	The `signedresourcetype` field. This is required and should be `o` for "object."
`sp`	The `signedpermissions` field. This is required and should be `r` for "read only."
`st`	The `signedstart` field. This is optional and specifies, in a UTC format compatible with ISO 8601, the time at which the shared access signature becomes valid. If omitted, the start time for this call is assumed to be the time when the storage service receives the request.
`se`	The `signedexpiry` field. This is required and specifies, in a UTC format compatible with ISO 8601, the time at which the shared access signature becomes invalid.
`spr`	The `signedprotocol` field. This is optional and specifies which HTTP protocol (`http` or `https`) the container should use for access. We recommend `https`.
`sig`	The `signature` field. This is required and should be whatever the Azure portal provided.

⚠ WARNING

Always keep track of the `se` expiry date. After it has passed, Fastly will not be able to access your private container.

TCP connection settings for improved performance

By default, Fastly keeps established TCP connections opened to your origin to improve performance. Azure's default behavior, however, closes idle connections. Specifically, the Azure Load Balancer's default behavior silently drops flows when the default idle timeout of a flow reaches four minutes. To ensure successful integration, we suggest tuning Azure in two ways:

- increase the Azure Load Balancer's TCP idle timeout setting
- configure Azure to send a bidirectional TCP Reset (RST packet) on idle timeout

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Oracle Cloud Storage

/en/guides/oracle-cloud-storage

Oracle Cloud Storage public and private buckets can be used as origins with Fastly.

Before you begin

Before you begin the setup and configuration steps required to use Oracle Cloud as an origin, keep in mind the following:

- You must have a valid Oracle Cloud account. Before you can create a new bucket and upload files to it for Fastly to use, you must first create an Oracle Cloud account at the Oracle website.
- Oracle Cloud implements both its own proprietary API and an S3 Compatible API. Currently, Fastly supports private buckets only via the S3 Compatible API.

Using Oracle Cloud Storage as an origin

To use Oracle Cloud Storage as an origin, follow the steps below.

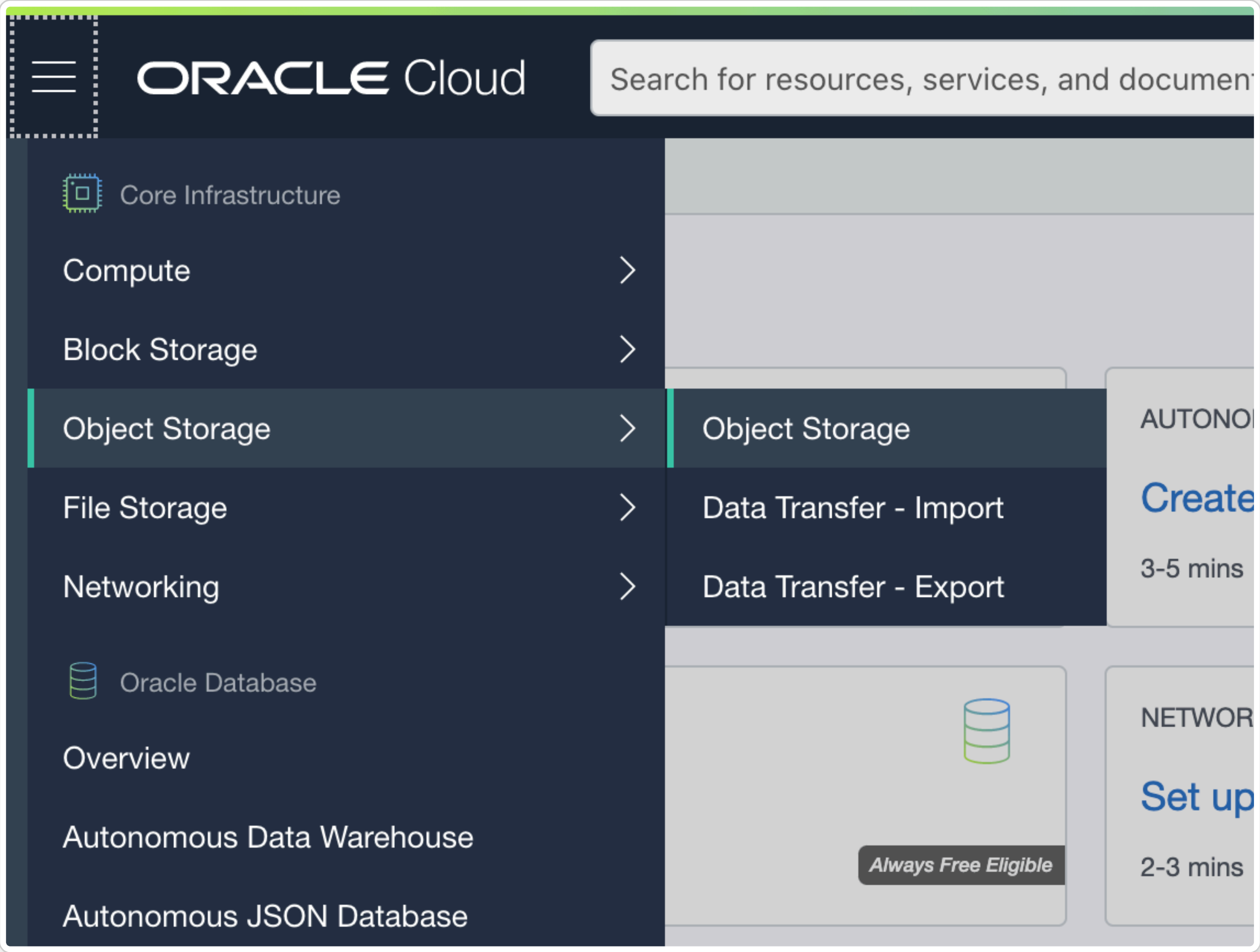
Creating a new bucket

Data in Oracle Cloud Storage is stored in buckets. Follow these steps to create a new bucket via the Oracle Cloud web interface.

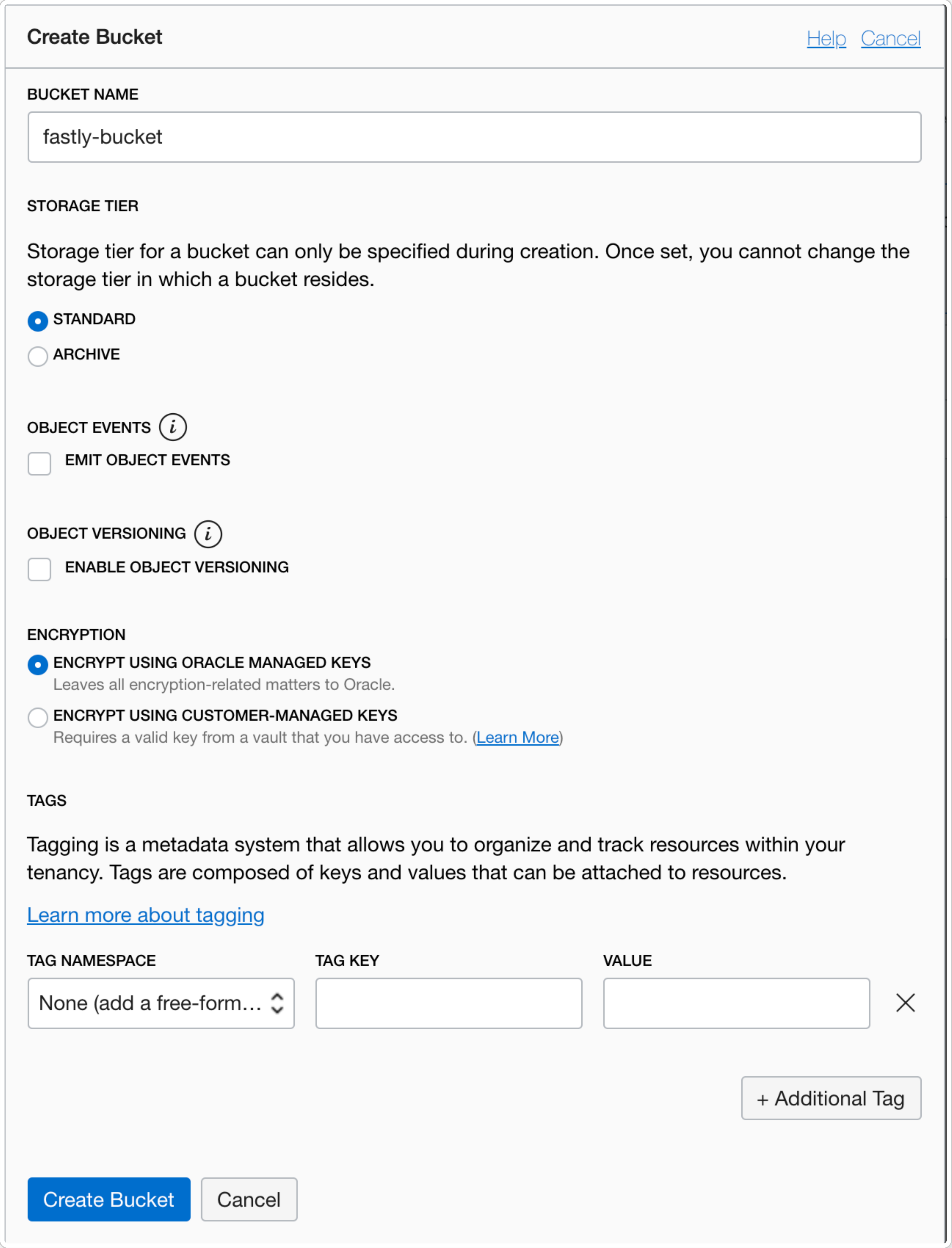
TIP

The Oracle Guide provides more details on how to create a bucket.

1. Log in to your Oracle account.
2. Open the navigation menu in the upper left and navigate to **Object Storage**, then select **Object Storage**.


ORACLE Cloud
Search for resources, services, and documen
Core Infrastructure
Compute
Block Storage
Object Storage
File Storage
Networking
Oracle Database
Overview
Autonomous Data Warehouse
Autonomous JSON Database
Object Storage
Data Transfer - Import
Data Transfer - Export
AUTONO
Create
3-5 mins
Always Free Eligible
NETWOR
Set up
2-3 mins

3. Select a compartment from the **Compartment** list on the left side of the page.
4. Click **Create a Bucket**.
5. In the **Bucket Name** field, enter a unique bucket name. Bucket names must be unique within the namespace and cannot be nested. The name can contain letters, numbers, dashes, and periods.



Create Bucket

BUCKET NAME

fastly-bucket

STORAGE TIER

Storage tier for a bucket can only be specified during creation. Once set, you cannot change the storage tier in which a bucket resides.

● STANDARD

○ ARCHIVE

OBJECT EVENTS

☐ EMIT OBJECT EVENTS

OBJECT VERSIONING

☐ ENABLE OBJECT VERSIONING

ENCRYPTION

● ENCRYPT USING ORACLE MANAGED KEYS
Leaves all encryption-related matters to Oracle.

○ ENCRYPT USING CUSTOMER-MANAGED KEYS
Requires a valid key from a vault that you have access to. (Learn More)

TAGS

Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources.

Learn more about tagging

TAG NAMESPACE	TAG KEY	VALUE
None (add a free-form…		

6. Click **Create a Bucket**. The new bucket appears in the list of buckets on the Oracle Cloud Storage Buckets page.

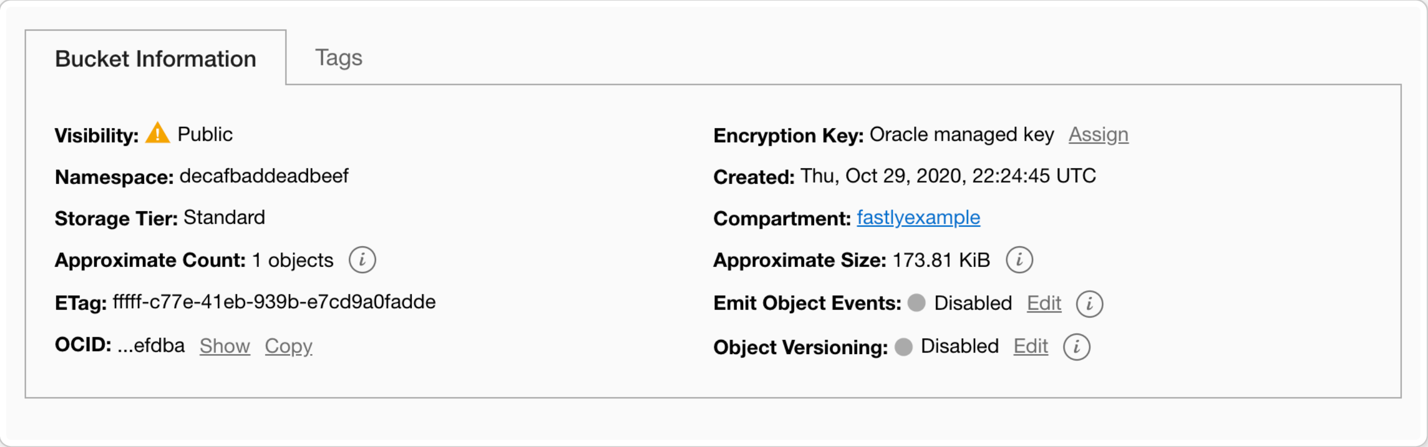
7. By default, new buckets are private. Click on three dots on the right side of the bucket and select **Edit Visibility**. Change the visibility to **Public** and deselect the **Allow users to list objects from this bucket** option.

8. Upload a file to the new bucket you just created.

Finding your bucket's namespace and hostname

To set up a Fastly service that interacts with your Oracle Cloud Storage, you will need to know the namespace identifier and hostname assigned to the bucket you created and uploaded files to.

To find your namespace, click on the bucket and examine the **Bucket Information** tab. In this example the namespace is `decafbaddeadbeef`.



Visibility: ⚠ Public

Namespace: decafbaddeadbeef

Storage Tier: Standard

Approximate Count: 1 objects

ETag: fffff-c77e-41eb-939b-e7cd9a0fadde

OCID: ...efdba Show Copy

Encryption Key: Oracle managed key Assign

Created: Thu, Oct 29, 2020, 22:24:45 UTC

Compartment: fastlyexample

Approximate Size: 173.81 KiB

Emit Object Events: ● Disabled Edit

Object Versioning: ● Disabled Edit

To determine your bucket's hostname:

- If you're using the native Oracle API then the hostname takes the form of `objectstorage.<region>.oraclecloud.com` (e.g `objectstorage.us-ashburn-1.oraclecloud.com`).
- If you're using the S3 Compatible API then the hostname takes the form of `<namespace id>.compat.objectstorage.<region>.oraclecloud.com` (e.g `decafbaddeadbeef.compat.objectstorage.us-ashburn-1.oraclecloud.com`).

Creating a new service

To create a new Fastly service, you must first create a new domain and then create a new host and edit it to accept traffic for Oracle Cloud Storage. Instructions to do this appear in our guide to creating a new service. While completing these instructions, keep the following in mind:

- When you create the new host, enter the Oracle bucket's hostname in the **Hosts** field on the **Origins** page. See Finding your bucket's namespace and hostname.
- When you edit the host details on the **Edit this host page**, confirm the Transport Layer Security (TLS) area information for your host. Specifically, make sure you do the following:
  - Secure the connection between Fastly and your origin.
  - Enter your bucket's hostname in the **Certificate hostname** field.
  - Select the checkbox to match the SNI hostname to the Certificate hostname (it appears under the SNI hostname field).
  - *(Optional)* Enable shielding by choosing the appropriate shielding location from the **Shielding** menu. When using Oracle Cloud Storage, this means you must choose a shielding location closest to the most

appropriate Oracle region.

- Decide whether or not you should specify an override host in the **Advanced options** area which is the same as your bucket hostname.

Using the Oracle Cloud API with public objects

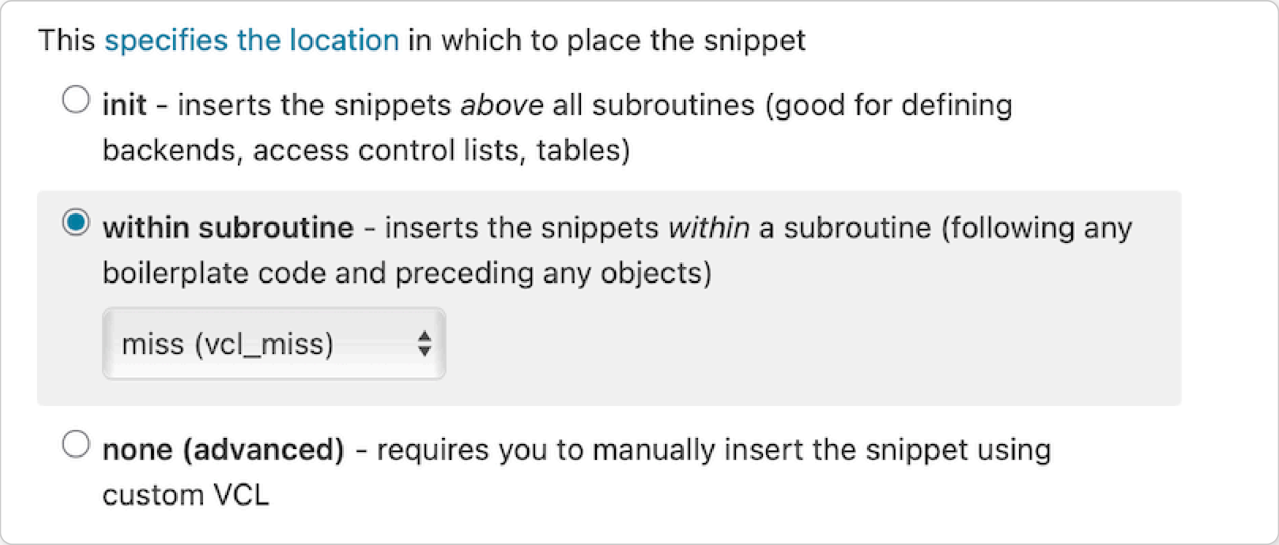
To use the Oracle Cloud API with public objects, you need to either create a new header, or a VCL Snippet. The purpose of the header or VCL snippet is to rewrite request URLs for your Oracle Cloud Storage instance.

Using a Header object

1. On your Fastly service's configuration page, click **Create header** to create a new header.
2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Rewrite Oracle Cloud Storage URL`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `url`.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `20`.
3. In the **Source** field, enter `"/n/<namespace id>/b/<bucket name>/o" req.url` (e.g., `"/n/decafbaddeadbeef/b/fastly-bucket/o" req.url`).
4. Click **Create**.
5. Click **Add a condition** next to the `Rewrite Oracle Cloud Storage URL` header.
6. Click **Create a new request condition**.
7. Fill out the condition fields as follows:
    - In the **Name** field, enter `Oracle Cloud Storage Shielding`.
    - In the **Apply if** field, enter `(req.method == "GET" && !req.backend.is_shield) {}`.
8. Click **Save and apply**.
9. Click **Activate** to deploy your configuration changes.

Using a VCL Snippet

1. Click **VCL Snippets** on your service's configuration page, then click **Create Snippet**.
2. In the Create a VCL Snippet page, enter a name for the snippet.
3. Select **within subroutine** to specify its placement, and **miss** as the subroutine type.



This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceding any objects)

miss (vcl_miss)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

4. Add the following code to the **VCL** field. Change the values of the `oracleNamespace` and `oracleBucket` variables to match your Oracle namespace and bucket.

```
declare local var.oracleNamespace STRING;
declare local var.oracleBucket STRING;
set var.oracleNamespace = "YOUR_ORACLE_NAMESPACE_ID";   # Change this value to your
set var.oracleBucket = "YOUR_ORACLE_BUCKET_NAME";   # Change this value to your own

if (req.method == "GET" && !req.backend.is_shield) {
  set bereq.url = "/n/" var.oracleNamespace "/b/" var.oracleBucket "/o/" bereq.url;
}
```

Using the S3 Compatible API with public objects

To use the S3 Compatible API with public objects you must create a new header, as explained below.

1. On your Fastly service's configuration page, click **Create header** to create a new header.
2. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter `Rewrite Oracle Cloud Storage URL`.
    - From the **Type** menu, select **Request**, and from the **Action** menu, select **Set**.
    - In the **Destination** field, enter `url`.
    - From the **Ignore if set** menu, select **No**.
    - In the **Priority** field, enter `20`.
3. In the **Source** field, enter `"/<bucket name>/" req.url` (e.g., `"/fastly-bucket/o" req.url`).
4. Click **Create**.
5. Click **Activate** to deploy your configuration changes.

Private Buckets

IMPORTANT

Currently, Fastly can only support private objects using the S3 Compatible API.

To use an Oracle Cloud Storage private bucket with Fastly you must implement version 4 of Amazon's header-based authentication. You can do this using custom VCL. Keep in mind the following:

- You will need an Oracle **Customer Secret Key** which consists of an **Access Key** and **Secret Key**.
- You must use path-based access. Virtual host-style access (for example, accessing a bucket as `<bucketname>.<namespace>.compat.objectstorage.<region>.oraclecloud.com`) is not supported.

The following table lists the information you need to obtain from Oracle Cloud Storage before starting.

Item	Description
Namespace	The namespace identifier assigned to your bucket (see Finding your bucket's namespace and hostname.
Bucket name	The name of your OCS bucket. When you download items from your bucket, this is the string listed in the URL path or hostname of each object.
Region	The OCS region code of the location where your bucket resides (e.g., `us-east-1`).
Access key	The OCS access key string for your account that has at least read permission on the bucket.
Secret key	The OCS secret access key paired with the access key above.

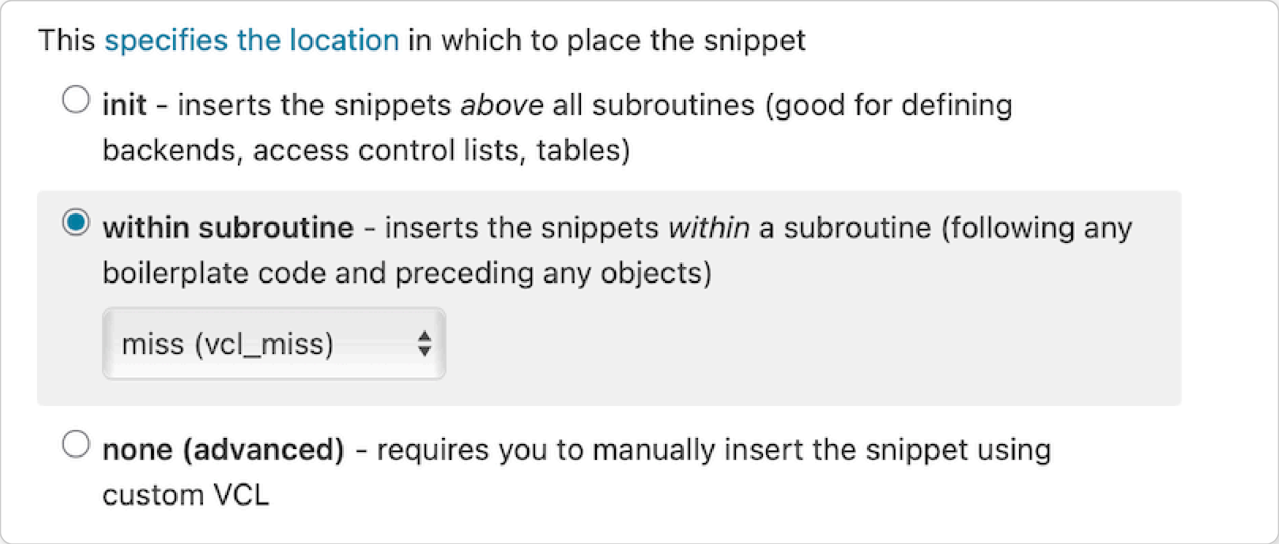
Once you have this information, you can configure your Fastly service to authenticate against your S3 bucket using header authentication by calculating the appropriate header value in VCL.

IMPORTANT

Consider leaving the **Override host** field for the origin blank in your service settings. This setting will override the host header from the snippets shown here and may invalidate the signature that authenticates the information being sent.

Start by creating a regular VCL snippet. Give it a meaningful name, such as `AWS protected origin`. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code (be sure to change specific values as noted to ones relevant to your own AWS bucket):

1. Click **VCL Snippets** on your service's configuration page, then click **Create Snippet**.
2. In the Create a VCL Snippet page give the snippet a meaningful name, such as `AWS protected origin`.
3. Select **within subroutine** to specify snippet placement, and **miss** as the subroutine type.



This specifies the location in which to place the snippet

○ **init** - inserts the snippets *above* all subroutines (good for defining backends, access control lists, tables)

● **within subroutine** - inserts the snippets *within* a subroutine (following any boilerplate code and preceding any objects)

miss (vcl_miss)

○ **none (advanced)** - requires you to manually insert the snippet using custom VCL

4. Add the following code to the **VCL** field. Be sure to change the values of the variables (`ocsNamespace`, `ocsAccessKey`, etc.) to match your Oracle environment.

```
declare local var.ocsNamespace STRING;
declare local var.ocsAccessKey STRING;
declare local var.ocsSecretKey STRING;
declare local var.ocsS3Bucket STRING;
declare local var.ocsRegion STRING;
declare local var.canonicalHeaders STRING;
declare local var.signedHeaders STRING;
declare local var.canonicalRequest STRING;
declare local var.canonicalQuery STRING;
declare local var.stringToSign STRING;
declare local var.dateStamp STRING;
declare local var.signature STRING;
declare local var.scope STRING;

set var.ocsNamespace = "YOUR_OCS_NAMESPACE"; # Change this value to your own data
set var.ocsAccessKey = "YOUR_OCS_ACCESS_KEY";   # Change this value to your own dat
set var.ocsSecretKey = "YOUR_OCS_SECRET_KEY";   # Change this value to your own dat
set var.ocsS3Bucket = "YOUR_OCS_BUCKET_NAME";   # Change this value to your own dat
set var.ocsRegion = "YOUR_OCS_REGION";   # Change this value to your own data

if (req.method == "GET" && !req.backend.is_shield) {

  set bereq.http.x-amz-content-sha256 = digest.hash_sha256("");
  set bereq.http.x-amz-date = strftime({"%Y%m%dT%H%M%SZ"}, now);
  set bereq.http.host = var.ocsNamespace ".compat.objectstorage." var.ocsRegion ".o
  set bereq.url = querystring.remove(bereq.url);
  set bereq.url = regsuball(urlencode(urldecode(bereq.url.path)), {"%2F"}, "/");
  set var.dateStamp = strftime({"%Y%m%d"}, now);
  set var.canonicalHeaders = ""
    "host:" bereq.http.host LF
    "x-amz-content-sha256:" bereq.http.x-amz-content-sha256 LF
    "x-amz-date:" bereq.http.x-amz-date LF
    ;
  set var.canonicalQuery = "";
  set var.signedHeaders = "host;x-amz-content-sha256;x-amz-date";
  set var.canonicalRequest = ""
    "GET" LF
    bereq.url.path LF
    var.canonicalQuery LF
    var.canonicalHeaders LF
    var.signedHeaders LF
    digest.hash_sha256("")
  ;

  set var.scope = var.dateStamp "/" var.ocsRegion "/s3/aws4_request";

  set var.stringToSign = ""
    "AWS4-HMAC-SHA256" LF
    bereq.http.x-amz-date LF
    var.scope LF
    regsub(digest.hash_sha256(var.canonicalRequest),"^0x", "")
  ;

  set var.signature = digest.awsv4_hmac(
    var.ocsSecretKey,
    var.dateStamp,
    var.ocsRegion,
    "s3",
    var.stringToSign
  );

  set bereq.http.Authorization = "AWS4-HMAC-SHA256 "
    "Credential=" var.ocsAccessKey "/" var.scope ", "
    "SignedHeaders=" var.signedHeaders ", "
    "Signature=" + regsub(var.signature,"^0x", "")
  ;
  unset bereq.http.Accept;
  unset bereq.http.Accept-Language;
  unset bereq.http.User-Agent;
  unset bereq.http.Fastly-Client-IP;
}
```

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Outbound data transfer from Azure

/en/guides/outbound-data-transfer-from-azure

Fastly has integrated local circuits with Microsoft Routing Preference Unmetered to create private connections to Azure. If using Azure as your origin, you can take advantage of the improved reliability, faster speeds, lower latencies, and higher security that this private local circuit offers over typical public internet connections.

To configure your Fastly service to use direct connectivity via this local circuit, enable shielding using your Azure service region to determine the shielding location. Additionally, configure the routing preference for your storage account based on the service region and shielding location combination. Our documentation lists the available shielding locations for the routing preference you've chosen.

NOTE

You can configure a routing preference after the storage account is created.

Once you configure your Fastly service and Azure storage account correctly, outbound data transfers from the appropriate Azure regions to Fastly will use this local circuit. Because Fastly has purchased this local circuit from Microsoft, Microsoft does not apply outbound data transfer rates to traffic traveling over it. Use of this local circuit should work with Azure services like Container Instances, Functions, and Media Services, as well as Blob Storage.

WARNING

We encourage you to use Microsoft Azure's Billing Tools to monitor traffic on Microsoft Routing Preference Unmetered for the purpose of zero-rated egress. Despite this connection to Fastly's services being in place, in certain circumstances your data may egress from Azure over the public internet. In such cases, your traffic to the public internet will be metered according to your commercial arrangement with Microsoft.

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

	Storj DCS Object Storage
🔗	/en/guides/storj-dcs-object-storage

Storj DCS can be used as an origin for public and private Storj buckets via the Storj DCS S3 Gateway. Built on the Storj Network, Storj DCS is a decentralized object storage service that is S3 compatible and end-to-end encrypted by default.

Prerequisites

Before adding Storj DCS as an origin for Fastly services, you will need to create a Storj DCS account, project and access credentials, and a bucket that will serve as your origin.

Using Storj DCS as an origin

To use Storj DCS as an origin and make your Storj bucket available through Fastly via the Storj DCS S3 Gateway, follow the steps below.

Creating a new service

Follow the instructions for creating a new service.

1. When you create the new domain and the new host:
    - In the **Domain Name** field on the **Create a domain** page, enter the hostname you want to use as the URL (e.g., `cdn.example.com`).
    - In the **Hosts** field on the **Origins** page, enter the IP address or hostname of your Storj DCS Gateway Endpoint using the format `<BUCKET>.gateway.<REGION>.storjshare.io` including your bucket (e.g., `origin.gateway.us1.storjshare.io`).
2. When you edit the host details on the **Edit this host page**:
    - In the **Name** field, enter any descriptive name for your service if you haven't already done so.
    - In the **Address** field, ensure you've entered the IP address or hostname of your Storj DCS Gateway Endpoint. You entered this information during host creation.
3. When you edit the Transport Layer Security (TLS) area information for your host:
    - If you've set up TLS for your Storj DCS S3 Gateway, leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
    - Under the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname**. The address you entered during host creation appears.
    - In the **Certificate hostname** field, enter the IP address or hostname of your Storj DCS S3 Gateway.

Testing your results

By default, we create a DNS mapping called `yourdomain.global.prod.fastly.net`. In the example above, it would be `cdn.example.com.global.prod.fastly.net`. Create a DNS alias for the domain name you specified (e.g., CNAME `cdn.example.com` to `global-nossl.fastly.net`).

Fastly will cache any content without an explicit `Cache-Control` header for 1 hour. You can verify whether you are sending any cache headers using curl. For example:

```
$ curl -I https://cdn.example.com

Accept-Ranges: bytes
Content-Length: 250
Content-Type: application/xml
Server: MinIO/DEVELOPMENT.GOGET
Vary: Origin
Date: Wed, 07 Oct 2020 02:31:27 GMT
```

In this example, no Cache-Control headers are set so the default TTL will be applied.

Enhanced cache control

If you need more control over how different types of assets are cached (e.g., JavaScript files, images), check out our documentation on cache freshness.

Using a Storj DCS bucket for origin hosting

To use a Storj DCS S3 Gateway as an origin with Fastly, you must implement version 4 of Amazon's header-based authentication. You can do this using custom VCL. Start by obtaining the following information from AWS:

Item	Description
Bucket name	The name of your private bucket. When you download items from your bucket, this is the string listed in the URL path or hostname of each object.
Access key	The access key string associated with a Storj DCS Access Grant that has at least read permissions on the bucket.
Secret key	The secret access key paired with the access key above.

Once you have this information, you can configure your Fastly service to authenticate against your private bucket using header authentication by calculating the appropriate header value in VCL.

IMPORTANT

Consider leaving the **Override host** field for the origin blank in your service settings. This setting will override the host header from the snippets shown here and may invalidate the signature that authenticates the information being sent.

Start by creating a regular VCL snippet. Give it a meaningful name, such as `Storj DCS Origin`. When you create the snippet, select **within subroutine** to specify its placement and choose **miss** as the subroutine type. Then, populate the **VCL** field with the following code (be sure to change specific values as noted to ones relevant to your own bucket):

```
declare local var.accessKey STRING;
declare local var.secretKey STRING;
declare local var.storjBucket STRING;
declare local var.storjGateway STRING;
declare local var.region STRING;
declare local var.canonicalHeaders STRING;
declare local var.signedHeaders STRING;
declare local var.canonicalRequest STRING;
declare local var.canonicalQuery STRING;
declare local var.stringToSign STRING;
declare local var.dateStamp STRING;
declare local var.signature STRING;
declare local var.scope STRING;


set var.accessKey = "YOUR_ACCESS_KEY";   # Change this value to your own data
set var.secretKey = "YOUR_SECRET_KEY";   # Change this value to your own data
set var.storjBucket = "YOUR_BUCKET_NAME";   # Change this value to your own data
set var.storjGateway = "STORJ-DCS_GATEWAY";   # Change this value to your own data
set var.region = "decentralized";


if (req.method == "GET" && !req.backend.is_shield) {

  set bereq.http.x-amz-content-sha256 = digest.hash_sha256("");
  set bereq.http.x-amz-date = strftime({"%Y%m%dT%H%M%SZ"}, now);
  set bereq.http.host = var.storjBucket "." var.storjGateway;
  set bereq.url = querystring.remove(bereq.url);
  set bereq.url = regsuball(urlencode(urldecode(bereq.url.path)), {"%2F"}, "/");
  set var.dateStamp = strftime({"%Y%m%d"}, now);
  set var.canonicalHeaders = ""
    "host:" bereq.http.host LF
    "x-amz-content-sha256:" bereq.http.x-amz-content-sha256 LF
    "x-amz-date:" bereq.http.x-amz-date LF
  ;
  set var.canonicalQuery = "";
  set var.signedHeaders = "host;x-amz-content-sha256;x-amz-date";
  set var.canonicalRequest = ""
    "GET" LF
    bereq.url.path LF
    var.canonicalQuery LF
    var.canonicalHeaders LF
    var.signedHeaders LF
    digest.hash_sha256("")
  ;

  set var.scope = var.dateStamp "/" var.region "/s3/aws4_request";


  set var.stringToSign = ""
    "AWS4-HMAC-SHA256" LF
    bereq.http.x-amz-date LF
    var.scope LF
    regsub(digest.hash_sha256(var.canonicalRequest),"^0x", "")
  ;

  set var.signature = digest.awsv4_hmac(
    var.secretKey,
    var.dateStamp,
    var.region,
    "s3",
    var.stringToSign
  );


 set bereq.http.Authorization = "AWS4-HMAC-SHA256 "
    "Credential=" var.accessKey "/" var.scope ", "
    "SignedHeaders=" var.signedHeaders ", "
    "Signature=" + regsub(var.signature,"^0x", "")
  ;

  unset bereq.http.Accept;
  unset bereq.http.Accept-Language;
  unset bereq.http.User-Agent;
  unset bereq.http.Fastly-Client-IP;
}
```

* * *

Wasabi Hot Cloud Storage

🔗 /en/guides/wasabi-hot-cloud-storage

Wasabi Hot Cloud Storage public and private buckets can be used as origins with Fastly.

Using Wasabi as an origin

To make your Wasabi Hot Cloud Storage bucket available through Fastly, follow the steps below.

Creating a new service

Follow the instructions for creating a new service.

1. When you create the new domain and the new backend host:
    - In the **Domain Name** field on the **Create a domain** page, enter the hostname you want to use as the URL (e.g., `cdn.example.com`).
    - In the **Hosts** field on the **Origins** page, enter the appropriate address for your Wasabi Hot Cloud Storage bucket's region. For the us-east-1 region, enter `<BUCKET>.s3.wasabisys.com`. For all other regions, enter `<BUCKET>.s3.<REGION>.wasabisys.com`, replacing `<REGION>` as appropriate (e.g., `<BUCKET>.s3.eu-central-1.wasabisys.com`).
2. When you edit the host details on the **Edit this host page**:
    - In the **Name** field, enter any descriptive name for your service if you haven't already done so.
    - In the **Address** field, ensure you've entered the appropriate address for your Host (e.g., `<BUCKET>.s3.wasabisys.com`). You entered this information during Host creation.
3. When you edit the Transport Layer Security (TLS) area information for your host:
    - Leave the **Enable TLS?** default set to **Yes** to secure the connection between Fastly and your origin.
    - In the **Certificate hostname** field, enter the same address that appears in the Address field (e.g., `<BUCKET>.s3.wasabisys.com`).
    - Under the **SNI hostname** field, select the checkbox to **Match the SNI hostname to the Certificate hostname**. The address you entered during Host creation appears.
4. In the **Override host** field, enter an appropriate address for your Host (e.g., `<BUCKET>.s3.wasabisys.com`). You entered this information during Host creation.
5. From the **Shielding** menu below the TLS area, select an appropriate shielding location. For more information about this setting and which locations to select, see our enabling shielding information.

Enabling shielding

We strongly encourage you to enable shielding for your origin server. Wasabi imposes soft caps on free egress. Without shielding enabled, Fastly will request the same objects from all Fastly edge POPs instead of just one, which may not follow Wasabi's free egress guidelines.

When you select a shielding location from the **Shielding** menu, choose the location appropriate for your Wasabi Hot Cloud Storage bucket as follows:

Wasabi bucket region	Shielding location
eu-central-1	Amsterdam, NL
us-east-1	Ashburn, VA
us-west-1	Seattle, WA

Read our guidance on choosing a shield location for more information.

Testing your results

By default, we create a DNS mapping called **yourdomain.global.prod.fastly.net**. In the example above, it would be `cdn.example.com.global.prod.fastly.net`. Create a DNS alias for the domain name you specified (e.g., CNAME `cdn.example.com` to `global-nossl.fastly.net`).

Fastly will cache any content without an explicit `Cache-Control` header for 1 hour. You can verify whether you are sending any cache headers using curl. For example:

```
$ curl -I opscode-full-stack.s3.wasabisys.com

HTTP/1.1 200 OK
x-amz-id-2: ZpzRp7IWc6MJ8NtDEFGH12QBdk2CM1+RzVOngQbhMp2f2ZyalkFsZd4qPaLMkSlh
x-amz-request-id: ABV5032583242618
Date: Fri, 18 Mar 2012 17:15:38 GMT
Content-Type: application/xml
Transfer-Encoding: chunked
```

In this example, no Cache-Control headers are set so the default TTL will be applied.

Enhancing cache control

If you need more control over how different types of assets are cached (e.g., JavaScript files, images), check out our documentation on cache freshness.

Using private Wasabi Hot Cloud Storage buckets

To use a Wasabi Hot Cloud Storage private bucket with Fastly, you must implement version 4 of Amazon's header-based authentication. You can do this using custom VCL and following the instructions below.

Before you begin

Make your Wasabi Hot Cloud Storage bucket available to Fastly. Be sure you've set your origin to port 443. This needs to be done before implementing header-based authentication with the instructions below.

Gathering Wasabi information

Start by obtaining the following information from Wasabi:

Item	Description
Bucket Name	The unique name of your Wasabi Hot Cloud Storage bucket. When you download items from your bucket, this is the string listed in the URL path or hostname of each object (e.g., `widget-project`).
Region	The Wasabi region code of the location where your bucket resides (e.g., `us-east-1`).
Access Key ID	The Wasabi access key ID string for an IAM account that has at least read permission on the bucket.
Secret Access Key	The Wasabi secret access key paired with the access key above.

You should review the user access separation document to make sure you are not inadvertently exposing files you didn't intend e.g. allowing ListBucket operations etc. Alternatively you can use the VCL snippet from the bottom of the document to block bucket listing.

Once you have this information, you can configure your Fastly service to authenticate against your Wasabi bucket using header authentication by calculating the appropriate header value in VCL.

Creating a VCL snippet for authentication

Create a regular VCL snippet.

- In the **Name** field, enter `Wasabi protected origin`.
- In the **Type (placement of the snippet)** field, select **within subroutine** then choose `miss (vcl_miss)`.
- In the **VCL** field, place the following code (be sure to change specific values as noted to ones relevant to your own Wasabi bucket):

```
if ( req.request == "GET" && req.backend.is_origin) {

  declare local var.wasabiAccessKey STRING;
  declare local var.wasabiSecretKey STRING;
  declare local var.wasabiBucket STRING;
  declare local var.wasabiRegion STRING;
  declare local var.canonicalHeaders STRING;
  declare local var.signedHeaders STRING;
  declare local var.canonicalRequest STRING;
  declare local var.canonicalQuery STRING;
  declare local var.stringToSign STRING;
  declare local var.dateStamp STRING;
  declare local var.signature STRING;
  declare local var.scope STRING;

  # Supply your own credentials
  set var.wasabiAccessKey = "YOUR_BUCKET_ACCESS_KEY";   # Change this value to your own
  set var.wasabiSecretKey = "YOUR_BUCKET_SECRET";       # Change this value to your own
  set var.wasabiBucket = "YOUR_BUCKET_NAME";            # Change this value to your own
  set var.wasabiRegion = "YOUR_BUCKET_REGION";          # Change this value to your own

  set bereq.http.x-amz-content-sha256 = digest.hash_sha256("");
  set bereq.http.x-amz-date = strftime({"%Y%m%dT%H%M%SZ"}, now);
  set bereq.http.host = var.wasabiBucket ".s3." var.wasabiRegion ".wasabisys.com";
  set bereq.url = querystring.remove(bereq.url);
  set var.dateStamp = strftime({"%Y%m%d"}, now);
  set var.canonicalHeaders = ""
    "host:" bereq.http.host LF
    "x-amz-content-sha256:" bereq.http.x-amz-content-sha256 LF
    "x-amz-date:" bereq.http.x-amz-date LF
  ;
  set var.canonicalQuery = "";
  set var.signedHeaders = "host;x-amz-content-sha256;x-amz-date";
  set var.canonicalRequest = ""
    "GET" LF
    bereq.url.path LF
    var.canonicalQuery LF
    var.canonicalHeaders LF
    var.signedHeaders LF
    digest.hash_sha256("")
  ;

  set var.scope = var.dateStamp "/" var.wasabiRegion "/s3/aws4_request";

  set var.stringToSign = ""
    "AWS4-HMAC-SHA256" LF
    bereq.http.x-amz-date LF
    var.scope LF
    regsub(digest.hash_sha256(var.canonicalRequest),"^0x", "")
  ;

  set var.signature = digest.awsv4_hmac(
    var.wasabiSecretKey,
    var.dateStamp,
    var.wasabiRegion,
    "s3",
    var.stringToSign
  );

  set bereq.http.Authorization = "AWS4-HMAC-SHA256 "
    "Credential=" var.wasabiAccessKey "/" var.scope ", "
  "SignedHeaders=" var.signedHeaders ", "
    "Signature=" + regsub(var.signature,"^0x", "")
  ;
  unset bereq.http.Accept;
  unset bereq.http.Accept-Language;
  unset bereq.http.User-Agent;
  unset bereq.http.Fastly-Client-IP;
  }
```

Creating a VCL snippet to remove added response headers

You may also remove the headers that Wasabi adds to the response. Do this by creating another VCL snippet.

- In the **Name** field, enter `Strip Wasabi response headers`.
- In the **Type (placement of the snippet)** field, select **within subroutine** then select `deliver (vcl_deliver)`.
- In the **VCL** field, place the following code:

```
if ( !req.http.Fastly-Debug ) {
  unset resp.http.x-amz-id-2;
  unset resp.http.x-amz-request-id;
  unset resp.http.server;
}
```

Blocking directory listing

If you don't set up correct IAM privileges you may allow users to list contents of your bucket folders. If you want to disallow that on Fastly please create following snippet

- In the **Name** field, enter `Disallow bucket listing`.
- In the **Type (placement of the snippet)** field, select **within subroutine** then select `recv (vcl_recv)`.
- In the **VCL** field, place the following code:

```
if ( req.url.path ~ "/$" ) {
  error 403;
}
```

This article describes an integration with a service provided by a third party. Read our note on integrations for details.

* * *

Security guides

Category: Access Control Lists

These articles describe how to restrict access to resources by allowing or blocking IP addresses with access control lists (ACLs).

About ACLs

🔗 /en/guides/about-acls

Malicious actors can present themselves in a variety of ways on the internet. Automated tools can scrape information from your website, bots can probe your application for vulnerabilities, and hackers can exploit them. Using access control lists (ACLs) at the edge can help prevent the offending IP addresses they use from ever accessing your information resources.

When ACLs can be useful

Access control lists at the edge might be useful for:

- E-commerce companies preventing scraping from certain IP ranges
- Offices restricting access to their administrative portals
- Advertising technology companies blocking bad-actors at the edge
- Mobile applications accepting only calls from specific proxies or IP ranges
- System administrators restricting access to groups of backends from an office IP address or subnet range

How ACLs work

ACLs have two parts: an ACL container and the ACL entries within it. In combination, containers and entries allow you to store a list of permissions that Varnish will use to grant or restrict access to URLs within your services.

Once you attach an ACL container to a version of your service and that service is activated, the data in the container (the ACL entries) becomes *versionless*. This means that once your service is activated, any further changes to the data within, such as the addition of ACL entries, will become effective immediately.

Ways to create ACLs

To create an ACL at the edge and use it within your service, start by creating an empty ACL container and then add its entries in a working version of a service that's unlocked and not yet activated. You can create ACLs in several ways:

- **Via Fastly's web interface or API:** You can create your ACLs at the edge via the Fastly web interface or via the Fastly API. We recommend these options for most configurations that integrate websites or applications with an ACL at the edge.
- **Using custom VCL:** You can manually create an ACL using VCL. We recommend this option only if you have simple access control requirements and can hardcode a few IP addresses in your VCL. Manually created ACLs are versioned with your services and any changes to the ACL will require changes to your VCL.

Example ACL use

After you've used the Fastly API to create an ACL and add ACL entries, the VCL for the ACLs and ACL entries will be automatically generated, as shown below. For example, this VCL shows an ACL called `office_ip_ranges` has been created:

```
# This VCL is automatically generated when you create an ACL container and entries
# using the Fastly API. In this example, the ACL name is office_ip_ranges.
acl office_ip_ranges {
  "192.0.2.0"/24;                               # internal office
  "198.51.100.4";                               # remote VPN office
  "2001:db8:ffff:ffff:ffff:ffff:ffff:ffff";     # ipv6 address remote
}
```

Once created, you can add logic to interact with your ACL at the edge by uploading custom VCL. You could use the `office_ip_ranges` ACL as an allow list by uploading the following custom VCL:

```
sub vcl_recv {
  # block all requests to Admin pages from IP addresses not in office_ip_ranges
  if (req.url ~ "^/admin" && client.ip !~ office_ip_ranges) {
    error 403 "Forbidden";
  }
}
```

With this VCL, access to `/admin` is denied for everyone by default, but the IP addresses listed in the ACL are allowed to access `/admin` without restriction.

> TIP
>
> Because ACL entries have a boolean option for negation, you can specify whether or not an IP address is allowed (false or `0`) or blocked (true or `1`).

Limitations

When working with ACL containers and entries specifically, remember the following:

- **ACL entry changes via the API don't appear in the event logs.** If you use the API to add, update, or remove an ACL entry, there will be no record of it in the audit log or event log. The only record of a change will exist when you compare service versions and view the exact point at which the ACL was associated with the service version in the first place.
- **ACL entry deletions are permanent.** ACL entries are versionless. This means that if you delete an entry within an ACL container, that entry is permanently removed from all service versions and cannot be recovered.
- **ACL containers are limited to 1000 ACL entries.** If you find your containers approaching this entry limit, contact support. We may be able to help you figure out an even more efficient way to do things with your ACLs at the edge.
- **Deleted ACL containers are only removed from the service version you're editing.** ACL containers are tied to versions of services, which can be cloned and reverted. When you delete an ACL container, only the configuration of the service version you're editing will be affected. We remove the ACL entries inside a container but only for the specific service version you're editing. The ACL entries themselves are not deleted from the ACL in earlier versions of your service's configuration. This allows you to revert your configuration to a previous version in as few steps as possible.

When creating and manipulating ACLs at the edge, keep the following limitations in mind as you develop your service configurations:

- **ACLs created with custom VCL are always versioned.** ACLs created with custom VCL are always tied to a service and require a new service version each time they are updated in any way. This is true for both the ACLs created using custom VCL and for any logic created to interact with those ACLs.
- **ACLs created with custom VCL cannot be manipulated using the API.** If you create an ACL using custom VCL, that ACL must always be manipulated via custom VCL and can never be manipulated using the Fastly API. ACLs created using the API, however, can be manipulated both using the API and custom VCL.

* * *

Manually creating access control lists

/en/guides/manually-creating-access-control-lists

Varnish allows you to use access control lists (ACLs), a feature that enables fast matching of a client's IP address against a list of defined IP addresses. An ACL looks like this:

```
# Who is allowed access ...
acl local {
  "localhost";
  "192.0.2.0"/24; /* and everyone on the local network */
  ! "192.0.2.1"/32; /* except for the dial-in router */
}
```

Defining an ACL

Using ACLs requires you to create and add custom VCL to Fastly's boilerplate VCL. To define an ACL in your Fastly configuration:

1. Read about how to mix and match custom VCL with Fastly VCL.
2. Create a custom VCL file with your ACL definitions included in the appropriate location. Use the example shown below as a guide. You can reference the ACL in your configuration (`vcl_recv`) using a match operation that can be located above or below `#FASTLY recv`. The placement only matters for the order of operations within Varnish's execution of your configuration.

```
# If you are using the "include" keyword
include "myACL1.vcl";

# And/or if you are using an actual ACL block
acl local {
  "localhost";
  "192.0.2.0"/24; /* and everyone on the local network */
  ! "192.0.2.1"/32; /* except for the dial-in router */
}

sub vcl_recv {
  # block any requests to Admin pages not from local IPs
  if (req.url ~ "^/admin" && req.http.Fastly-Client-IP !~ local) {
    error 403 "Forbidden";
  }
}
```

3. Upload the file in the Varnish Configuration area of your service.

* * *

Using the IP block list

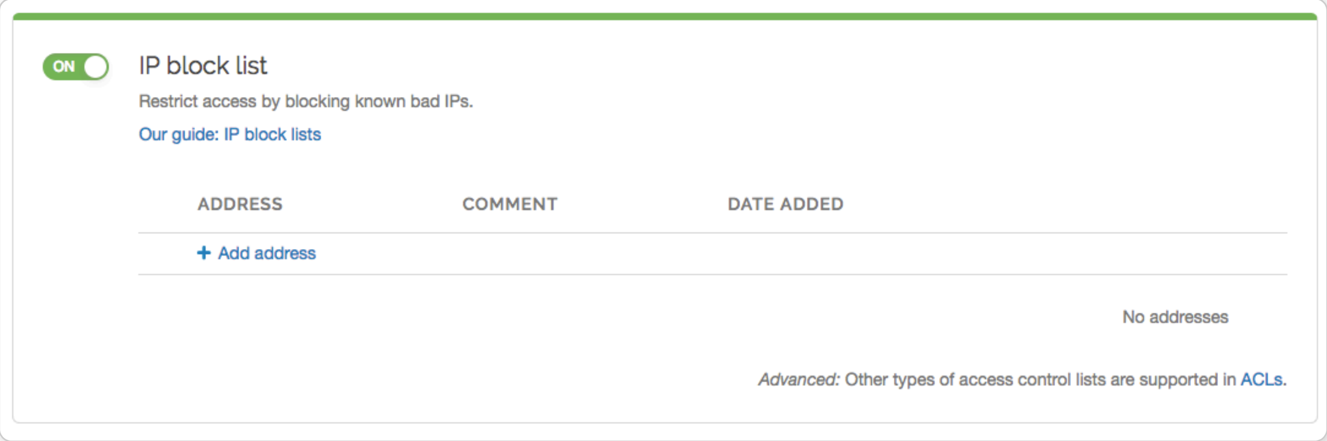
/en/guides/using-the-ip-block-list

You can prevent specific IP addresses from accessing your service by adding them to a block list. Enabling this feature creates a condition and response that returns a 403 error to anyone trying to access the service from a blocked IP address. You can use this feature to prevent bad actors from interfering with the operation of your web application.

Enabling the IP block list

To enable the IP block list, follow the steps below:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.
5. Click the **IP block list** switch to **On**.



ON

IP block list

Restrict access by blocking known bad IPs.

Our guide: IP block lists

ADDRESS	COMMENT	DATE ADDED
+ Add address		

No addresses

Advanced: Other types of access control lists are supported in ACLs.

6. Click **Activate** to deploy your configuration changes.

1. Open a terminal application and enter the following command to create an ACL:

```
$ curl -i  -X POST "https://api.fastly.com/service/[service_id]/version/[service_
```

2. Enter the following command to create a condition:

```
$ curl -i  -X POST "https://api.fastly.com/service/[service_id]/version/[service_
```

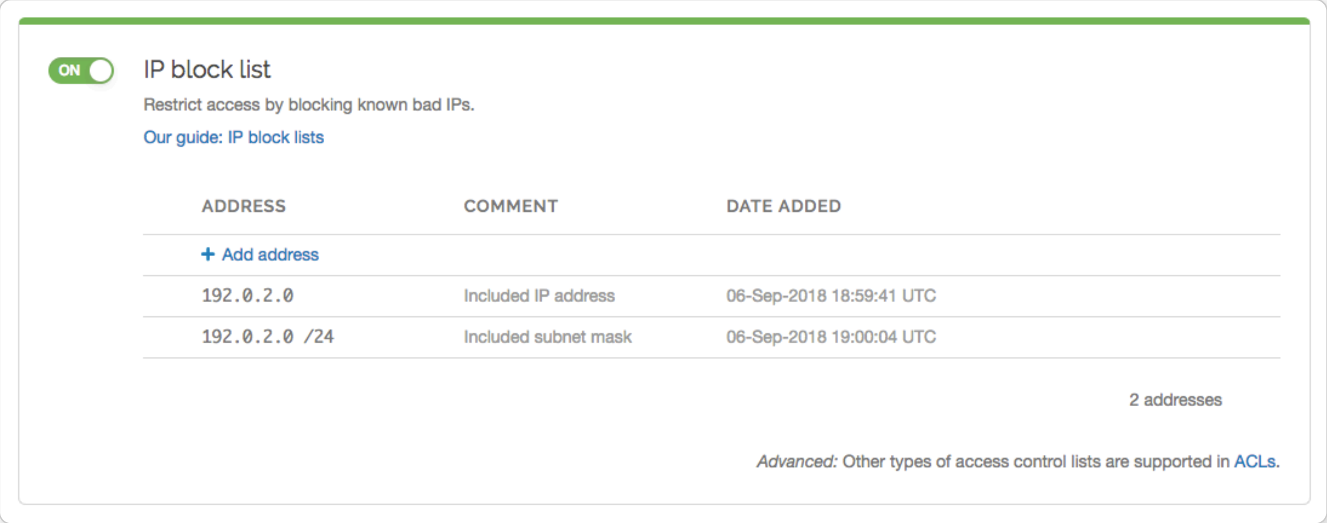
3. Enter the following command create a response object:

```
$ curl -i  -X POST "https://api.fastly.com/service/[service_id]/version/[service_
```

Blocking an IP address

To block an IP address, follow the steps below:

1. Click **Add address**.
2. In the **Address** field, enter an IP address or subnet mask (a range of IP addresses) to block for this service. To add an exception for an IP address, use an exclamation point (for example, use `!192.0.2.0` or `!192.0.2.0/24`).
3. *(Optional)* In the **Comment** field, enter a comment that describes the IP address or subnet mask.
4. Click **Add**. The IP address or subnet mask appears in the list. This addition will become effective immediately.



IP block list

Restrict access by blocking known bad IPs.

Our guide: IP block lists

ADDRESS	COMMENT	DATE ADDED
+ Add address		
192.0.2.0	Included IP address	06-Sep-2018 18:59:41 UTC
192.0.2.0 /24	Included subnet mask	06-Sep-2018 19:00:04 UTC

2 addresses

Advanced: Other types of access control lists are supported in ACLs.

1. Find the ID of the IP block list ACL. Open a terminal application and enter the following command to list the ACLs on your service:

```
$ curl -i  "https://api.fastly.com/service/[service_id]/version/[service_version]
```

Find the ID of the IP block list ACL in the output:

```
[{"updated_at":"2023-11-01T20:46:51Z","version":"3","deleted_at":null,"created_at":"2
```

2. Using the ACL ID, enter the following command to block an IP address by creating a new ACL entry:

```
$ curl -i  -X POST "https://api.fastly.com/service/[service_id]/acl/[acl_id]/entr
```

Editing a blocked IP address

You can edit a blocked IP address or subnet mask at any time. To edit an IP address or a subnet mask, follow the steps below:

1. Find the IP block list associated with your service in which the associated IP addresses or subnet masks appear. Because these entries are versionless, the service version you choose doesn't matter. Choose the one that makes the most sense to you.
2. In the IP block list area, hover your cursor over an entry, then click the pencil that appears.
3. Edit the IP address, subnet mask, or comment as necessary.
4. Click **Save**. The changes you make will be immediately applied to your configuration. If your IP block list has already been associated with a deployed service version, those changes will happen live.

1. Find the ID of the IP block list ACL. Open a terminal application and enter the following command to list the ACLs on your service:

   ```
   $ curl -i  "https://api.fastly.com/service/[service_id]/version/[service_version]
   ```

   Find the ID of the IP block list ACL in the output:

   ```
   [{"updated_at":"2023-11-01T20:46:51Z","version":"3","deleted_at":null,"created_at":"2
   ```

2. Using the ACL ID, enter the following command to list the ACL entries on your service:

   ```
   $ curl -i  "https://api.fastly.com/service/[service_id]/acl/[acl_id]/entries?dire
   ```

   Find the ACL entry ID of the IP address you want to update in the output.

   ```
   [{"acl_id":"[acl_id]","negated":"0","service_id":"[service_id]","ip":"192.168.1.1","c
   ```

3. Using the ACL entry ID of the IP address you want to update, enter the following command to update the IP address:

   ```
   $ curl -i  -X PATCH "https://api.fastly.com/service/[service_id]/acl/[acl_id]/ent
   ```

Deleting an IP block list entry

You can delete individual entries in the IP block list at any time. To delete an IP address or subnet mask that was created via the web interface:

1. Find the IP block list associated with your service in which the associated IP addresses or subnet masks appear. Because these entries are versionless, the service version you choose doesn't matter.
2. In the IP block list area, hover your cursor over an entry, then click the trash that appears.
3. Click **Confirm and delete**.

1. Find the ID of the IP block list ACL. Open a terminal application and enter the following command to list the ACLs on your service:

```
$ curl -i  "https://api.fastly.com/service/[service_id]/version/[service_version]
```

Find the ID of the IP block list ACL in the output:

```
[{"updated_at":"2023-11-01T20:46:51Z","version":"3","deleted_at":null,"created_at":"2
```

2. Using the ACL ID, enter the following command to list the ACL entries on your service:

```
$ curl -i  "https://api.fastly.com/service/[service_id]/acl/[acl_id]/entries?dire
```

Find the ACL entry ID of the IP address you want to delete in the output.

```
[{"acl_id":"[acl_id]","negated":"0","service_id":"[service_id]","ip":"192.168.1.1","c
```

3. Using the ACL entry ID of the IP address you want to delete, enter the following command to update the IP address:

```
$ curl -i  -X DELETE "https://api.fastly.com/service/[service_id]/acl/[acl_id]/en
```

Disabling the IP block list

The IP block list and its associated entries can be disabled in any unlocked service version. To disable the IP block list, follow the steps below:

1. Find the IP block list associated with an unlocked version of your service.
2. Click the **IP block list** switch to **Off**.
3. Click **Yes**. This disables the block list and deletes all associated entries.
4. Click **Activate** to deploy your configuration changes.

Creating other ACL types

If you need other types of ACLs, you'll need to create them in the Data page of the web interface.

* * *

Working with ACLs using the web interface

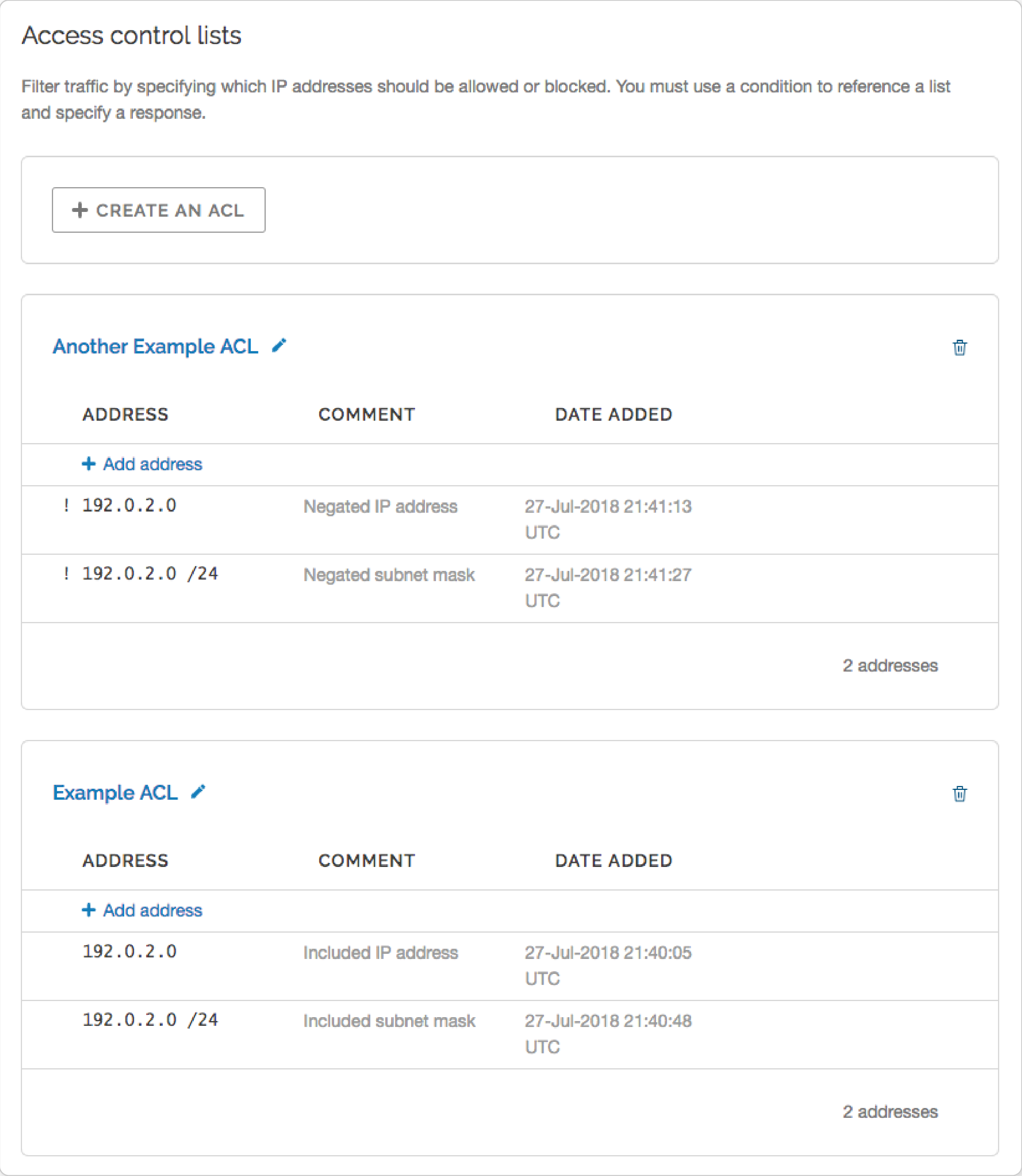
/en/guides/working-with-acls-using-the-web-interface

Access control lists (ACLs) allow you to store a list of permissions that Varnish will use to grant or restrict access to URLs within a service. You can use the web interface to add, remove, and update ACLs.

Viewing ACLs

To view an ACL, navigate to the ACL management area of your service:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Configuration** and then select **View Active**.
4. From the service version menu, select an appropriate service version.
5. Click **Data**. The Data page appears. Existing ACLs, if any, associated with the currently selected service version appear in the Access control lists area.



Access control lists

Filter traffic by specifying which IP addresses should be allowed or blocked. You must use a condition to reference a list and specify a response.

Another Example ACL

ADDRESS	COMMENT	DATE ADDED
+ Add address		
! 192.0.2.0	Negated IP address	27-Jul-2018 21:41:13 UTC
! 192.0.2.0 /24	Negated subnet mask	27-Jul-2018 21:41:27 UTC

2 addresses

Example ACL

ADDRESS	COMMENT	DATE ADDED
+ Add address		
192.0.2.0	Included IP address	27-Jul-2018 21:40:05 UTC
192.0.2.0 /24	Included subnet mask	27-Jul-2018 21:40:48 UTC

2 addresses

NOTE

Remember that ACL containers are versioned. If you don't see an ACL attached to your service, check the service version to make sure you're looking at the right one.

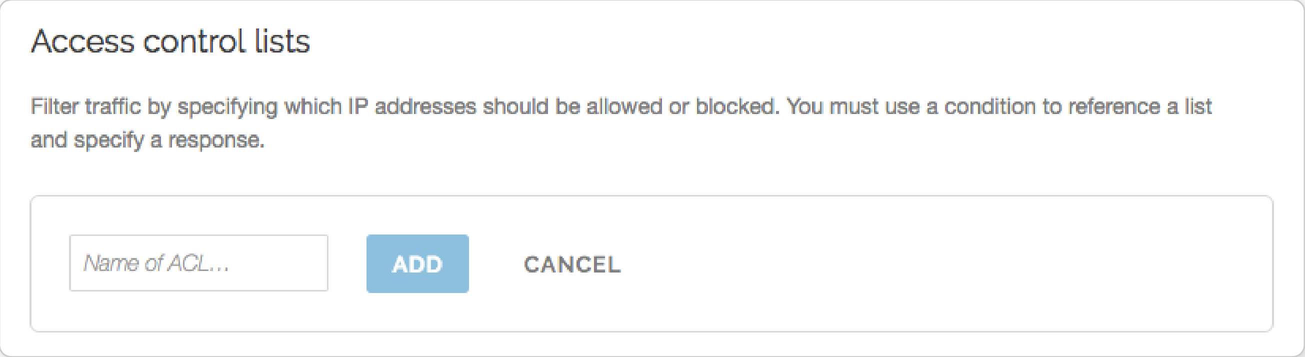
Creating an ACL

ACLs have two parts: an ACL container and the ACL entries within it.

Creating an ACL container

To create an ACL, start by creating an ACL container:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Data**.
5. Click **Create an ACL**.



Access control lists

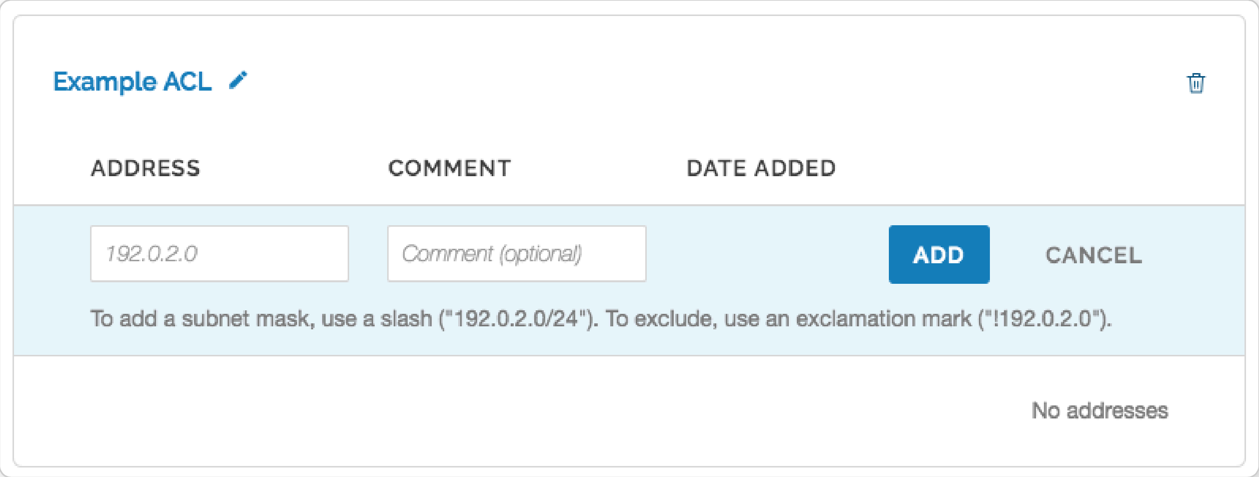
Filter traffic by specifying which IP addresses should be allowed or blocked. You must use a condition to reference a list and specify a response.

6. In the **Name of ACL** field, enter a descriptive name for the ACL (e.g., `Example ACL`).
7. Click **Add**. The empty ACL container you created appears.
8. Click the **Activate** button to deploy your configuration changes to the service version you're editing.

Creating an ACL entry

Once your ACL container is created, add ACL entries into it:

1. Click **Add address**.



Example ACL

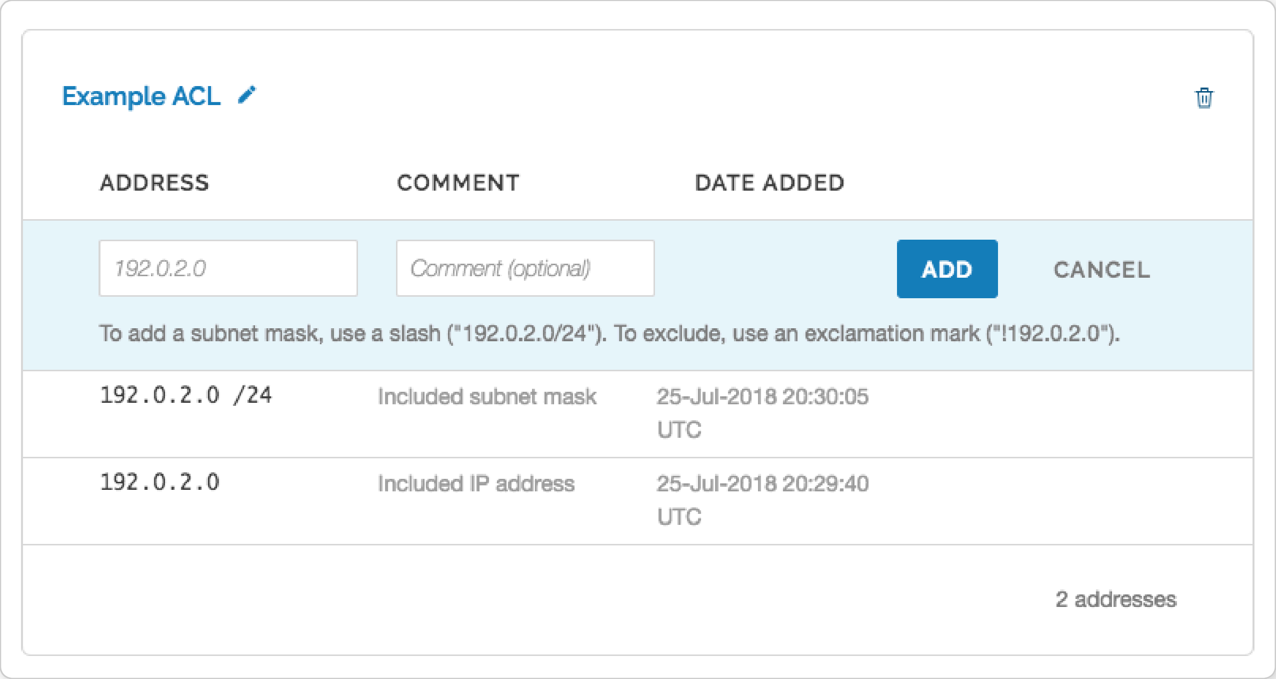
ADDRESS	COMMENT	DATE ADDED
192.0.2.0	Comment (optional)	

To add a subnet mask, use a slash ("192.0.2.0/24"). To exclude, use an exclamation mark ("!192.0.2.0").

No addresses

2. In the **Address** field, enter an IP address or subnet mask (a range of IP addresses) to allow or block for this service. To exclude or block an IP address or subnet mask, use an exclamation point (for example, use `!192.0.2.0` or `!192.0.2.0/24`).
3. In the **Comment** field, enter an optional comment that describes the IP address or subnet mask.

4. Click **Add**. The IP address or subnet mask appears in the ACL container. This addition will become effective immediately.



Example ACL

ADDRESS	COMMENT	DATE ADDED
192.0.2.0	Comment (optional)	
`192.0.2.0 /24`	Included subnet mask	25-Jul-2018 20:30:05 UTC
`192.0.2.0`	Included IP address	25-Jul-2018 20:29:40 UTC

To add a subnet mask, use a slash ("192.0.2.0/24"). To exclude, use an exclamation mark ("!192.0.2.0").

2 addresses

Editing an ACL

Keeping in mind their limitations, the containers and entries of ACLs can be edited via the web interface.

Editing an ACL container

You can edit the name of an ACL container that was created via the web interface in any unlocked service version:

1. Find an ACL associated with an unlocked version of your service.
2. Click the pencil next to the ACL container name.
3. Change the name, then click **Save**.

Editing an ACL entry

You can edit the ACL entries within a container at any time. To edit an IP address or subnet mask included in an ACL container that was created via the web interface:

1. Find any ACL associated with your service in which the associated IP addresses or subnet masks appear. Because ACL entries are versionless, the service version you choose doesn't matter. Choose the one that makes the most sense to you.
2. Hover your cursor over an ACL entry, then click the pencil that appears.
3. Edit the IP address, subnet mask, or comment as necessary.
4. Click **Save**. The changes you make will be immediately applied to your configuration. If you ACL container has already been associated with a deployed service version, those changes will happen live.

Deleting an ACL

Keeping in mind their limitations, the containers and entries of ACLs can be deleted via the web interface.

Deleting an ACL container

You can delete an ACL container that was created via the web interface in any unlocked service version:

1. Find an ACL associated with an unlocked version of your service.
2. Click the trash in the top right corner of the ACL.
3. Click **Confirm and delete**.
4. Click **Activate** to deploy your configuration changes to the service version you're editing.

Deleting an ACL entry

You can delete the ACL entries within a container at any time. To delete an IP address or subnet mask included in an ACL container that was created via the web interface:

1. Find any ACL associated with your service in which the associated IP addresses or subnet masks appear. Because ACL entries are versionless, the service version you choose doesn't matter. Choose the one that makes the most sense to you.
2. Hover your cursor over an ACL entry, then click the trash icon that appears.
3. Click **Confirm and delete**.

* * *

Category: Rate Limiting

These articles describe how to work with Fastly's rate limiting features.

About rate limiting

/en/guides/about-rate-limiting

Rate limiting is a way to control the rate at which traffic flows through Fastly's network to your origins. You might need rate limiting if you need to do things like prevent abusive bots, mitigate DDoS attacks, measure types of activity so you can meter them, and create queueing solutions like waiting rooms to prevent traffic surges.

> NOTE
>
> This page discusses basic information about using the main dashboard for our Edge Rate Limiting product. Edge Rate Limiting is designed for use cases where a faster response to large volumes of traffic is required (e.g., mitigating DDoS attacks). For details about Advanced Rate Limiting and its use in the Fastly Next-Gen WAF, refer to the advanced rate limiting rules documentation.

About the Edge rate limiting page

The Edge rate limiting page displays a summary of all rate limiting policies currently in effect across your services. Each summary includes the following details:

- **Service:** the name of the service
- **Policy name:** the name of the rate limiting policy
- **Requests per second:** the maximum number of requests per second within the detection window counted before enacting the rate limiting policy
- **Detection window:** the duration of the rate limiting window
- **Action:** the action taken once the rate limit is exceeded, either `Block` or `Log only`

Security products note

No security product, such as a WAF or DDoS mitigation product, including those security services offered by Fastly, will detect or prevent all possible attacks or threats. As a subscriber, you should maintain appropriate security controls on all web applications and origins. The use of Fastly's security products do not relieve you of this obligation. As a subscriber, you should test and validate the effectiveness of Fastly's security services to the extent possible prior to deploying these services in production, continuously monitor their performance, and adjust these services as appropriate to address changes in your web applications, origin services, and configurations of the other aspects of your Fastly services.

* * *

Working with rate limiting policies

/en/guides/working-with-rate-limiting-policies

You can use the Fastly Rate Limiting feature to create rate limiting policies. When you create a rate limiting policy, you define the criteria to track requests counts and their rates over time. Accumulated counts are converted to an estimated rate computed over the time window you specify: either 1s, 10s or 60s. Rates are always measured in requests per second (RPS). If the rate limit is exceeded, Fastly logs the event and can also block additional requests for a time period you specify.

Limitations and considerations

When creating or making changes to a rate limiting policy, consider the following questions:

- Should the service be protected, or should different protection levels be configured for different endpoints? For example, authentication-related endpoints may require more strict limits.
- How should clients be identified? For example, should the client key be the client IP address, or a more complex key that identifies individual clients behind a single IP address?
- How should the rates be set? A typical client rate should be established, and rate limiting should be configured over an appropriate time window to allow normal client usage, including some overhead to prevent false positives.

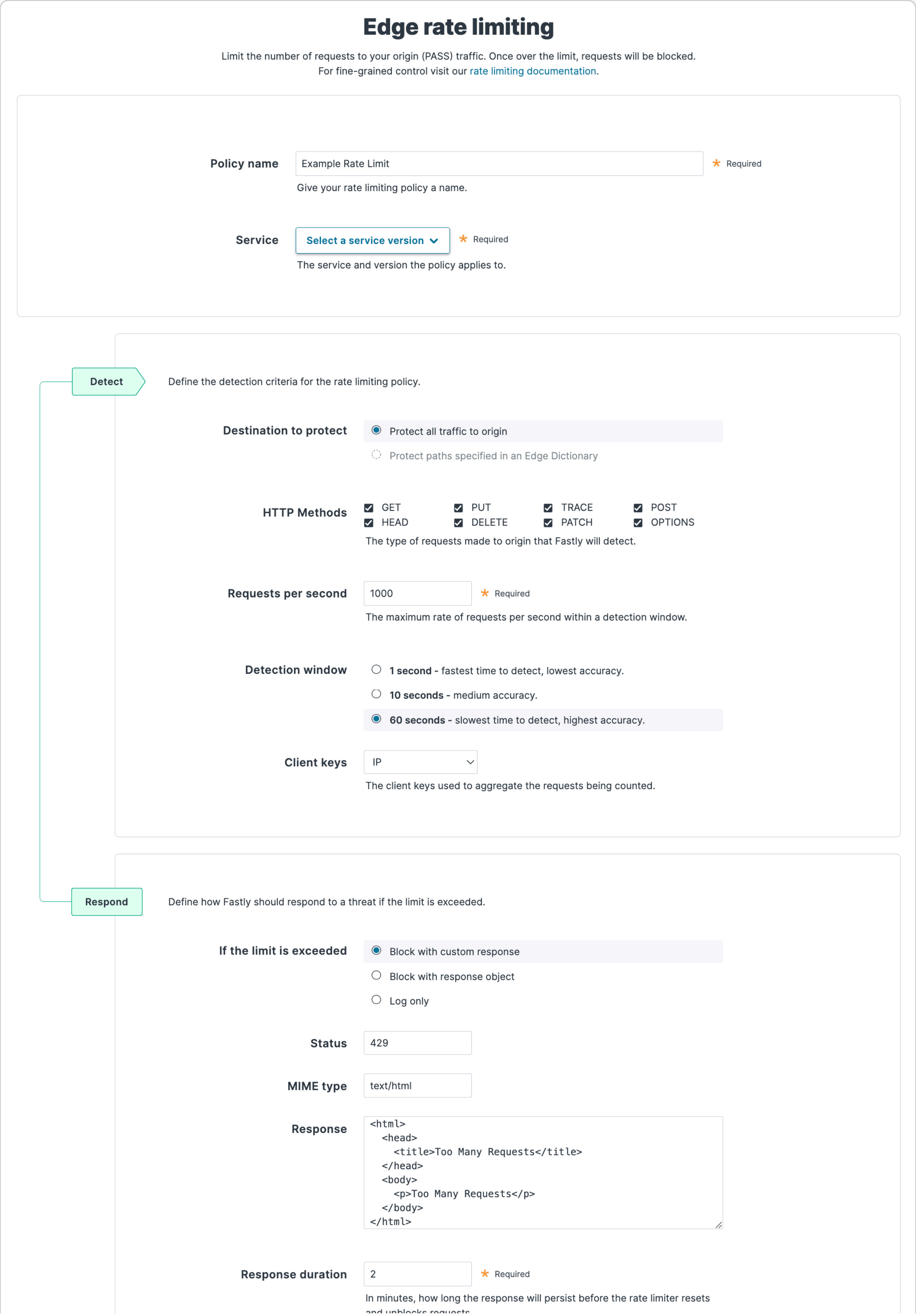
When testing a rate limiting policy, keep the following things in mind:

- You can use logging to test that your rate limiting configuration is behaving as desired.
- You can identify clients or client IP addresses to be excluded. For example, remote performance monitoring, testing solutions, and penetration testing solutions could be excluded from a rate limiting policy.
- If you're using shielding, you may need to make modifications to your rate limiting policy. Refer to the developer documentation for more information.

Creating a rate limiting policy at the edge

Create a rate limiting policy at the edge by following these steps:

1. Log in to the Fastly web interface.
2. Go to **Security** > **Edge rate limiting**.
3. Click **Add rate limiting to a service**.



Edge rate limiting

Limit the number of requests to your origin (PASS) traffic. Once over the limit, requests will be blocked.
For fine-grained control visit our rate limiting documentation.

Policy name Example Rate Limit * Required
Give your rate limiting policy a name.

Service Select a service version * Required
The service and version the policy applies to.

Define the detection criteria for the rate limiting policy.

Destination to protect
● Protect all traffic to origin
○ Protect paths specified in an Edge Dictionary

HTTP Methods
☑ GET ☑ PUT ☑ TRACE ☑ POST
☑ HEAD ☑ DELETE ☑ PATCH ☑ OPTIONS
The type of requests made to origin that Fastly will detect.

Requests per second 1000 * Required
The maximum rate of requests per second within a detection window.

Detection window
○ **1 second** - fastest time to detect, lowest accuracy.
○ **10 seconds** - medium accuracy.
● **60 seconds** - slowest time to detect, highest accuracy.

Client keys IP
The client keys used to aggregate the requests being counted.

Define how Fastly should respond to a threat if the limit is exceeded.

If the limit is exceeded
● Block with custom response
○ Block with response object
○ Log only

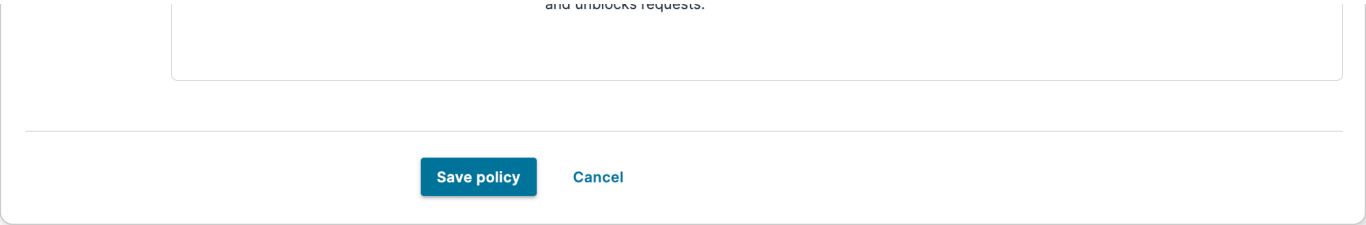
Status 429

MIME type text/html

Response

```
<html>
  <head>
    <title>Too Many Requests</title>
  </head>
  <body>
    <p>Too Many Requests</p>
  </body>
</html>
```

Response duration 2 * Required
In minutes, how long the response will persist before the rate limiter resets



4. In the **Policy name** field, enter a human-readable name for your policy. This name is displayed on the rate limiting dashboard.
5. From the **Service** menu, select the service and version you want to apply the policy to. Use the search box to search by ID or name.
6. In the **Detect** section, fill out the fields to define the detection criteria for the rate limiting policy:
    - In the **Destination to protect** field, select whether to protect all traffic to origin or specific requests via a dictionary. If the service you selected doesn't have a dictionary, this option is disabled.
    - In the **HTTP Methods** field, select the check box next to the types of request to detect.
    - In the **Requests per second** field, enter the maximum number of requests per second to count within the detection window before enacting the rate limiting policy. The lowest rate limit supported by this feature to demonstrate effective behavior is 10 RPS. Using a limit below 10 RPS may result in unpredictable accuracy and detection time.
    - In the **Detection window** field, select the duration of the rate limiting window. The window size helps determine the effective time to detection (TTD) for excessive traffic to your origin. You can use a shorter window to improve the detection time for attacks at some expense of accuracy. For more information, review the limitations and caveats.
    - From the **Client keys** menu, select either **IP**, **User-Agent**, or **IP and User-Agent** to aggregate the requests being counted.
7. In the **Respond** section, select how Fastly should respond to your rate limiting policy being exceeded, then fill out the additional fields that appear.
    - **Block with custom response:** block requests once the rate limit is exceeded and display a custom response in the browser. See Block with custom response.
    - **Block with response object:** block requests once the rate limit is exceeded and display a response object in the browser. Note that you must have a response object created to use this option. See Block with response object.
    - **Log only:** log when the rate limit is exceeded but still allow requests. See Log only.
8. Click **Save policy**.

NOTE

You can create an advanced configuration by configuring rate limiting in VCL. Refer to the developer documentation for more information.

Using dictionaries for more specific protection

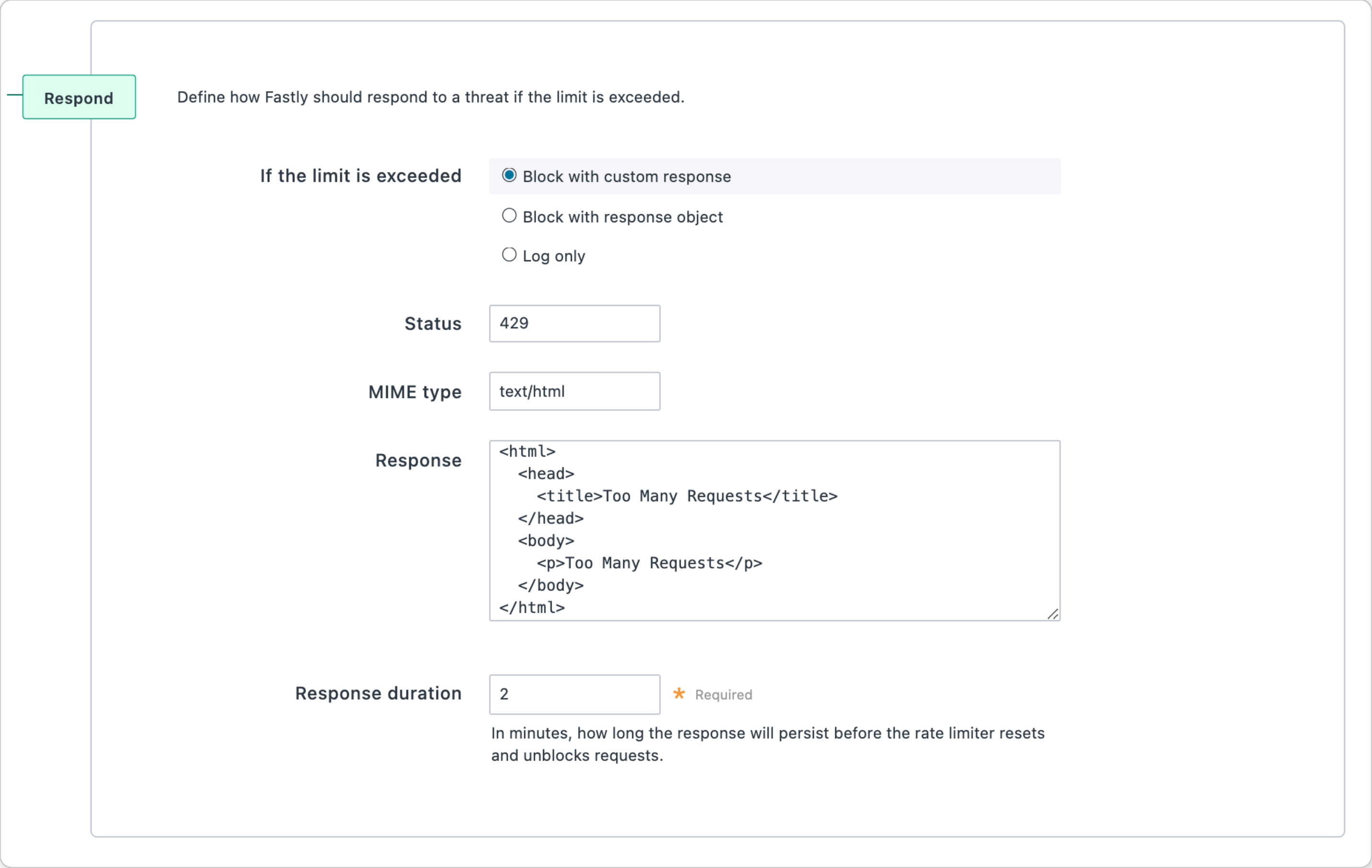
Edge Rate Limiting allows you to specify paths to protect using dictionaries. You must specify a path to protect as a key in the key-value pair within a dictionary. For example, to protect a specific API, such as `/checkout`, you would create a key-value pair where the key is `/checkout` and the value is `Checkout`.

Note the following:

- Keys must be specified using relative paths and without using a trailing `/`.
- Query strings are excluded.
- Keys using regex are not supported.

Block with custom response

Select the **Block with custom response** option to block requests once the rate limit is exceeded and display a custom response. If you select this option, the following fields become available:



Define how Fastly should respond to a threat if the limit is exceeded.

If the limit is exceeded
● Block with custom response
○ Block with response object
○ Log only

Status: 429

MIME type: text/html

Response:

```
<html>
  <head>
    <title>Too Many Requests</title>
  </head>
  <body>
    <p>Too Many Requests</p>
  </body>
</html>
```

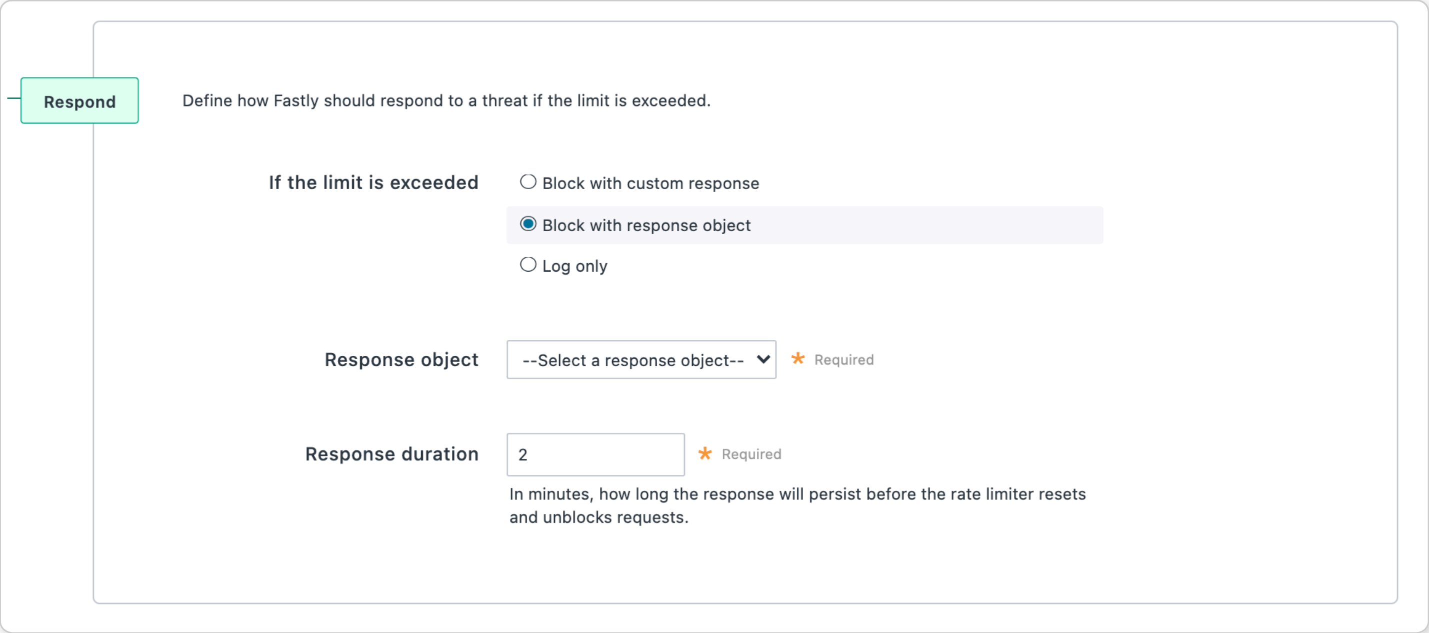
Response duration: 2 * Required

In minutes, how long the response will persist before the rate limiter resets and unblocks requests.

- In the **Status** field, enter an HTTP status code to display in the browser.
- In the **MIME type** field, enter a media type identifier. Any MIME type can be specified as long as it's compatible with the text entered in the `Response` field.
- In the **Response** field, enter the custom response that appears in the browser when the rate limit is exceeded.
- In the **Response duration** field, enter a value between 1 and 60 to determine how long, in minutes, to display the custom response before the rate limiter resets and unblocks requests.

Block with response object

Select the **Block with response object** option to block requests once the rate limit is exceeded and display a response object. Note that you must have a response object created to use this option. If you select this option, the following fields become available:



Define how Fastly should respond to a threat if the limit is exceeded.

If the limit is exceeded
○ Block with custom response
● Block with response object
○ Log only

Response object: --Select a response object-- ★ Required

Response duration: 2 ★ Required

In minutes, how long the response will persist before the rate limiter resets and unblocks requests.

- From the **Response object** menu, select the response object that will appear in the browser when the rate limit is exceeded.

> NOTE
>
> Make sure the response object selected has a condition attached or else the response will be returned for all requests.

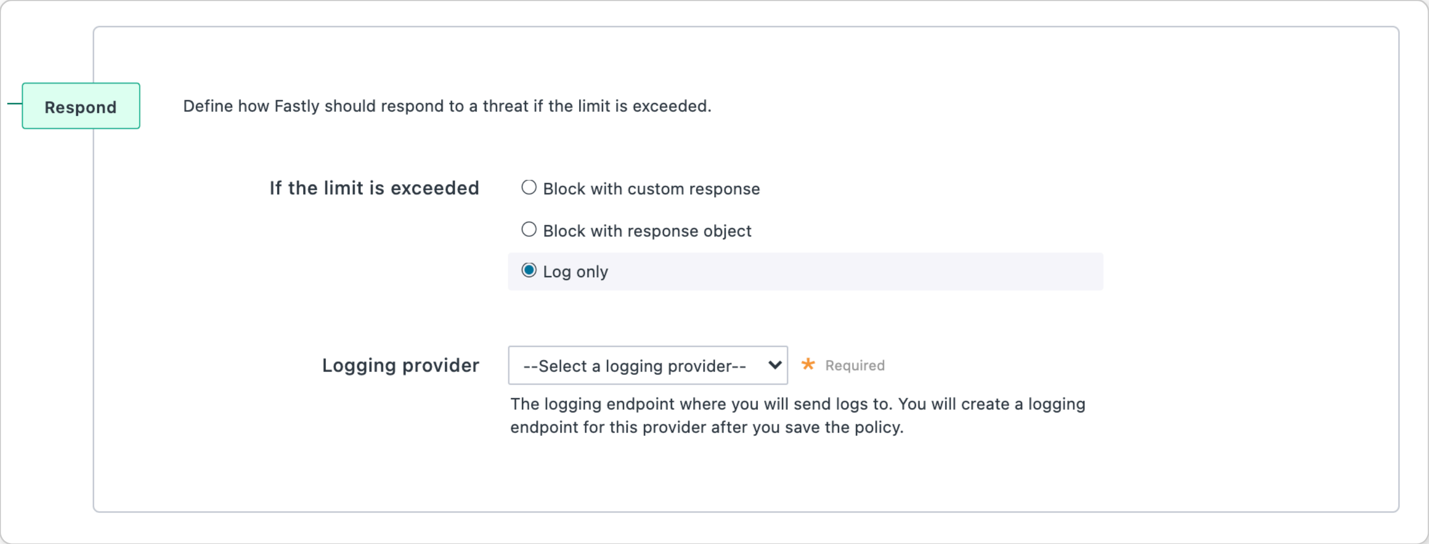
- In the **Response duration** field, enter the number of minutes you want the custom response to appear before the rate limiter resets and unblocks requests.

Log only

Select the **Log only** option to log when the rate limit is exceeded but still allow requests. By default, the following fields are logged: `timestamp`, `policy_name`, `url`, `limit`, `window`, `entry`, and `rate`.

> NOTE
>
> In place of the `timestamp` field, Datadog logs `time_start` and Honeycomb logs `time`.

If you select this option, the **Logging provider** menu becomes available. Select the logging endpoint where you want to send logs. Once you click **Save policy**, an abbreviated form appears to create a logging endpoint for the provider you specified. Complete the fields and then click **Save**.



Define how Fastly should respond to a threat if the limit is exceeded.

If the limit is exceeded
○ Block with custom response
○ Block with response object
● Log only

Logging provider: --Select a logging provider-- * Required

The logging endpoint where you will send logs to. You will create a logging endpoint for this provider after you save the policy.

NOTE

You can only have one logging endpoint for all rate limiting policies. By default, the logging endpoint is named `ratelimit-debug`. You can also create this endpoint via the web interface or API.

Log formats

The following JSON log formats are used when you choose the Log only option. These schemas can be useful to understand how data is written to different logging providers.

Unless otherwise specified, the default format is used to log information about particular policies that exceed the rate limit.

```
{
  "timestamp": “%{strftime({"%Y-%m-%dT%H:%M:%S"}, time.start)}V”,
  "policy_name": "%{json.escape(“<rate limiter name>”)}V”,
  "url": “%{json.escape(req.url)}V”,
  "limit": <limit>,
  "window": <window>,
  "entry": "<entry>",
  "rate":  “<rate>"
}
```

```
{
  "time_start": “%{strftime({"%Y-%m-%dT%H:%M:%S%Z"}, time.start)}V”,
  "ddsource": "fastly",
  "service": “%{req.service_id}V",
  "policy_name": "%{json.escape(“<rate limiter name>”)}V”,
  "url": “%{json.escape(req.url)}V”,
  "limit": <limit>,
  "window": <window>,
  "entry": "<entry>",
  "rate":  “<rate>"
}
```

```
{
  "time": “%{strftime({"%Y-%m-%dT%H:%M:%SZ"}, time.start)}V”,
  "data":{
    "service_id": “%{req.service_id}V",
    "policy_name": "%{json.escape(“<rate limiter name>”)}V”,
    "url": “%{json.escape(req.url)}V”,
    "limit": <limit>,
    "window": <window>,
    "entry": "<entry>",
    "rate":  “<rate>"
  }
}
```

Editing a rate limiting policy

After you've created a rate limiting policy, you can edit the same controls specified when you created the policy. You can only edit policies attached to services in a Draft state. You cannot edit policies attached to Active or Locked services.

To edit a rate limiting policy, follow the steps below:

1. Log in to the Fastly web interface.
2. Go to **Security** > **Edge rate limiting**.
3. Locate the rate limiting policy you want to edit.
4. For policies attached to a Draft service, click the pencil to edit the rate limiting policy. For policies attached to an Active or Locked service, click **Clone version to edit**.
5. Update the values of the edge rate limiting controls you want to edit.
6. Click **Save policy**.

Deleting a rate limiting policy

To delete a rate limiting policy, follow the steps below. You can only delete policies attached to services in a Draft state. You cannot delete policies attached to Active or Locked services. To delete a policy attached to an Active or Locked service, you must first clone the service.

To delete a rate limiting policy:

1. Log in to the Fastly web interface.
2. Go to **Security** > **Edge rate limiting**.
3. Locate the rate limiting policy you want to delete.
4. For policies attached to a Draft service, click the trash and then **Confirm and delete** to delete the rate limiting policy.

Security products note

No security product, such as a WAF or DDoS mitigation product, including those security services offered by Fastly, will detect or prevent all possible attacks or threats. As a subscriber, you should maintain appropriate security controls on all web applications and origins. The use of Fastly's security products do not relieve you of this obligation. As a subscriber, you should test and validate the effectiveness of Fastly's security services to the extent possible prior to deploying these services in production, continuously monitor their performance, and adjust these services as appropriate to address changes in your web applications, origin services, and configurations of the other aspects of your Fastly services.

* * *

Category: Securing communications

These articles describe how to secure communications between Fastly, your origin servers, and your customers.

Support for App Transport Security

/en/guides/support-for-app-transport-security

Apple uses App Transport Security (ATS) to improve the security of connections between web services and applications installed on devices using iOS 9 or later, as well as OS X 10.11 (El Capitan) and later. Fastly is fully compliant with all ATS requirements. You shouldn't run into any issues supporting iOS or OS X users while using our service.

Results from the ATS diagnostics tool

We used Apple's ATS diagnostics tool to ensure that Fastly is compliant with all ATS requirements. You can review the output from the diagnostics tool below.

```
$ /usr/bin/nscurl --ats-diagnostics https://www.fastly.com
Starting ATS Diagnostics

Configuring ATS Info.plist keys and displaying the result of HTTPS loads to https://www.fast
A test will "PASS" if URLSession:task:didCompleteWithError: returns a nil error.

Use '--verbose' to view the ATS dictionaries used and to display the error received in URLSe
================================================================================

Default ATS Secure Connection
---
ATS Default Connection

Result : PASS
---

================================================================================
Allowing Arbitrary Loads
```

```
---
Allow All Loads

Result : PASS
---

================================================================================

Configuring TLS exceptions for www.fastly.com

---
TLSv1.2

Result : PASS
---

---
TLSv1.1

Result : PASS
---

---
TLSv1.0

Result : PASS
---

================================================================================

Configuring PFS exceptions for www.fastly.com

---
Disabling Perfect Forward Secrecy

Result : PASS
---

================================================================================

Configuring PFS exceptions and allowing insecure HTTP for www.fastly.com

---
Disabling Perfect Forward Secrecy and Allowing Insecure HTTP

Result : PASS
---

================================================================================

Configuring TLS exceptions with PFS disabled for www.fastly.com
```

```
---
TLSv1.2 with PFS disabled

Result : PASS
---


---
TLSv1.1 with PFS disabled

Result : PASS
---


---
TLSv1.0 with PFS disabled

Result : PASS
---

================================================================================

Configuring TLS exceptions with PFS disabled and insecure HTTP allowed for www.fastly.com

---
TLSv1.2 with PFS disabled and insecure HTTP allowed

Result : PASS
---


---
TLSv1.1 with PFS disabled and insecure HTTP allowed

Result : PASS
---


---
TLSv1.0 with PFS disabled and insecure HTTP allowed

Result : PASS
---
```

* * *

Category: Security measures

These articles provide information about the administrative, physical, and technical safeguards that protect Fastly's product and services.

Data management

/en/guides/data-management

Fastly maintains a “privacy and protection by design” approach that is manifested in Fastly’s data governance program. Fastly is intentional about data processing, collection and access to customer and personal data. Fastly does not collect more data than needed to perform its services. Fastly considers legal, compliance, regulatory, and commercial obligations when working with data. Fastly appropriately protects records and information that are private, confidential, privileged, secret, essential to business continuity, or that otherwise require protection.

Fastly production data

In addition to the Fastly security program, Fastly maintains the following data management practices in production environments.

Data Insights

- *Service management:* Fastly collects and processes data related to the functional performance of Fastly services, anomalous activity, and suspicious behavior detected by the services. Fastly retains and uses this data to monitor, maintain, and improve its services, business operations, and security and compliance programs.
- *Confidentiality:* Fastly only discloses this data in an anonymized and aggregated form and subject to its confidentiality obligations to customers.

IP addresses

- *Security events:* Fastly may indefinitely retain any non-anonymized, non-aggregated client or customer IP addresses associated with security-related incidents or administrative connections to Fastly’s services. Fastly may retain non-anonymized, non-aggregated client or customer IP addresses associated with this anomalous activity or suspicious behavior for a period of up to 30 days.
- *Suspicious activity:* Fastly keeps internal systems logs, including access logs, related to events triggered by anomalous activity or suspicious behavior for at least one year. Fastly may retain IP addresses from Fastly event logs or configurations indefinitely.
- *Fastly application:* Fastly independently collects the IP addresses of users who access services within the Fastly web interface or through the Fastly API.
- *Endpoints:* If a customer defines origin servers or syslog endpoints with IP addresses, Fastly will save those IP addresses as part of the customer’s configurations.
- *Client IPs:* Fastly retains client IP addresses in a non-anonymized, non-aggregated fashion for up to two business days, or up to seven days if those addresses are associated with transmission errors.
- *Origin IPs:* Fastly may retain dynamically-resolved origin IP addresses for up to two business days, or up to seven days if associated with transmission errors. The IP addresses are discarded thereafter.

Customer data management

The duration of any data retention will vary based on the type of data and its use.

- *Customer content:* Customer content enters, transits, and departs Fastly's network in response to requests. Generally, customers manage which content is processed, where, and for how long by setting policies that control that content.
- *Customer configurations:* Customer configurations may be stored indefinitely, but can be deleted upon request. Fastly may directly access or modify customer accounts or configurations as necessary to provide services, to prevent or address service or technical issues, as required by law, or as customers expressly permit. Fastly retains encrypted backups of customer configurations, including VCL, and customer provided packages for business continuity purposes.
- *Cached content:* Cached content is retained per customer configuration and use of purge functionality. Customers may control length and type of retention through configuration options to meet requirements for regulatory reasons such as HIPAA or PCI DSS. Fastly deletes cached content according to a customer's use of the purge functionality and as described in documentation.
- *Customer packages deployed to Compute:* Customer provided compiled code may be stored indefinitely, but can be deleted upon request.

Customer request logs

- *Content request logs:* Customers may stream their content request logs, which may include request headers, including client IP addresses, to a customer-owned and managed endpoint for analysis and use.
- *Request logs retention:* Fastly does not retain customers' request logs except where explicitly stated in the Documentation and related to the functional performance of the services.

Note regarding Signal Sciences data management

The Signal Sciences security measures describe the Signal Sciences data management practices.

Note regarding privacy law

For more information regarding Fastly's compliance with global privacy laws and regulations, refer to the Fastly data processing terms, the list of sub-processors, Fastly's privacy policy, and additional resources on the Fastly Trust page.

* * *

	Fastly Next-Gen WAF security measures
🔗	/en/guides/fastly-next-gen-waf-security-measures

Fastly's security measures for the Fastly Next-Gen WAF (powered by Signal Sciences) (Next-Gen WAF) include safeguards that help protect your data as it moves through the Next-Gen WAF. It has three deployment options: Edge, Core, and Cloud WAF. The security measures described on this guide apply to the entirety of the Core and Cloud WAF deployments. For Edge deployments, the security aspects and associated data handling are covered by the terms on this page. The hosted components of Edge deployments are hosted within Fastly's Compute at Edge environment and are subject to our security measures.

The Fastly Next-Gen WAF now collectively refers to the products that were previously known as the Signal Sciences Cloud WAF and Signal Sciences Next-Gen WAF. The functionality of those products has not changed as part of the new naming convention. Fastly Next-Gen WAF continues to be powered by Signal Sciences technology.

Authentication and authorization

- Our systems and devices enforce user roles or similar measures to control the extent of access we grant individual users.
- We control access to privileged systems using Zero Trust access policies that use client certificates and two-factor authentication.
- Our authentication requirements, such as passwords, are in line with industry standard practices.

Business continuity and operational resilience

- We monitor production operation systems and supporting systems to detect service-related and non-compliance issues on a continuous basis. The systems are monitored 24×7 to ensure constant availability to clients.
- If an update has potential impact to customer uptime, we will determine a timeline for the update and communicate the impact to customers via https://www.fastlystatus.com.
- We maintain our services in multiple Availability Zones (AZs) to operate production applications and databases that are more highly available, fault-tolerant, and scalable than would be possible from a single data center.
- We update impacted customers using various communication methods (such as https://www.fastlystatus.com), depending on an incident's scope and severity.

Cloud infrastructure data center and physical security

We rely on data center space under the control of Amazon Web Services (AWS) and their physical security controls. As part of our third-party security review process, we confirm that these providers maintain appropriate physical security measures to protect their data center facilities.

Customer and end user data management

- We do not store sensitive customer data processed by Core deployments in the cloud. Customers process this data in local environments under their control with no remote access by our employees.
- We store and retain customer data that is sent to us and that is processed via the security components of Next-Gen WAF for up to 30 days.
- The Next-Gen WAF analyzes requests. We retain and use data about the operation and reliability of our processing of requests to monitor, maintain, and improve our services, our business operations, and our security and compliance programs. Subject to confidentiality obligations to our customers, we only disclose this data in anonymized and aggregated form.

Encryption

We use industry-accepted encryption technologies to encrypt sensitive information. All client data is encrypted in transit using TLS.

Governance

- We have formally assigned information security duties to our personnel. Our Chief Security Officer and Security organization work with other departments to safeguard sensitive information related to our services.
- Our policies and procedures help us maintain security in our systems, processes, and employee practices. Our Security organization formally reviews these policies and procedures at least annually.
- We integrate risk assessment activities with various processes to identify and address information security risk to the company and customer data on our network.
- We perform risk-based evaluations of the security measures of our vendors. We review these security measures before we begin using a vendor, and we ask the vendor to formally acknowledge these measures. We re-evaluate vendor security measures on a recurring basis thereafter.

Human resources security

- Our employees formally agree to safeguard the sensitive information they may view, process, or transmit as part of their job functions.
- We train our people to protect the data and devices they use. Each employee receives security awareness training as part of new hire procedures, and current employees take this training annually.
- We screen new employees as part of the hiring process. Screening activities depend on applicable local regulations and may include criminal background checks and reference checks.

Identity and access management

- We periodically inspect access privileges to make sure our personnel have appropriate access to our systems and data.
- We promptly update or remove an employee's access to our network to match that employee's current job function or employment status.

Logging and monitoring

- We configure thresholds within our monitoring tool to alert when a security policy has been violated. Threshold policies are reviewed on an annual basis for accuracy and appropriateness.
- We restrict, log, and monitor information security management systems activity with anomaly alerting. We aggregate and securely store the activity in a centralized internal log server.

Network and infrastructure security

- We review and validate information systems and network device configurations against established security policies and procedures.
- We regularly perform vulnerability scans and third-party penetration tests on our network. We review and address findings from these activities to help maintain the security of our network.
- To maintain awareness of potential security vulnerabilities, we monitor public and private distribution lists, as well as reports submitted through our responsible disclosure process. We validate and implement security

patches for critical vulnerabilities within 24 hours of discovery. For non-critical vulnerabilities and updates, we schedule and deploy vendor-provided patches on a regular basis.

- To protect from known vulnerabilities, we maintain assets at the latest version and patch levels currently supported by vendors. Priority of patch deployment is based on vulnerabilities and risks it poses to the environment.

Security incident management

- We maintain a formal incident response plan with established roles and responsibilities, communication protocols, and response procedures. We review and update this plan periodically to adapt it to evolving threats and risks to our services.
- We will notify affected customers within 48 hours of validating an unauthorized disclosure of customer confidential information.

* * *

Penetration testing your service behind Fastly

/en/guides/penetration-testing-your-service-behind-fastly

We understand the need for our customers to validate the security of their service behind Fastly.

IMPORTANT

Penetration tests that interfere with or disrupt the integrity or performance of Fastly services violate our acceptable use policy. You must respond immediately to any communication from Fastly regarding your test to help ensure your testing does not adversely affect other customers or the Fastly network.

To perform security testing of your Fastly service configurations, contact support at least two (2) business days before you begin any security testing. In your ticket, include these details:

- the IDs of the services that will be tested
- the source IP address of the test
- the date of the test
- the start and end time of the test, including the time zone
- the contact information for the individual or third party performing the test, including a phone number and e-mail address
- whether or not the security test is likely to lead to significantly increased traffic volume

The following requirements apply to any security testing you perform:

- Only test Fastly services you own or are authorized by the owner to test. You may not perform tests against other customers without explicit permission or against Fastly-owned resources.
- Do not begin testing until after Fastly has responded affirmatively to your ticket and authorized your request.

- Update the ticket if either the scope or timeframe of your testing changes.
- If you discover vulnerabilities in the Fastly platform during your test, update the ticket with your findings as soon as possible so we can address them.

Fastly maintains programs for security and technology compliance. To perform an independent audit of these programs, contact sales@fastly.com to discuss purchase of Assurance Services.

TIP

We welcome security professionals researching potential vulnerabilities in our network under our guidelines for reporting a security issue.

* * *

Security program

/en/guides/security-program

Fastly operates a comprehensive information security program that includes administrative, physical, and technical safeguards to protect its infrastructure, data, services, and customers.

Foundation

Fastly's security program is based on the NIST Cybersecurity Framework comprised of annually reviewed security policies, designated roles and responsibilities for its experienced professionals, and formal procedures developed focused on risk.

Security policies

Fastly institutes information security policies that are published internally and reviewed annually. The policies contain principles and point to standards that cover controls and procedures designed to protect Fastly and Fastly's customers.

Experienced professionals

Fastly designates roles and responsibilities for the security of its services. Fastly assigns a Chief Information Security Officer to oversee its security program and retains best in class professionals in the field to apply it.

Risk-based approach

Fastly maintains formal procedures for the identification, assessment, and treatment of information security and availability risks, threats, and vulnerabilities to its services. The procedures include an annual risk assessment, risk analysis and treatment plan, and a risk register.

- *Annual risk assessment:* Fastly conducts an annual risk assessment to measure the state of security risk across the company. The results of this assessment are shared with the senior leadership team to ensure appropriate visibility and treatment.

- *Risk analysis and treatment plan:* Each identified enterprise security risk is evaluated and ultimately managed to acceptable levels by implementing associated controls and mitigation plans commensurate with the risk.
- *Risk register:* Fastly maintains documentation of identified risks, threats, and vulnerabilities related to its services. Assigned personnel help assess and remediate identified items, in line with the related risk and vulnerability management procedures.

Defense-in-depth

Fastly understands that to adequately protect its services, customers, and customer data, multiple safeguards must be applied to all layers of Fastly's business and technology practices. Fastly's process, technology, and physical security controls are designed specifically to provide a defense-in-depth approach and can be categorized as follows:

Identity and access management

Fastly manages access to its production systems using the following:

- *Authentication:* Employees are required to use unique user accounts and multi-factor authentication for remote access to production systems.
- *Authorization:* Employee access to production systems is restricted based on appropriate roles.
- *Audit:* Logs of access attempts (both success and failure) to production systems are kept and monitored.
- *Access grants and revocations:* Employee access to production systems is granted based on the principle of least privilege and manager approval. That access is reviewed at least quarterly and is removed when no longer needed or upon employee separation. Access roles are enforced by Fastly systems and devices.

Data security

Fastly manages data security using the following:

- *Customer credentials management:* Fastly secures customer-provided private keys and credentials throughout their lifecycle and stores private keys and API tokens in encrypted repositories. Customer-provided private keys are encrypted at rest and are re-encrypted on a regular interval. The key encryption keys are stored in a secrets management system and private keys are decrypted in memory at the edge when requested and removed from memory after a short period of time. Customer passwords are salted and hashed at rest and Fastly enables encryption for customer account passwords in transit. Access to private keys is restricted to only those individuals whose role requires it.
- *Authorized access to customer data:* Fastly may directly access or modify customer accounts or configurations as necessary to provide the services, prevent or address service or technical issues, as required by law, or as customers expressly permit. For the same reasons, Fastly may also access or modify equipment, systems, or services that manage customer data.
- *Privacy and protection-by-design approach:* Fastly maintains a "privacy and protection-by-design" approach that is manifested in a data governance program and documented separately in the data management documentation online.

Application security

Fastly manages application security using the following:

- *Secure development practices:* Fastly engineers are trained annually on secure coding concepts, including the OWASP Top 10 and CWE Top 25. Code is peer-reviewed and run through automated testing before deployment to production systems. After review and testing, code is initially deployed to a limited number of locations in the Fastly network for further monitoring. If no problems are encountered, code is gradually deployed across the Fastly network.

- *Application security analysis:* Fastly security engineers and third-party validators conduct periodic analysis and regular penetration testing of Fastly-written code.
- *Automated code analysis:* Fastly deploys technology to automatically identify and report on identified vulnerable dependencies.

System and network security

Fastly manages system and network security using the following:

- *Asset management:* Fastly maintains an inventory of its hardware and services deployed within the Fastly network.
- *Configuration standards:* Fastly maintains secure configuration standards, including restricted ports, protocols, and services, and removal of insecure default settings.
- *Patch management:* Fastly patches its production systems on a regular basis and applies out-of-band patches for newly-identified risks.
- *Endpoint management:* Fastly manages its production systems by verifying appropriate security settings are in place, including logging and monitoring, host-based firewalls, and session management.
- *Audit and monitoring:* Fastly logs relevant security-related events, including authentication successes or failures to production systems and the use of certain commands. Fastly investigates events triggered by anomalous activity or suspicious behavior.
- *Documentation:* Fastly maintains accurate network diagrams and internal documentation of its systems and services.
- *Access Control List (ACL) review:* On at least a semi-annual basis, Fastly conducts a production system ACL review of its endpoint firewall and router rulesets.
- *Intrusion Detection:* Fastly maintains mechanisms designed to detect potential intrusions at the network and host level. Fastly inspects and responds to detected events, as necessary, to address threats.

Physical security

Fastly production systems reside in a combination of Fastly-managed data centers and cloud infrastructure environments. Regardless of the physical location of the infrastructure or its operator, Fastly evaluates and applies the same minimum, mandatory physical security controls.

- *Physical access management:* Fastly uses providers that maintain industry standard physical and environmental protections, including perimeter protection, security guard assignment, access logging and review, and video surveillance.
- *Physical access to production systems:* Physical access is granted only to approved personnel. Requests for access are evaluated by authorized personnel and based on proof of proper credentials, appropriate and documented use-case, and limited to areas specified in their permissions.
- *Environmental security safeguards:* Providers protect their systems with controls including power redundancy, fire suppression, and other environmental controls.

- *Secure hardware destruction:* Providers use industry standard secure destruction of all production hardware prior to disposal.

Human security

Fastly manages human security using the following:

- *Employee background screening:* Fastly conducts background screenings on each of its employees upon hire, with recurring criminal conviction checks periodically thereafter, and maintains a policy requiring employees to report any criminal convictions during the course of employment, each as permitted by applicable local regulations.
- *Confidentiality agreements:* To safeguard sensitive information that employees may view, process, or transmit as part of their job functions, all employees enter into confidentiality agreements with Fastly.
- *Awareness training:* All employees receive security training upon hire and annually thereafter designed to help protect Fastly and its customers. Mandatory annual training includes security awareness that covers application of best security practices in day-to-day work and privacy to ensure each employee understands how to identify sensitive information and comply with regulations.

Continuous monitoring and improvement

To ensure that the controls described above are consistently applied and effective in their intended use, Fastly continuously monitors and improves its security measures. Fastly institutes strict processes and testing procedures as follows.

Change management process

Fastly follows a defined set of procedures to develop and deploy technology changes. These changes include updates to software, configurations, and devices that support Fastly’s services.

- *Testing:* Fastly tests changes at various stages of development and confirms the changes operate as expected in a non-production environment before completing a deployment into its services.
- *Change approval and notification:* Fastly prepares, approves, and communicates change notices to maintain awareness among employees who manage the Fastly network and systems. Fastly maintains rollback procedures to address deployment issues if they arise.
- *Post-implementation review:* Fastly confirms the success of changes after deployment.
- *Change monitoring:* Fastly uses multiple monitoring and alert mechanisms to enhance the visibility of technical changes and help ensure adherence to change management processes.

Vulnerability management

Fastly monitors for vulnerabilities in its production systems using the following measures:

- *Internal and external vulnerability scanning:* On a regular basis, Fastly automatically analyzes its production systems for vulnerabilities.
- *Vulnerability mitigation:* Fastly assesses the risk of identified or reported vulnerabilities, and mitigates vulnerabilities in a timely manner. Mitigations for vulnerabilities deemed highest severity are implemented within twenty-four (24) hours of validation.

- *Distribution lists and vendor notification:* Fastly monitors publicly disclosed and vendor confidential distribution lists and notifications from software vendors for vulnerabilities.

Penetration testing

On a semi-annual basis, Fastly engages a third-party to conduct a penetration test of Fastly production systems. Identified issues are prioritized and handled in order based upon the severity of the evaluated risk they pose.

Compliance and audits

Fastly maintains recurring audits and assessments that confirm its security program meets various industry standards and regulatory requirements.

Fastly vendor management

Fastly uses third-party vendors and service providers to support its services. Fastly evaluates its vendors for security controls and risk to Fastly and its services prior to using vendor services, and regularly thereafter based on vendor risk.

When something goes wrong

Fastly aims to provide a consistently reliable and secure platform. With this in mind, Fastly is always monitoring for threats and systems disruptions so incidents are detected, responded to, and recovered from in a timely manner.

Incident management plan

Fastly maintains a formal incident response plan to address security-related incidents. The plan contains established roles and responsibilities, communication protocols, and response procedures. Fastly reviews and updates the plan periodically to adapt it to evolving threats and risks to its services. Representatives from key departments are assigned to address security-related incidents. These personnel coordinate the full lifecycle of incidents, from detection, through response, and recovery. Included within these processes is communication with external contacts as needed.

Incident notification

Fastly notifies affected customers within forty-eight (48) hours of validating any unauthorized disclosure of customer data. Following any security-related incident, Fastly investigates and takes corrective action in a timely manner according to the incident management plan and provides affected customers with periodic updates.

Business continuity

Fastly manages business continuity using the following:

- *Service failover:* Fastly production systems are designed to be prepared for service failover. Production systems are deployed on infrastructure in multiple regions or zones to provide redundancy in the event of degraded performance or operational issues with a provider. If failure of a service occurs within a single region or zone, Fastly will automatically attempt to use infrastructure in another region or another infrastructure provider.
- *Internet redundancy:* Fastly data centers and cloud infrastructure providers have connections with multiple internet service providers.
- *Service monitoring:* Fastly monitors reporting channels to detect service-related issues. Personnel are available 24×7×365 to confirm and respond to disruptions of its services.

- *Communication and Reporting:* Fastly provides service interruption updates to customers using various communication methods (including fastlystatus.com), depending on an incident's scope and severity.
- *Business continuity plan and testing:* Fastly has a business continuity plan for its production systems that is reviewed, approved, and updated annually. Fastly tests its business continuity plan on an annual basis.
- *Data backups:* Fastly conducts regular backups of data, excluding cached customer data, to support the recovery and availability of its services. Data backups are tested on a quarterly basis to validate backup recovery procedures.

* * *

Category: TLS

These articles describe how to set up TLS certificates with Fastly services.

About the TLS dashboard

/en/guides/about-the-tls-dashboard

The TLS dashboard provides a high-level overview of the status of your Fastly-managed and self-managed TLS certificates. It alerts you to any issues with your certificates and summarizes the actions you may need to take as a result of any of those alerts.

Viewing the TLS dashboard

You can access the TLS dashboard after you set up your first Fastly-managed or self-managed certificate.

To view the TLS dashboard:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Dashboard**.

About the TLS certificate dashboard

The TLS certificate dashboard provides a summary of the status of your Fastly-managed certificate subscriptions and self-managed TLS certificates, depending on which certificate type you're using to secure your domains.

The Subscriptions summary displays a count of:

- the number of certificate subscriptions that Fastly was unable to renew and that are no longer securing traffic. Refer to the TLS certificate alerts for the specific certificates that failed renewal and follow the instructions to retry.
- the number of certificates Fastly is currently attempting to reverify and renew.
- the number of certificates undergoing domain verification processing.
- the number of valid certificates.

For more information on Fastly-managed certificate expiration and renewals, refer to Certificate management and renewals.

The Self-managed certificates summary gives a count of:

- the number of certificates you manage that are expired. Refer to the TLS certificate alerts for the specific certificates that expired.
- the number of certificates you manage nearing expiration.
- the number of valid certificates.

For more information on self-managed certificate expiration and renewals, refer to Certificate expirations.

About the TLS alerts

The TLS alerts section appears any time you have certificates with errors. It displays a table of all certificates that have either expired or failed to renew and contains the following specific details about those certificates:

- **Status:** the status of the certificate with the alert.
- **Certificate name:** the name of the certificate.
- **Expiration/Renewal Date:** a UTC timestamp of when the certificate expired.
- **Domains:** a list of domains associated with the expired certificate and that may no longer be serving secured traffic.
- **Alert description:** a description of the error on the certificate. Click the linked text to take steps to resolve the error.

By default, all certificate alerts appear in this section. You can limit the displayed results by selecting a filter from the **Filter** menu or by using the search field to find a specific alert by domain.

About the Clean up tasks

The Clean up tasks section contains recommended actions to take based on the certificates and keys added to your services. This section appears if you have unused certificates or unmatched private keys.

Expand the **Unused certificates** section to view details about unused certificates including when the certificates expire and which domains are associated with the certificates. Unused certificates can be safely deleted. To delete unused certificates:

1. Navigate to the row with the unused certificate you want to delete.
2. Click **Delete**. A confirmation window appears.
3. Click **Delete Certificate**. The unused certificate is deleted and is no longer listed.

Expand the **Unmatched private keys** section to view details about private keys that don't have a matching certificate uploaded. Upload a matching certificate or safely delete the key. To delete unmatched private keys:

1. Navigate to the row with the unused certificate you want to delete.
2. Click **Delete**. A confirmation window appears.
3. Click **Delete Certificate**. The unmatched private key is deleted and is no longer listed.

* * *

Enabling HSTS through Fastly

/en/guides/enabling-hsts-through-fastly

The HTTP Strict Transport Security (HSTS) security enhancement specification provides a way to force modern browsers to communicate only via the Transport Layer Security (TLS) protocol. Once enabled, HSTS will force the browser to redirect (typically with a status code 307) to the HTTPS URL as long as the URL has previously been visited. For example, making a request for `http://www.example.com` would force a redirect to `https://www.example.com` as long as `https://www.example.com` has been visited once before.

> NOTE
>
> Because HSTS only takes effect after a site has been visited on a trusted HTTPS connection, we recommend forcing TLS and enabling HSTS. If you'd prefer not to automatically enable HSTS, you can still manually enable it after setting up TLS redirects.

Prerequisites

These instructions assume that you've set up TLS service with Fastly.

Forcing TLS and enabling HSTS

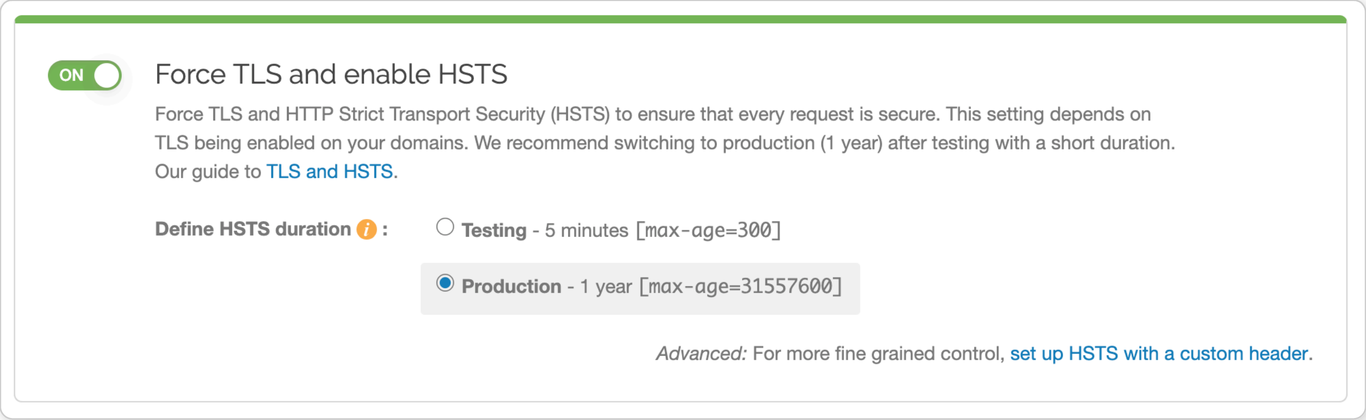
To force TLS and enable HSTS, follow these steps.

> NOTE
>
> Services activated using a previous version of the Force TLS controls may temporarily display an additional, older testing duration. Once you select the recommended new testing duration, this older option will disappear.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Settings**.

   > NOTE
   >
   > If you have **Force TLS** enabled for any request settings without a condition, conflicts in the VCL logic may occur. Delete the existing request setting or modify it to add a condition.

5. Click the **Force TLS and enable HSTS** switch to force TLS and enable HSTS for the service.



ON Force TLS and enable HSTS

Force TLS and HTTP Strict Transport Security (HSTS) to ensure that every request is secure. This setting depends on TLS being enabled on your domains. We recommend switching to production (1 year) after testing with a short duration. Our guide to TLS and HSTS.

Define HSTS duration :

○ **Testing** - 5 minutes [max-age=300]

● **Production** - 1 year [max-age=31557600]

Advanced: For more fine grained control, set up HSTS with a custom header.

The request setting for forcing TLS and the header for enabling HSTS will automatically be created for you.

6. Click **Activate** to deploy your configuration changes.

WARNING

You may experience problems if you enable this setting along with the override host setting. Instead of enabling the override host setting, create a new request setting and specify the override host in the advanced options.

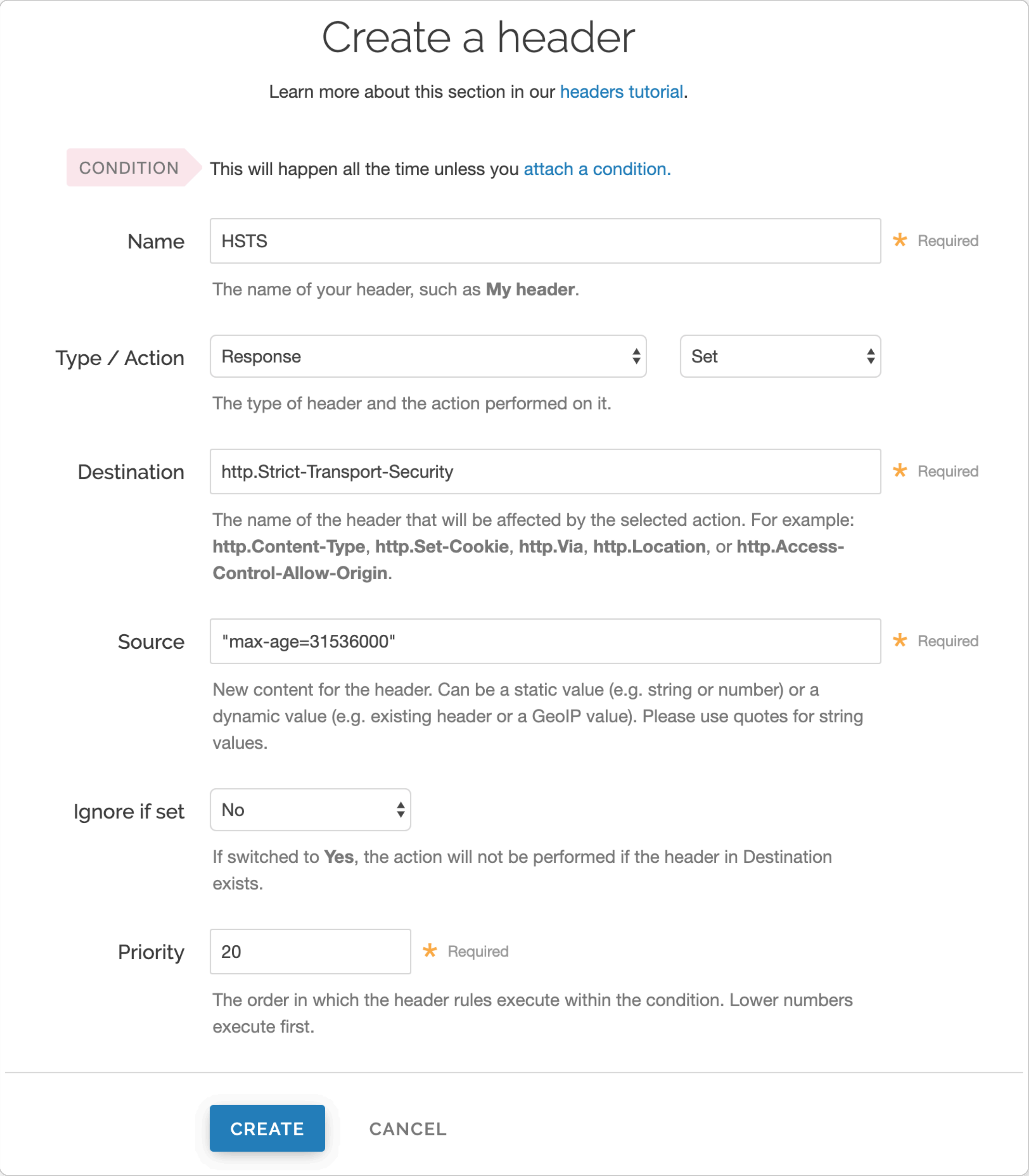
Manually enabling HSTS

If you'd like configure additional HSTS options, you'll need to manually enable HSTS by adding a new header as follows.

NOTE

If you followed the instructions in the previous section, click the **Force TLS and enable HSTS** switch to remove the request setting and header that were automatically created.

1. Follow the instructions in forcing a TLS redirect to force unencrypted requests over to TLS.
2. Click **Content**.
3. Click **Create header**.


Create a header
Learn more about this section in our headers tutorial.
CONDITION
This will happen all the time unless you attach a condition.

Name: HSTS * Required

The name of your header, such as **My header**.

Type / Action: Response | Set

The type of header and the action performed on it.

Destination: http.Strict-Transport-Security * Required

The name of the header that will be affected by the selected action. For example: **http.Content-Type**, **http.Set-Cookie**, **http.Via**, **http.Location**, or **http.Access-Control-Allow-Origin**.

Source: "max-age=31536000" * Required

New content for the header. Can be a static value (e.g. string or number) or a dynamic value (e.g. existing header or a GeoIP value). Please use quotes for string values.

Ignore if set: No

If switched to **Yes**, the action will not be performed if the header in Destination exists.

Priority: 20 * Required

The order in which the header rules execute within the condition. Lower numbers execute first.

CREATE
CANCEL

4. Fill out the **Create a header** fields as follows:
    - In the **Name** field, enter a human-readable name, such as `HSTS`. This name is displayed in the Fastly web interface.
    - From the **Type** menu, select **Response**, and from the **Action** menu select **Set**.
    - In the **Destination** field, enter `http.Strict-Transport-Security`.

  - In the **Source** field, enter `"max-age=<max age in seconds>"`. For example, `"max-age=31536000"`. As described below, `max-age` is required and two additional HSTS options can be specified.
  - Leave the **Ignore if set** menu and the **Priority** field set to their defaults (or set them as appropriate for your service).

5. Click **Create**.
6. Click **Activate** to deploy your configuration changes.

HSTS options

If you manually configured the HSTS header, you can specify additional HSTS options.

HSTS requires the max-age directive be set in order to function properly. It specifies how long in seconds to remember that the current domain should only be contacted over HTTPS. The example shown above sets `max-age` to one year (31536000 seconds = 1 year). You may want to experiment using a smaller value than what is shown.

Two additional options can be specified with the HSTS response header:

- `includeSubdomains` - This token applies HSTS to all of your site's subdomains. Before you include it, be certain none of your subdomains require functionality on HTTP in a browser. Ensure your TLS certificate is a wildcard or has coverage for all subdomain possibilities.

  IMPORTANT

  All subdomains will be unreachable on HTTP by browsers that have seen the HSTS header once `includeSubdomains` is enabled.

- `preload` - This token allows you to submit your domain for inclusion in a preloaded HSTS list that is built into several major browsers. Although the token is not part of the HSTS specification, including it in the header is a prerequisite for submitting to this preloaded list.

  WARNING

  Don't request browser preload inclusion unless you're sure that you can support HTTPS for the long term. Inclusion in the HSTS Preload List cannot be undone easily. See https://hstspreload.org/ for submission instructions and more information.

Combining all of these options together in the **Source** field would look like this:

```
"Strict-Transport-Security: max-age=<max age in seconds>; includeSubDomains; preload"
```

To disable HSTS for whatever reason, simply set the `max-age` to `0` on an HTTPS connection.

The HSTS Preload List is managed by a third party, not by Fastly. Consult https://hstspreload.org/ for more information.

Additional reading

- RFC 6797, which describes the HSTS specification
- the Wikipedia description of HSTS, including the currently known limitations and a browser support list

- the OWASP.org explanation of HSTS, including descriptions of the threats it addresses
- the Chromium Projects description of HSTS and preloading HSTS sites

---

* * *

Enabling TLS 1.3 through Fastly

/en/guides/enabling-tls-1-3-through-fastly

IMPORTANT

TLS 1.3 is now the default TLS version. You only need to follow the instructions in this guide if you created your account on or before June 29, 2020.

This guide describes how to use Fastly TLS to enable TLS 1.3 for a domain using a TLS certificate you provide or one that Fastly provides and manages.

About TLS 1.3

To serve secure, encrypted traffic from Fastly using the Hypertext Transfer Protocol Secure (HTTPS) protocol, a website or application must provide a valid TLS certificate that is digitally signed by a trusted certification authority. Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are the protocols that allow clients to form secure communication connections between web browsers or applications and the servers they request information from.

TLS 1.3, the newest version of the TLS protocol, was designed to improve the performance and security of traffic for HTTPS domains. Specifically, this version of the protocol was designed to help speed up encrypted connections to servers by eliminating an entire round trip from its connection establishment handshake. The Zero Round Trip Time (0-RTT) feature can reduce the latency of resumed connections by encrypting requests in the initial ClientHello, a step in the client handshake process that specifies the maximum protocol version the client wishes to support.

In addition, TLS 1.3 allows only cipher suites that offer Perfect Forward Secrecy (PFS) for securing and encrypting traffic. TLS 1.3 also specifically prohibits TLS renegotiation, a process that allows changing the details of a TLS handshake after a connection has already been established with the server. Both restrictions make TLS 1.3 more secure than previous versions of the protocol.

When to use the web interface and when to contact Support

You can only enable TLS 1.3 via the web interface if you have purchased or are using:

- Fastly TLS,
- Concierge TLS, or
- Fastly's Legacy Customer-Provided TLS Certificate Hosting Service

and your domains have not been configured on Dedicated IP addresses that Fastly maintains and manages for you.

If you have purchased or are using Platform TLS or Dedicated IP addresses, or you have custom domain mapping in place, you must contact support and have them enable TLS 1.3 for you.

Limitations and key behaviors

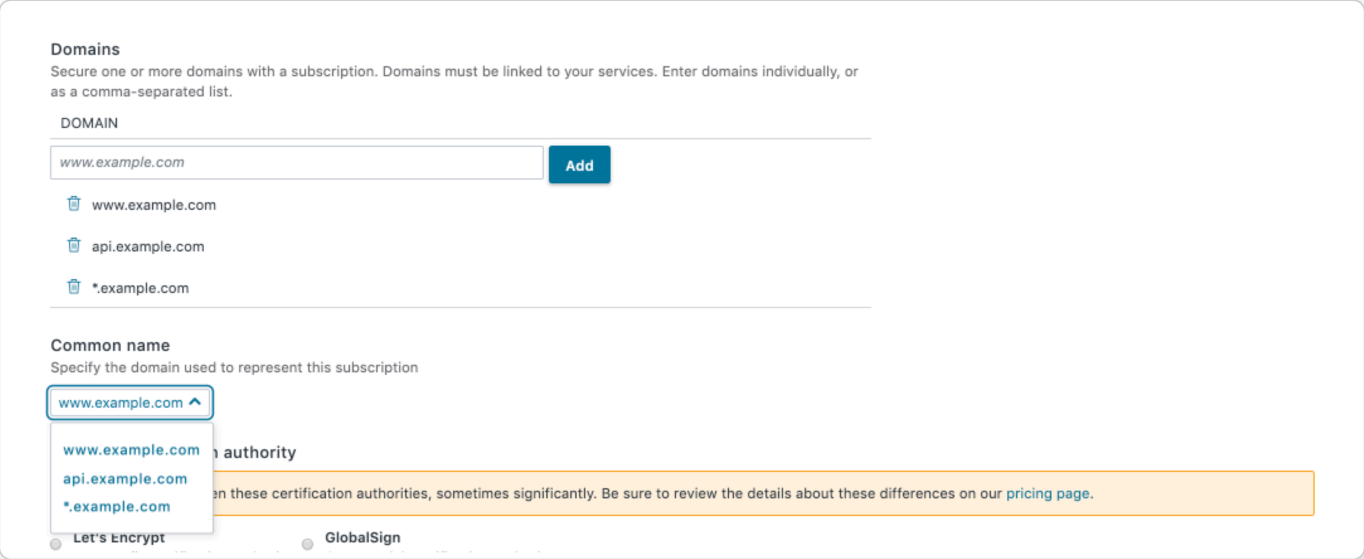
Before enabling or requesting this functionality, keep the following in mind:

- Negotiation of the TLS protocol will only happen if the requesting client also supports TLS 1.3. If a request comes from an older client, Fastly's default behavior is to downgrade to TLS 1.2.
- Fastly currently only supports 0-RTT between Fastly and requesting clients. We do not support 0-RTT between Fastly and your origin servers.
- By default, Fastly only answers idempotent requests (GET and HEAD requests without query parameters) over 0-RTT. This helps protect customer applications from replay attacks.
- Requests issued with 0-RTT will include an `Early-Data:1` header per RFC 8470. This attribute can be queried and logged via VCL using `req.http.early-data`.

Setting up TLS 1.3 for a new domain

Setting up TLS for a domain requires you to secure the domain by registering it with a certification authority. To start this process through Fastly's web interface (instead of programmatically) follow these steps.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Domains**.
3. Click **Secure another domain**.
4. Decide what to do next:
    - If you have your own TLS certificates and private keys, click **Use certificates you've provided** and then follow the instructions in the guide to uploading and deploying your own certificates instead of this one.
    - If you want Fastly to procure and manage your TLS certificates and keys, continue with the remaining steps that follow.
5. From the selection menu that appears, select **Use certificates Fastly obtains for you**.
6. In the **Domain** field, enter one or more apex domains (e.g., `example.com`), subdomains (e.g., `www.example.com` or `api.example.com`), or a wildcard domain (e.g., `*.example.com`) and click **Add**. Domains you add appear in the Common name area of the page.



Domains

Secure one or more domains with a subscription. Domains must be linked to your services. Enter domains individually, or as a comma-separated list.

Common name

Specify the domain used to represent this subscription

…n these certification authorities, sometimes significantly. Be sure to review the details about these differences on our pricing page.

If you only have one domain, the common name will be the same as the domain name. If you add more than one domain, they will appear in a menu. By default, the first domain you add will be selected for you. Select another domain from the **Common name** menu if that's not the one you want.

7. From the **Select a certification authority** controls, choose one of the certification authorities to secure your certificate. Prices vary between certification authorities, sometimes significantly. Be sure to review the details about these differences on our pricing page.

8. If you previously enabled TLS in your Fastly account, use the **Select a TLS configuration** menu to select a TLS 1.3 configuration to apply. Your selection will specify both the IP addresses that the certificate will be deployed to and the associated TLS settings that will be applied to them.

   - Select **HTTP/3 & TLS v1.3** to apply the latest version of the protocol, but without 0-RTT.
   - Select **HTTP/3 & TLS v1.3 + 0RTT** to apply the latest version of the protocol with 0-RTT.

   However, if you are enabling TLS in your Fastly account for the first time on or after March 29, 2022, this menu will not appear. Your TLS configuration will use HTTP/3 & TLS v1.3 + 0RTT by default.

9. Click **Submit**. The Subscription details page appears displaying your domains along with detailed steps on how to verify you own them.

10. Click **View details** to view information for your domain and use it to update your DNS records with your DNS provider.

Applying TLS 1.3 to an existing domain

To migrate an existing domain to a new TLS 1.3 configuration, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Domains**.
3. Find the card for the appropriate domain.
4. Click **View/Edit Activation** next to the appropriate certificate.
5. From the list of configurations that appear, click **Activate** next to the TLS 1.3 configuration you want to apply. Your selection will specify both the IP addresses that the certificate will be deployed to and the associated TLS

settings that will be applied to them.
  - Activate **HTTP/3 & TLS v1.3** to apply the latest version of the protocol, but without 0-RTT
  - Activate **HTTP/3 & TLS v1.3 + 0RTT** to apply the latest version of the protocol with 0-RTT.

6. Click **Done**.
7. Watch the **TLS status** area for the certificate. Once the configuration is selected, the status for the domain will change to `Deploying`. When the TLS status area changes back to `Activated`, the TLS configuration selected will have been applied to the domain.
8. Click **View details** to view information for your domain and use it to update your DNS records with your DNS provider.
9. Confirm the new DNS records have propagated across the internet (this can take up to 48 hours), then delete the old TLS configuration by clicking the trash icon.

* * *

Forcing a TLS redirect

/en/guides/forcing-a-tls-redirect

If you want to only allow TLS on your site, we have you covered. We've built a setting into the request settings that will allow you to force unencrypted requests over to TLS. It works by returning a **301 Moved Permanently** response to any unencrypted request, which redirects to the TLS equivalent. For instance, making a request for `http://www.example.com` would redirect to `https://www.example.com`.

NOTE

Because requests can still happen over HTTP first even if you force a TLS redirect using these instructions, we recommend enabling HSTS as well. Fastly provides a different setting that lets you easily force TLS and enable HSTS at the same time. Alternatively, you can follow these instructions to force a TLS redirect and manually enable HSTS later.

Prerequisites

These instructions assume that you've set up TLS service with Fastly.

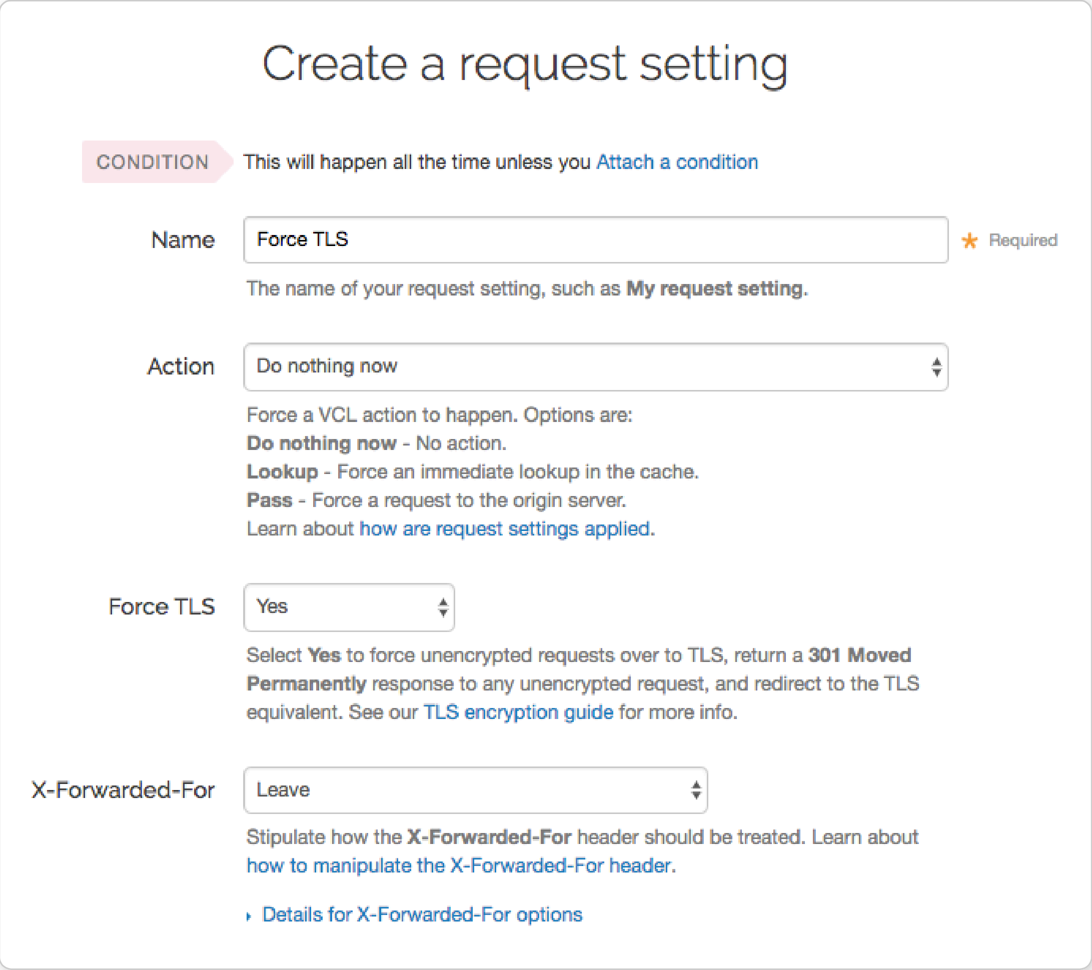
Forcing a TLS redirect

To force a TLS redirect, follow these steps:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.

4. Click **Settings**.

5. Click **Create request setting**.



Create a request setting

CONDITION This will happen all the time unless you Attach a condition

Name: Force TLS ★ Required

The name of your request setting, such as **My request setting**.

Action: Do nothing now

Force a VCL action to happen. Options are:
Do nothing now - No action.
Lookup - Force an immediate lookup in the cache.
Pass - Force a request to the origin server.
Learn about how are request settings applied.

Force TLS: Yes

Select **Yes** to force unencrypted requests over to TLS, return a **301 Moved Permanently** response to any unencrypted request, and redirect to the TLS equivalent. See our TLS encryption guide for more info.

X-Forwarded-For: Leave

Stipulate how the **X-Forwarded-For** header should be treated. Learn about how to manipulate the X-Forwarded-For header.

▸ Details for X-Forwarded-For options

6. Fill out the **Create a request setting** fields as follows:
   - In the **Name** field, enter a human-readable name for the request setting. This name is displayed in the Fastly web interface.
   - From the **Force TLS** menu, select **Yes**.

7. Click **Create** to save your request setting changes.

8. Click **Activate** to deploy your configuration changes.

* * *

Setting up Mutual TLS authentication

/en/guides/setting-up-mutual-tls-authentication

Mutual TLS (mTLS) is an additional layer of network connection security that is added on top of our existing TLS product. By default, the TLS protocol only requires a server to present a trusted certificate to the client. mTLS requires the client to also present a trusted certificate to the server. Instead of having to rely on traditional authentication methods like passwords or API keys, the server to client connection is secured using TLS certificates.

> TIP
>
> Are you looking for information on applying TLS on connections between Fastly and your origin? Refer to our Working with hosts guide.

Prerequisites

To use mTLS, be sure you have the following prerequisites in place:

- a paid account with a contract for Fastly's services.
- an existing TLS activation consisting of valid domains, a TLS configuration with the relevant domains added, and TLS certificate. The certificate may be either Fastly-managed or self-managed.
- a `.pem` file containing one or more certificates certified by a certification authority (CA). This file is used as your chain of trust to verify the client certificates for your connection.

> NOTE
>
> mTLS is supported on Compute services for origins configured via Dynamic Backends. To set up mTLS for Compute services, refer to the developer documentation.

Setting up mTLS for the first time

Setting up mutual TLS authentication consists of uploading an mTLS certificate and defining the domains you want to secure with mTLS. You can enforce an mTLS connection on all requests to your domains, denying a connection if a valid certificate isn't presented. Or, you can allow connections to proceed whether or not the mTLS connection is successful. The latter option let's you secure sensitive communications with mTLS while still allowing less sensitive data to be transmitted over non-mTLS connections, which can be useful as you transition to using mTLS.

To apply mTLS:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Drag and drop your certificate file into the drag and drop area to upload your certificate file. Alternately, click **Browse for certificate file** to navigate to the file on your system using the file picker. The Mutual TLS certificate details page appears.

   > TIP
   >
   > If you are using multiple certificates, combine them into one certificate bundle.

4. In the **Mutual TLS certificate name** field, enter a name used to easily identify the certificate or certificate bundle in the web interface.
5. Do one of the following:
   - Leave the **Require mTLS** checkbox selected to enforce mTLS and only allow a connection when mTLS authentication is successful.

   - Deselect the checkbox to allow a connection to proceed even if mTLS authentication fails. Useful when transitioning to using mTLS and required to log or track requests sent without a client certificate.

6. Click **Save and next** to continue.
7. From the **Add domains** menu, select the active domains you want mTLS applied to. You can use the search box to search for domains by name, certificate, or TLS configuration.
8. Click **Done**. A card for the new mTLS configuration is added to the Mutual TLS page.

Uploading additional mTLS certificates

You can upload additional mTLS certificates to apply mutual TLS authentication to your domains.

To upload additional certificates:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click **Upload mutual TLS certificate**.
4. Navigate to the file on your system using the file picker. The Mutual TLS certificate details page appears.
5. In the **Mutual TLS certificate name** field, enter a name used to easily identify the certificate in the web interface.
6. Leave the **Require mTLS** checkbox selected to enforce mTLS and only allow a connection when mTLS authentication is successful. Deselect the checkbox to allow a connection to proceed even if mTLS authentication fails.
7. Click **Save and next** to continue.
8. From the **Add domains** menu, select the active domains you want mTLS applied to. You can use the search box to search for domains by name, certificate, or TLS configuration.
9. Click **Done**. A card for the new mTLS configuration is added to the Mutual TLS page.

Adding and removing domains

From the mTLS certificate details page, you can edit the domains on which mTLS is enforced.

To add domains:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click **View certificate details**.
4. From the **Add domains** menu, select the active domains you want mTLS applied to. You can use the search box to search for domains by name, certificate, or TLS configuration.
5. Click **Done** to save your changes.

To remove domains:

1. Log in to the Fastly web interface.

2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click **View certificate details**.
4. Click the trash next to the domain you want to remove.
5. Click **Done** to save your changes.

Editing Mutual TLS certificate details

From the mTLS certificate details page, you can edit the authentication name and the mTLS enforcement option.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click **View certificate details**.
4. Click **Back to certificate settings**.
5. In the **Mutual TLS certificate name** field, enter a name used to easily identify the certificate in the web interface.
6. Use the **Require mTLS** checkbox to determine whether mTLS is enforced. If selected, connections are only allowed when mTLS authentication is successful. If de-slected, connections proceed even if mTLS authentication fails.

Replacing an mTLS certificate

From the Mutual TLS page, you can replace the certificate used for mTLS.

To replace the certificate:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click **Replace** on the card for the mTLS configuration you want to update.
4. Drag and drop your certificate file into the drag and drop area to upload your certificate file. Alternately, click **Browse for certificate file** to navigate to the file on your system using the file picker.
5. Click **Submit** to save your changes.

Deleting an mTLS authentication

To delete an mTLS configuration, you must ensure there are no active domains on the mutual authentication. If there are, edit the configuration to remove the active domains before proceeding.

To delete an mTLS configuration:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Mutual TLS**.
3. Click the trash on the card for the mTLS configuration you want to update.
4. Confirm that you want to delete the mutual authentication and then click **Delete**.

* * *

Setting up TLS with certificates Fastly manages

/en/guides/setting-up-tls-with-certificates-fastly-manages

This guide describes how to use Fastly TLS to enable HTTPS for a domain using a certificate managed by Fastly. To serve secure traffic from Fastly using HTTPS, a website or application needs to provide clients with a valid TLS certificate signed by a trusted certification authority. TLS (Transport Level Security) and its predecessor SSL (Secure Sockets Layer) are the protocols that allow clients to form secure server connections so traffic can be served over HTTPS.

Fastly-managed certificates use the ACME protocol to procure and renew TLS certificates. You have several options for certification authorities:

- Certainly, Fastly's publicly-trusted certification authority
- Let’s Encrypt, a third-party non-profit certification authority
- GlobalSign, a third-party commercial certification authority (only available for paid accounts)

Trial accounts include Fastly-managed certificates for two domains using the Certainly or Let's Encrypt certification authority. Upgrade to a paid account to use GlobalSign or secure additional domains.

✓ TIP

Our TLS subscriptions API allows you to manage Fastly TLS subscriptions programmatically.

Before you begin

Before setting up TLS on your domains, be sure to review the Fastly TLS prerequisites and limitations.

Setting up TLS for a domain

Setting up TLS for a domain requires you to secure the domain by registering it with a certification authority. To start this process through Fastly’s web interface (instead of programmatically) follow these steps.

Setting up TLS for the first time

To set up TLS for the first time, complete the following:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. Click **Get started**.
4. In the **Domain Name** field, enter the apex domain (e.g., `example.com`), subdomain (e.g., `www.example.com` or `api.example.com`), or wildcard domain (e.g., `*.example.com`) you want to secure.

5. From the **Certification Authority** menu, select one of the certification authorities to secure your certificate. Prices vary between certification authorities, sometimes significantly. Be sure to review the details about these differences on our pricing page.

6. From the **Select a TLS Configuration** menu, select the TLS configuration to apply. The configuration defines both the IPs that the certificate will be deployed to and the associated TLS settings that will be applied. The default option is **HTTP/3 & TLS v1.3 +0RTT (t.sni)**.

7. Click **Continue**. The Domains page appears displaying your domain along with detailed steps on how to verify that you own it.

Setting up TLS for subsequent domains

After you've set up TLS for your first domain, you can secure multiple additional domains from the Domains page.

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.

2. Click **Manage certificates**.

3. Click **Secure another domain**.

4. In the **Domain** field, enter one or more apex domains (e.g., `example.com`), subdomains (e.g., `www.example.com` or `api.example.com`), or a wildcard domain (e.g., `*.example.com`) and click the **Add** button. Domains you add appear in the Common name area of the page.

   If you only have one domain, the common name will be the same as the domain name. If you add more than one domain, they will appear in a menu. By default, the first domain you add will be selected for you. Select another domain from the **Common name** menu if that's not the one you want.

5. From the selection menu that appears, select **Use certificates Fastly obtains for you**. The Enter subscription details page appears.

6. From the **Select a certification authority** controls, choose one of the certification authorities to secure your certificate. Prices vary between certification authorities, sometimes significantly. Be sure to review the details about these differences on our pricing page.

7. From the **Select a TLS Configuration** menu, select the TLS configuration to apply. The configuration defines both the IPs that the certificate will be deployed to and the associated TLS settings that will be applied. The default option is **HTTP/3 & TLS v1.3 +0RTT (t.sni)**.

8. Click **Submit**. The Subscription details page appears displaying your domains along with detailed steps on how to verify you own them.

Verifying domain ownership

To begin serving HTTPS traffic, Fastly needs to verify that you control any domain you’ve added to the web interface. Fastly allows you to verify apex domains and subdomains via the ACME DNS challenge, the ACME HTTP challenge, or via email validation. Each requires you to make specific DNS changes. Wildcard domains require the DNS challenge or email validation challenge type.

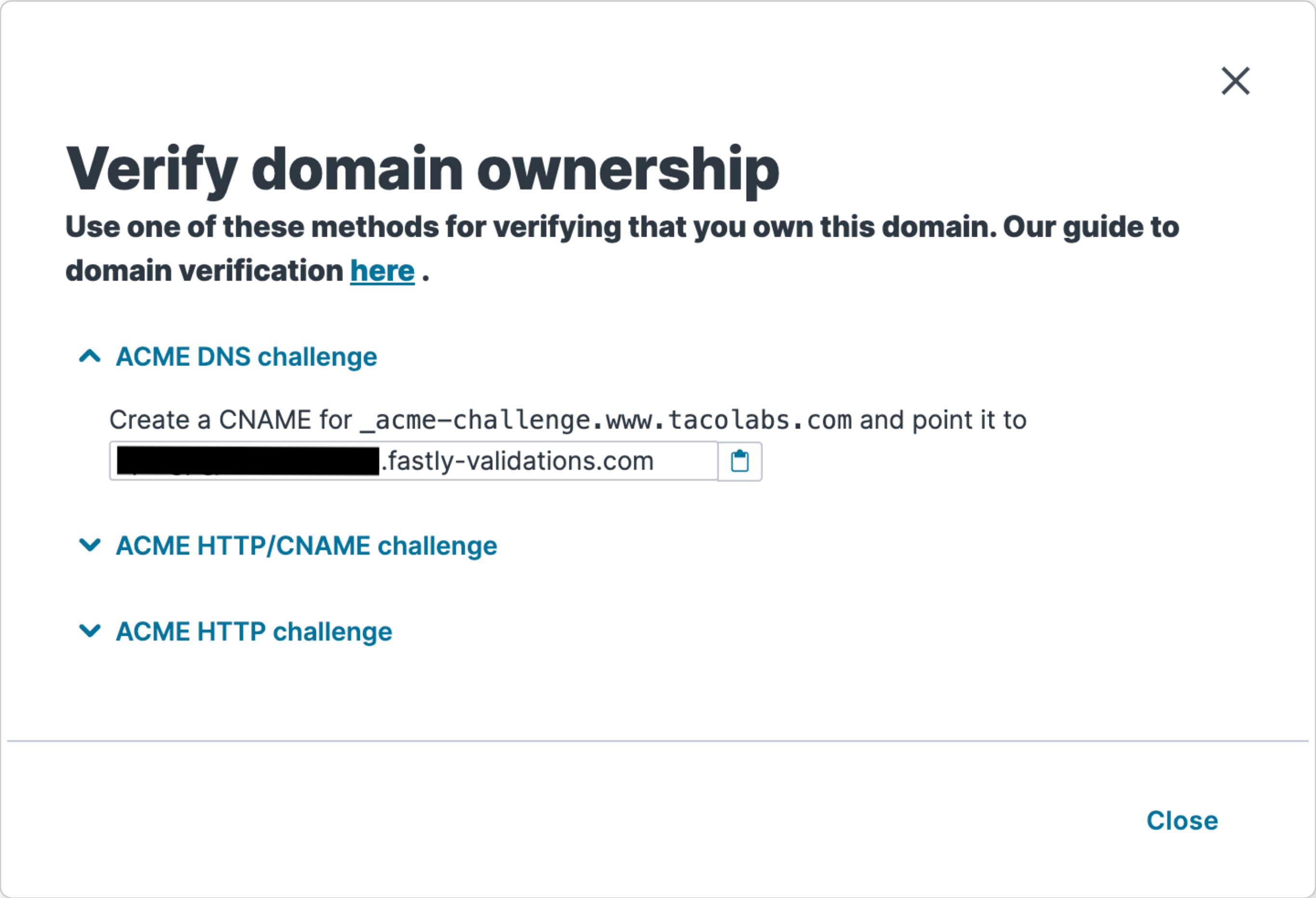
> IMPORTANT
>
> Fastly may modify the behavior of your services and DNS to complete ACME challenges for domain verification.

Using the ACME DNS challenge to verify domain ownership

The default method for verifying you control a domain being added to a Fastly managed TLS certificate uses the ACME DNS challenge type. It's suitable for all kinds of domains (apex, subdomains, and wildcards). It will only point the `_acme-challenge` subdomain at Fastly, allowing you to set up TLS first, before pointing production traffic at Fastly.

To use this verification method, create a CNAME record with a unique target for your domain. Follow the steps below to view the formats for the CNAME record and target.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Subscriptions**.
3. Click **View subscription details** for the subscription you want to make changes to.
4. Click **Verification options** in the **Verification** column for the domain you want to verify. The Verify domain ownership page appears with formats for the record and target.



Verify domain ownership

Use one of these methods for verifying that you own this domain. Our guide to domain verification here .

ACME DNS challenge

Create a CNAME for `_acme-challenge.www.tacolabs.com` and point it to

[redacted].fastly-validations.com

The steps to create the CNAME record will vary depending on your DNS provider's control panel interfaces. Refer to your DNS provider's documentation for exact instructions on how to do this. Your CNAME record must use the format `_acme-challenge.DOMAIN_NAME` (e.g., `_acme-challenge.www.example.com`) and must be pointed to a unique target for your domain (e.g., `domain_token.fastly-validations.com`). Once you've pointed your DNS records at Fastly, we encourage you to keep the `_acme-challenge` subdomain CNAME in place to avoid interruptions in service.

Using the ACME HTTP challenge to verify domain ownership

Another method for verifying you control a domain uses the ACME HTTP challenge. This method is only suitable for apex domains and subdomains (wildcard domains can only be verified using DNS or email challenges). It will point production traffic immediately at Fastly.

> ⚠ WARNING
>
> The ACME HTTP challenge domain verification method can potentially point all end-user traffic to Fastly before you've completed TLS setup. This means that your end users may get an insecure warning in their browser related to your website or be totally unable to access your domain. To avoid this, ensure you have a properly configured Fastly service and have disabled any forced TLS redirection before you use this verification method.

To use this verification method, follow the steps below to view the formats for the CNAME record and target.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Subscriptions**.
3. Click **View subscription details** for the subscription you want to make changes to.
4. Click **Verification options** in the **Verification** column for the domain you want to verify.
5. Choose the verification alternative that suits your needs for the ACME HTTP challenge:
    - for a subdomain, create a CNAME record that points directly to the Fastly hostname
    - for an apex domain, create A records for the domain with the noted IP addresses

Once set up using either alternative, production traffic will immediately begin flowing through Fastly.

Using an email challenge to verify domain ownership

Domain control can also be verified via email when you've chosen GlobalSign as your certification authority. (Let's Encrypt does not support email challenges for domain verification.) To use this verification method, follow the steps below.

1. Contact support to enable the setting for verifying domain control via email.
2. Log in to the Fastly web interface.
3. Go to **Security** > **TLS management** > **Subscriptions**.
4. Click **View subscription details** for the subscription you want to make changes to.
5. Click **Verification options** in the **Verification** column for the domain you want to verify.
6. In the **Email validation** section, use the menu to select the email address you want email verification sent to. When selected as the certification authority, GlobalSign will provide Fastly with a list of acceptable email addresses to which a verification email can be sent. Generally, the list will include email addresses like the following:
    - `admin@example.com`
    - `administrator@example.com`
    - `hostmaster@example.com`
    - `postmaster@example.com`
    - `webmaster@example.com`

7. Click **Get verification email**.

Fastly will then instruct GlobalSign to send a verification email to the address you specify. It will contain a link that you must click to complete the domain ownership verification process.

What happens next

It should take no more than an hour for the TLS enablement process to progress through all of the TLS statuses shown below:

TLS Status	Description
Checking domain DNS records... Step 1 of 3	Domain validation is in progress. Fastly is checking domain DNS records to verify that you control the domain being added to a certificate. To advance to the next enablement state, you must verify control of the domain by updating that domain's DNS records to complete one of the ACME challenge types.
Certificate requested. Waiting for response from CA... Step 2 of 3	Domain validation has been confirmed. Fastly has verified you control the domain and has requested a TLS certificate for it from the certification authority.
TLS enabled (certificate being deployed globally)	The certification authority has issued a TLS certificate. Newly issued certificates can take between 20 minutes to an hour to fully deploy across Fastly's global network.

NOTE

If using GlobalSign certificates, after verifying domain ownership you must monitor the DNS resolution process and ensure the domain resolves globally before the TLS certificate will validate and issue. Depending on your DNS vendor, it may take up to 72 hours for your DNS to resolve.

Troubleshooting

If more than an hour has passed and TLS enablement appears to be stalled in one of the stages of adding a domain, there is likely an issue.

Domains stuck in the Checking domain DNS records state

If the domain is stuck in the `Checking domain DNS records` state, it is likely that you have not configured your DNS records correctly in order to verify domain ownership. You can check the DNS records yourself using a `dig` command in a command line application as follows:

ACME challenge type	Command to type
HTTP	`dig www.example.com +short`
DNS	`dig _acme-challenge.www.example.com +short`

Be sure to replace `example.com` with the hostname you used when you configured your DNS records.

If you have correctly configured your DNS records, the result from this command will include one of the CNAME or A Records required for verification as defined in the Verifying domain ownership instructions.

If you recently added or modified DNS records, you may need to wait up to 72 hours for your DNS changes to propagate across the internet. If you don't see these addresses within that time period, you may have misconfigured your DNS records.

If you are still having issues, there may be a Certification Authority Authorization (CAA) record on your domain that is blocking the certification authority from issuing certificates. This CAA record is used to specify which certification authorities (CAs) are allowed to issue certificates for a domain. If a CAA record exists, you may need to correct or remove this record in order to use a managed Fastly TLS certificate.

The following lists the CAA record value needed for each certification authority supported by Fastly TLS:

Certification authority	CAA record value
Certainly	`certainly.com`
Let's Encrypt	`letsencrypt.org`
GlobalSign	`globalsign.com`

Domains stuck in the Certificate requested state

If the domain is stuck in the `Waiting for a response from CA` state, this is likely a temporary issue with the certification authority. Be sure to allow up to an hour in this state before contacting support for assistance. If this is a new certificate request, you can also try deleting the domain and starting again.

TLS activated but certificate not deployed everywhere

If the domain displays the `Activated` state but the certificate doesn't appear to be available everywhere, the certificate is likely still in the process of being deployed throughout the Fastly network. It can take anywhere from 20 minutes to an hour for certificates to fully deploy. Be sure to allow up to an hour in this state before contacting support for assistance.

Subscriptions listed as failed

A subscription displays the `Failed` state if it is a new subscription and fails to issue a certificate for seven days or an existing subscription that fails to renew a certificate for seven days beyond its expiration date. You will also receive an email notification about failed subscriptions.

Subscriptions typically fail if you are not pointing traffic to Fastly or if you don't have proper DNS records in place. GlobalSign subscriptions specifically may fail if the domain hasn't resolved globally, meaning it's not accessible from anywhere in the world. Use a tool like DNS Checker to ensure your domain is propagated globally.

Once you resolve these issues, you can try to re-issue the subscription using the steps below:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Subscriptions**.
3. Click **View subscription details** for the failed subscription.
4. Click **Retry now**.

After you retry a failed subscription, we will try to issue or renew the associated certificate for a 24-hour period. If we still cannot obtain a certificate after 24 hours, the subscription will again be marked as failed.

Pointing DNS to serve HTTPS traffic

To serve secure traffic via HTTPS once the certificate is deployed, follow these steps.

1. Ensure that the domains you've added via the TLS domains interface have been added to a properly configured Fastly service.
2. Configure your DNS records to point traffic at the newly created certificate’s IP addresses. If you used the HTTP challenge method to verify domain ownership, you’re already pointing traffic at the certificate. All DNS details (CNAME, A records, and optionally AAAA records) can be found by clicking **See DNS details** to view the TLS configuration associated with the domain.

   For an apex domain (e.g., `example.com`), you'll need to create an A record with your DNS provider. For subdomains and wildcard domains (e.g, `www.example.com` or `*.example.com`), you'll need to create a relevant CNAME record.

Your domains and certificates can be set to use one or more TLS configurations. For more information, refer to the details on managing DNS and TLS configurations.

> ⚠ WARNING
>
> If you point your DNS away from Fastly after the initial setup, we will be unable to terminate TLS on your behalf and we will also be unable to renew your certificate, which will expire after 90 days.

Managing TLS subscriptions

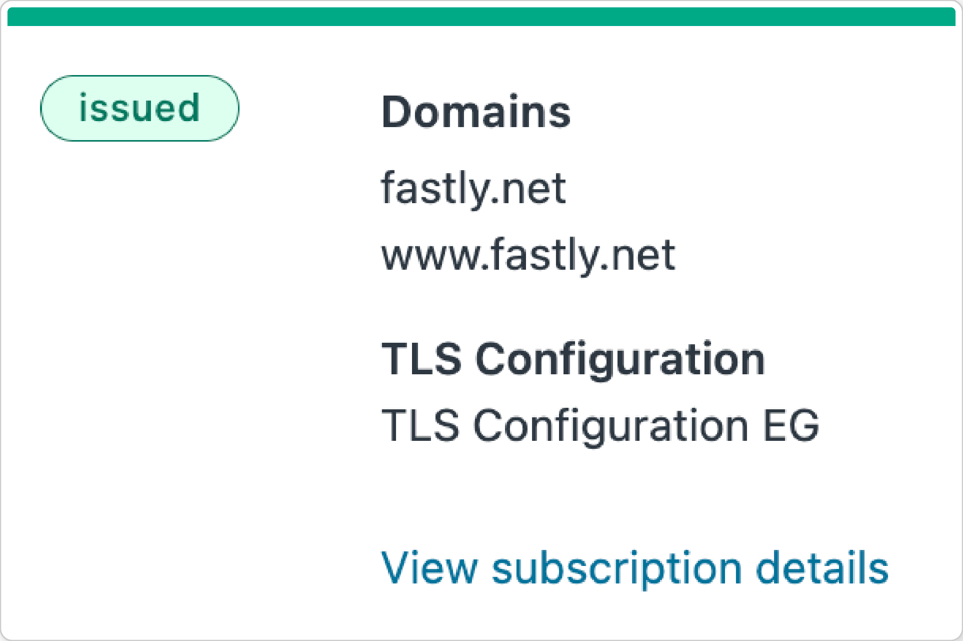
You can use the Subscriptions page to manage your TLS subscriptions and take actions such as:

- adding or removing domains
- changing the domain common name
- migrating to Certainly, Fastly's certification authority
- deactivating TLS

Managing domains on TLS subscriptions

To manage the domains on your TLS subscriptions, such as to add or remove domains or to change the domain common name, follow these steps:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Subscriptions**.
3. Click **View subscription details** for the subscription you want to make changes to.


issued

Domains

fastly.net

www.fastly.net

TLS Configuration

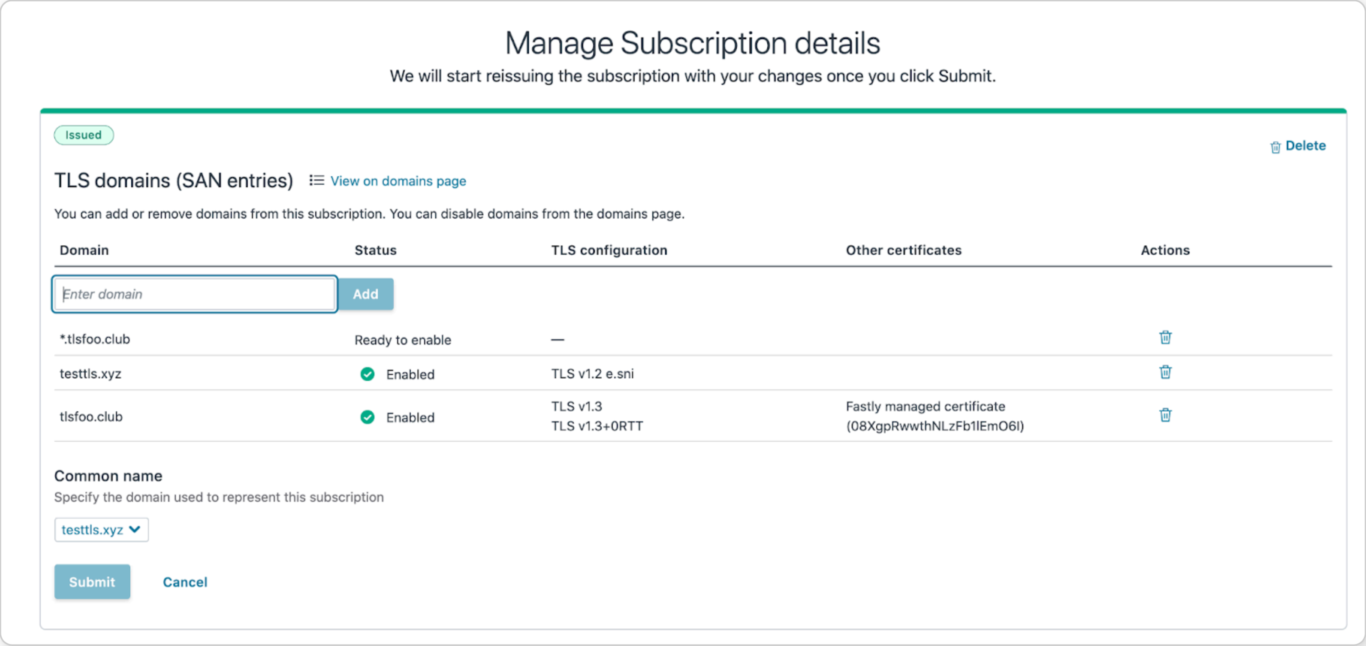
TLS Configuration EG

View subscription details

4. Click **Manage Subscription**.

> NOTE
>
> You can only manage a subscription in `Pending`, `Issued`, and `Renewing` states.

5. From the Manage Subscription details page, you can do the following:


Manage Subscription details

We will start reissuing the subscription with your changes once you click Submit.

Issued

Delete

TLS domains (SAN entries) View on domains page

You can add or remove domains from this subscription. You can disable domains from the domains page.

Domain	Status	TLS configuration	Other certificates	Actions
Enter domain Add				
*.tlsfoo.club	Ready to enable	—		🗑
testtls.xyz	✓ Enabled	TLS v1.2 e.sni		🗑
tlsfoo.club	✓ Enabled	TLS v1.3 TLS v1.3+0RTT	Fastly managed certificate (08XgpRwwthNLzFb1lEmO6I)	🗑

Common name

Specify the domain used to represent this subscription

testtls.xyz

Submit Cancel

- **Add new domains:** In the **Domain** field, enter one or more apex domains (e.g., `example.com`), subdomains (e.g., `www.example.com` or `api.example.com`), or a wildcard domain (e.g., `*.example.com`) and click **Add**. Separate multiple domains with a comma.
- **Remove existing domains:** Click the trash in the **Actions** column on the same line as the domain you want to delete. Follow the instructions in the confirmation window to complete the deletion.

  - **Change the subscription common name:** From the **Common name** menu, select the domain used to represent this subscription.

6. After making any changes to the subscription, click **Submit**. A message appears asking to confirm your changes.
7. Click **Confirm changes** to submit your changes. To return to the previous screen, click **No, review changes**.

> NOTE
>
> If you added a new domain, you must validate domain ownership after confirming the change. See Domain validation for TLS certificates.

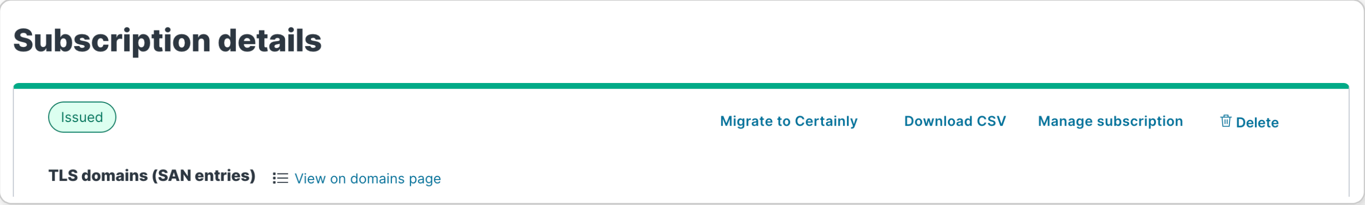
Migrating to Certainly

If you're using the Let's Encrypt or GlobalSign certification authorities to secure your domains, you can use the web interface to migrate to Certainly, Fastly's publicly-trusted certification authority.

> NOTE
>
> To migrate from GlobalSign to Certainly, first contact Fastly Support.

To migrate to Certainly:

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Subscriptions**.
3. Click **View subscription details** for the subscription you want to make changes to.
4. Click **Migrate to Certainly**.



5. Confirm that you want to migrate your certificate, then click **Migrate to Certainly**.

After clicking migrate, Fastly will automatically replace your certificate with a Certainly certificate and verify domain ownership without any downtime. Fastly will continue to serve your existing certificate until it is replaced by Certainly. If there were changes to your DNS records that prohibit Certainly from validating your domain and Fastly can't automatically verify the ownership, a notice will appear with instructions you'll need to make to your DNS details.

Deactivating TLS and deleting a TLS domain

Once a domain has TLS activated, you have the option to deactivate TLS via the **Deactivate TLS** button listed on each domain card on the TLS domains page. If a domain has multiple certificates, you can elect to deactivate a specific certificate by clicking **Add/Edit Activations** and clicking the **Deactivate** button next to any active configurations. If all certificates are deactivated, Fastly will no longer serve TLS traffic on the selected domain and it will become disabled. Fastly will attempt to renew a certificate for a disabled domain. To prevent this renewal process, delete the associated subscription after you disable it. Fastly will not renew certificates for deleted subscriptions.

Certificate management and renewals

Each certification authority has a separate verification and renewal time frame that Fastly follows when managing your certificates:

- **Certainly renewals.** Certainly issues certificates that are valid for 30 days. Fastly will attempt to re-verify your domain and renew your certificate after 20 days. However, if your DNS records no longer point at Fastly or if a CAA record blocks Certainly, the certificate will lapse at the end of the 30-day period.
- **Let's Encrypt renewals.** Let’s Encrypt issues certificates that are valid for 90 days. Fastly will attempt to re-verify your domain and renew your certificate after 60 days. However, if DNS records no longer point at Fastly or if a CAA record blocks Let's Encrypt, the certificate will lapse at the end of the 90-day period.
- **GlobalSign renewals.** GlobalSign issues certificates that are valid for 365 days. Fastly will attempt to re-verify your domain and renew your certificate after 275 days. However, if DNS records no longer point at Fastly, or if a CAA record blocks GlobalSign, the certificate will lapse at the end of the 365-day period. Certificates provided by GlobalSign are subject to the terms of GlobalSign's Subscriber Agreement, which can be found at https://www.globalsign.com/repository.

Fastly automatically runs a DNS check for certificate renewals within the following timeframes:

- **Certainly:** 10 days before certificates are due to expire
- **Let's Encrypt:** 30 days before certificates are due to expire
- **GlobalSign:** 90 to 120 days before certificates are due to expire

If a DNS check indicates that a renewal is failing, Fastly will automatically email all account users with TLS management permissions, notifying them of the upcoming expiration. If the renewal continues to fail, Fastly will continue to email users on the account on a schedule up until the expiry date.

In addition, you must verify domain ownership as part of the management process. If you have the correct DNS records for verifying domain ownership and there is no blocking CAA record, but you are still receiving renewal failure emails, contact support for assistance.

* * *

Setting up TLS with your own certificates

/en/guides/setting-up-tls-with-your-own-certificates

This guide describes how to use the Fastly TLS product to upload and deploy your own TLS certificates and private keys using the Fastly web interface.

To serve secure traffic from Fastly using HTTPS (Hypertext Transfer Protocol Secure), a website or application needs to provide clients with a valid TLS certificate signed by a trusted Certification Authority (CA). TLS (Transport Level Security) and its predecessor SSL (Secure Sockets Layer) are the protocols that allow clients to form secure server connections so traffic can be served over HTTPS.

> ✓ TIP
>
> Fastly offers an API for uploading and managing your keys and certificates used to activate TLS for your domains on Fastly.

Before you begin

Before setting up TLS on your domains, be sure to review the Fastly TLS prerequisites and limitations.

Setting up TLS for the first time

To set up TLS, you'll upload a TLS certificate and the matching private key (used to initially generate the certificate). Then, you'll apply a TLS configuration to the domain and activate TLS on the domain.

To set up TLS for the first time, complete the following:

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. Click **Get started**.
4. Click **Upload my own private key and certificate** to continue.
5. Drag and drop the key file into the drag and drop area for key files. Alternately, click the browse link to navigate to the file on your system using the file picker. A success message appears along with a visualization of your key.

   > IMPORTANT
   >
   > The key file upload tool currently only accepts 256-bit or 384-bit ECDSA keys or 2048-bit RSA keys.

6. Drag and drop the certificate file into the drag and drop area for certificate files. Alternately, click the browse link to navigate to the file on your system using the file picker.
7. Click **Continue** to move to the final steps of enabling TLS on your domain.
8. Select the checkbox next to the domain you want to activate.
9. From the **Select a TLS Configuration** menu, select the TLS configuration to apply. The configuration defines both the IPs that the certificate will be deployed to and the associated TLS settings that will be applied. The default option is **HTTP/3 & TLS v1.3 + 0RTT (t.sni)**.
10. Click **Activate**.

Securing additional domains

After you've set up TLS for your first domain, you can upload additional TLS certificates and private keys and activate TLS on other domains.

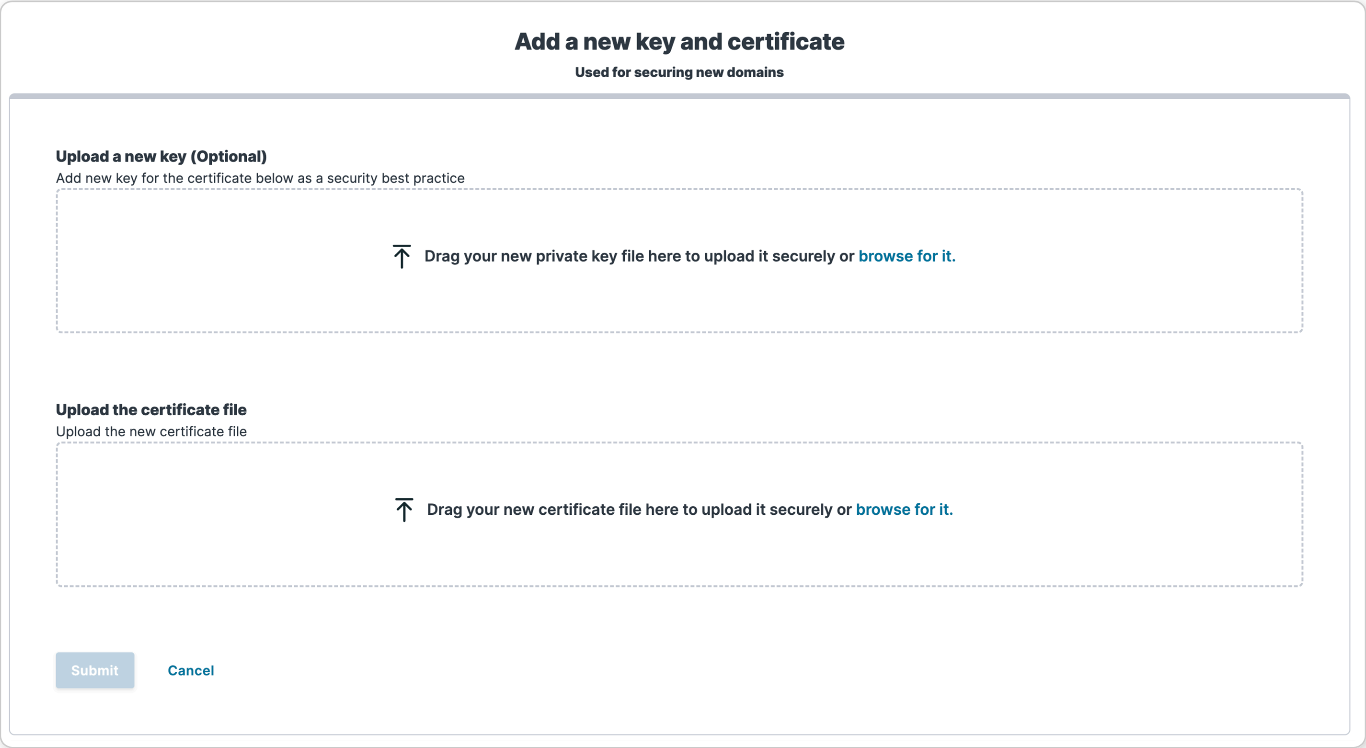
Uploading a private key and certificate

To upload a TLS certificate and the relevant private key, follow the steps below.

> IMPORTANT
>
> The key file upload tool currently only accepts 256-bit or 384-bit ECDSA keys or 2048-bit RSA keys.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Self-managed certificates**.

3. From the **Upload key or certificate** menu, select **Add a new key or certificate**.



Add a new key and certificate

Used for securing new domains

Upload a new key (Optional)

Add new key for the certificate below as a security best practice

Drag your new private key file here to upload it securely or browse for it.

Upload the certificate file

Upload the new certificate file

Drag your new certificate file here to upload it securely or browse for it.

4. We recommend generating a new key for the new certificate. Drag and drop the key file into the drag and drop area for key files. Alternately, click the browse link to navigate to the file on your system using the file picker. If you have multiple private keys, you can identify each by the unique SHA1 hash.

5. Drag and drop the certificate file into the drag and drop area for certificate files. Alternately, click the browse link to navigate to the file on your system using the file picker.

6. Click **Submit**. The Certificate details page appears.

Once you upload your certificate file, you can activate TLS on a domain.

Activating TLS on a domain

Once a valid certificate and private key have been uploaded, all domains that appear as SAN entries will be listed on the Domains page with a status of TLS ready. To serve HTTPS traffic using your certificate, follow the steps below to activate TLS for the domain and point the DNS records to the certificate's location.

1. Log in to the Fastly web interface.

2. Go to **Security** > **TLS management** > **Domains**.

3. Find the card for the domain with the certificates on which you want to activate TLS. Certificates in a disabled state will have the status of `Ready to activate`.

4. Click **Add TLS activation** to the right of the certificate you want to activate.

5. Select the TLS configuration you want to apply. If the configuration is already activated for a different certificate on the domain, a notice appears. You must click **Switch to this certificate** to continue.

   Fastly deploys your TLS certificate to the entire Fastly edge network. It may take up to an hour for your certificate to become available throughout the world.

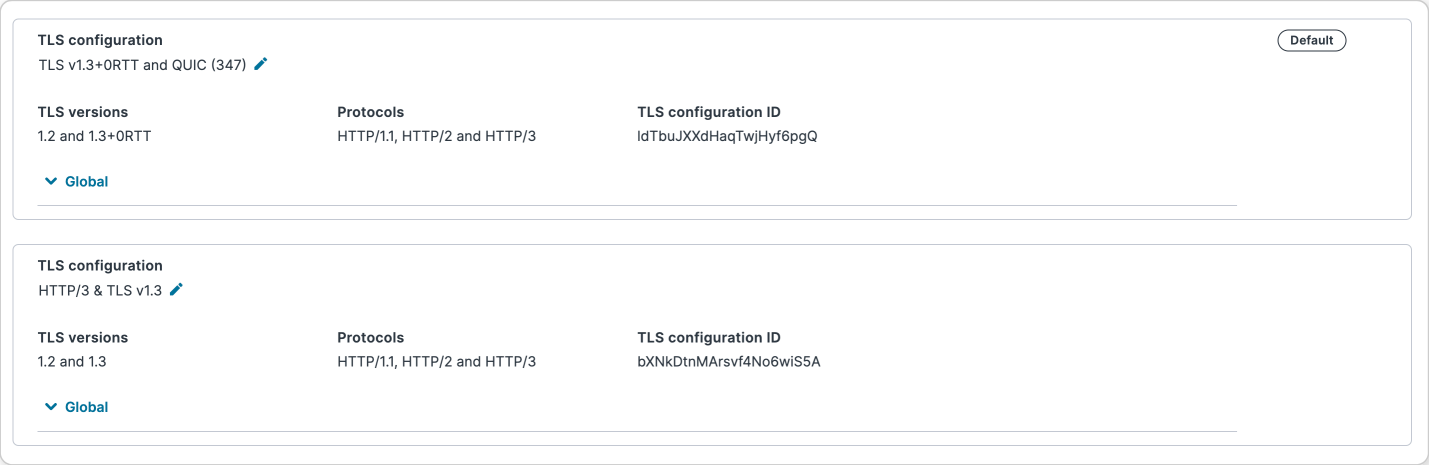
6. Click **View details** to view the DNS details for the domain. Use these details to configure your DNS records so that a TLS connection can be established using your certificate.
    - For TLS on an apex domain (e.g., `example.com`), you'll need to create an A record with your DNS provider.
    - For subdomains and wildcard domains (e.g, `www.example.com` or `*.example.com`), you'll need to create a relevant CNAME record.

IMPORTANT

It can take up to 48 hours for new DNS records to propagate across the internet.

Applying a TLS configuration to a domain

TLS configurations are a collection of TLS settings that include the supported versions of TLS and HTTP, along with networking and handshaking options that clients will use. For accounts with more than one TLS configuration, the default configuration has a label in the right corner of the card.



TLS configuration
TLS v1.3+0RTT and QUIC (347)

TLS versions	Protocols	TLS configuration ID
1.2 and 1.3+0RTT	HTTP/1.1, HTTP/2 and HTTP/3	IdTbuJXXdHaqTwjHyf6pgQ

TLS configuration
HTTP/3 & TLS v1.3

TLS versions	Protocols	TLS configuration ID
1.2 and 1.3	HTTP/1.1, HTTP/2 and HTTP/3	bXNkDtnMArsvf4No6wiS5A

TIP

TLS configuration names are editable by clicking the pencil icon next to the name.

To override the default TLS configuration applied to a domain or to migrate a domain to use a different configuration follow these steps.

1. Log in to the Fastly web interface.
2. Go to **Security** > **TLS management** > **Domains**.
3. Find the card for the domain with the certificates on which you want to add additional TLS activations.
4. Click **View details** next to the certificate on which you want to add additional activations.
5. Click **Add TLS activation**.
6. Select the TLS configuration you want to apply to the certificate.

> NOTE
>
> While you may have multiple certificates on a given domain, only one certificate can be active for a given TLS configuration. If the TLS configuration is already in use by another certificate, a **Switch to this certificate** button appears.

Once the configuration is selected, the TLS configuration is applied to the domain. Each TLS configuration is active and available at their respective IP addresses and DNS records.

7. Click **View details** and use the information to configure your DNS records so that a TLS connection can be established using your certificate.
8. Confirm the new DNS records have propagated across the internet (this can take up to 48 hours), then delete the old TLS configuration by clicking the trash.

> NOTE
>
> For HTTP/1.1, be sure to activate TLS for each of your domains in the web interface. If you upload a certificate with multiple SANs, each domain must have TLS explicitly activated if you want to secure these domains on Fastly.
>
> Exceptions may apply in the case of HTTP/2 if your browser coalesces secure connections and has previously received a TLS certificate from an earlier handshake. In this case, some browsers may reuse an existing secure connection to Fastly if its certificate has a matching SAN entry.

Updating a certificate

Fastly allows you to update a certificate by replacing it with a new one at any time. To help identify certificates that need updating and replacing, the following alerts appear on the certificate cards on the TLS certificates page:

- **Expiring:** Indicates a certificate is nearing its expiration date.
- **Expired:** Indicates a certificate is past its expiration date.
- **Replace:** Indicates a certificate Fastly recommends you update and replace.

Updating certificates when there are no removed domains

To update a TLS certificate by replacing it with one that contains all the domains as the original (either as a superset or a matching list) follow these steps.

1. Generate a new certificate with your preferred Certification Authority.
2. Log in to the Fastly web interface.
3. Go to **Security** > **TLS management** > **Self-managed certificates**.
4. Find the card for the certificate you're replacing.
5. Click the word **Replace** in the upper right corner of the certificate's card.
6. Drag and drop the replacement certificate file into the drag and drop area for certificate files. The certificate you select should be PEM-formatted and the SAN entries of this new certificate must contain all entries in the current certificate (i.e., it must either have an exact matching list or contain a superset). Alternately, click the browse link to navigate to the file on your system using the file picker.

7. If a new key was generated with the new certificate, drag and drop the key file into the drag and drop area for key files. Alternately, click the browse link to navigate to the file on your system using the file picker.
8. Below the drag and drop areas, verify that the certificate replacement menu is pre-selected with the certificate you want to replace with your new certificate.
9. Click **Submit**.

All domains actively serving TLS traffic on the old certificate will be automatically transitioned to the updated certificate within a matter of minutes. Any new domains will need to be manually activated by following the steps for activating TLS on a domain.

Updating certificates when there have been changes to domains

If you want to update one of your certificates that requires removing domains, you will need to procure a new certificate with the updated list of SAN entries. Follow the steps to upload this certificate as a new certificate.

1. Upload the new certificate.
2. Go to **Security** > **TLS management** > **Domains**.
3. Locate the previously existing domains that have already been activated with the certificate you're replacing.
4. Click **View details** next to the certificate on which you want to add additional activations.
5. Click the **Add TLS activation** button.
6. Select the TLS configuration you want to apply to the certificate. Once the configuration is selected, the TLS configuration is applied to the domain. Each TLS configuration is active and available at their respective IP addresses and DNS records.
7. Delete the old certificate.

Certificates for newly activated domains can take between 5 minutes to an hour to fully deploy across Fastly's global network. If the new certificate is not being used to serve TLS traffic within 1 hour, contact support for assistance.

Deactivating TLS and deleting certificates and private keys

Once a domain has TLS activated, you have the option to deactivate TLS via the **Deactivate TLS** button listed on the TLS domains page. If a domain has multiple certificates, you can elect to deactivate a specific certificate by clicking **Add/Edit Activations** and clicking the **Deactivate** button next to any active configurations. If all certificates are deactivated, Fastly will no longer serve TLS traffic on the selected domain and it will become disabled.

To delete a certificate from the Self-managed certificates page, be sure to disable TLS for all domains on that specific certificate. You will also need to delete all certificates before you can delete a matching private key. Private keys can only be deleted if they have no matching TLS certificate.

> ✓ TIP
>
> Unmatched and unused private keys aren't automatically deleted. To avoid reaching the maximum key limit, be sure to delete any unused keys.

Certificate expirations

Thirty days before any certificate is due to expire, the web interface will display warnings on certificates soon to expire. Fastly will also begin to periodically send automated expiration notification emails to all superusers. If the certificate is not replaced or removed, Fastly will continue to email users on the account until the certificate expires. Once expired, Fastly will no longer send automatic notifications.

* * *

TLS origin configuration messages

/en/guides/tls-origin-configuration-messages

When you are connecting to origins over TLS, you may have errors.

Hostname mismatches

- `Error: Hostname mismatch`

Why the error appears

Your origin server is serving a TLS certificate with a Common Name (CN) or list of Subject Alternate Names (SAN) that does not match the origin host or the origin's SSL hostname setting.

How to fix it

You can fix this by telling Fastly what to match against in the CN or SAN field in your origin's certificate.

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. Click the pencil to edit the affected host.
6. In the **Certificate Hostname** field, enter the hostname associated with your TLS certificate. This value is matched against the certificate common name (CN) or a subject alternate name (SAN) depending on the certificate you were issued. For example, if your certificate's CN field is `www.example.com`, enter that value for your hostname.
7. Click **Update**.
8. Click **Activate** to deploy your configuration changes.

When using custom VCL, you can specify the hostname to match against the certificate by using the `.ssl_cert_hostname` field of your origin's definition. For example: `.ssl_cert_hostname = www.example.com;`.

Certificate chain mismatches

- `Error: unable to verify the first certificate`
- `Error: self signed certificate`
- `Error: unable to get local issuer certificate`
- `Error: self signed certificate in certificate chain`
- `Error: unable to get issuer certificate`

Why the errors appear

Your origin server is serving a certificate chain that can not be validated using any of the certification authorities (CAs) that Fastly knows. This can happen for two reasons:

- Your certificate is self-signed or self-issued and you did not provide your generated CA certificate to Fastly for us to use for verification.
- Your certificate is issued by a CA that isn't in Fastly's CA certificates bundle.

How to fix them

In both cases, you can fix your configuration by adding the CA certificate that Fastly should use to verify the certificate to your service configuration:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. Click the pencil to edit the affected host.
6. In the **TLS CA certificate** field, copy and paste a PEM-formatted CA certificate.
7. Click **Update**.
8. Click **Activate** to deploy your configuration changes.

If you are using custom VCL, you can specify the CA for Fastly to use by setting the `.ssl_ca_cert` backend parameter to a PEM encoded CA certificate.

Alternatively, you can get a new certificate issued by a CA in Fastly's CA certificate bundle (e.g., Globalsign).

Connection failures

- `Error: Gethostbyname`
- `Error: Connection timeout`
- `Error: Connection refused`

Why each error appears and how to fix it

For `Gethostbyname` failures, the configured backend Host domain is returning NXDOMAIN. Double check that the DNS settings for your backend are correct.

For `Connection time out` failures, the connection to your server is timing out. Double check that your backend is accessible and responding in a timely fashion.

For `Connection refused` failures, the connection to your server is being refused, potentially by a firewall or network ACL. Double check that you have allowlisted the Fastly IP addresses and that your backend is accessible from our network.

Certificate expirations

`Error: Certificate has expired`

Why the error appears

The certificate your backend server is presenting Fastly has expired and needs to be reissued with an updated validity period.

How to fix it

If this is a self-signed certificate you can perform this update on your own by issuing a new CSR with your private key, creating the corresponding certificate, and installing it on the server.

If this is a CA signed certificate you will need to issue a new CSR with your private key, submit it to your CA, and install the signed certificate they provide you.

SSL and old TLS protocol errors

- `Error: Unknown protocol`
- `Error: SSL handshake failure`
- `Error: TLSv1 alert internal error`

Why the errors appear

Either your origin server is not configured to use TLS or it only supports older, outdated versions of the protocol. We do not support SSLv2 or SSLv3.

How to fix them

If the origin server is configured to use TLS, use the following information to troubleshoot the problem:

- Make sure your server software is up to date and running a recent version of the TLS libraries for your platform or operating system. You may have to explicitly enable a newer protocol version. Fastly supports TLS 1.3, 1.2, 1.1, and 1.0.
- Confirm that you can connect to your origin. For example, if you're using TLS 1.3, enter a command like:

  ```
  $ echo Q | openssl s_client -connect ${IP}:443 -tls1_3
  ```

  To test other versions of TLS, you can replace `-tls1_3` with `tls1_2`, `-tls1_1`, or `-tls1_0`. If the TLS handshake is successful, you should see output showing the certificate, the subject, the issuer, and additional diagnostic information.

- Use `sslscan` to list the TLS protocols and ciphers supported by the TLS server.

If the origin server is not configured to use TLS, change your service configuration to disable TLS and communicate with it on port 80 instead of port 443:

1. Log in to the Fastly web interface.
2. From the **Home** page, select the appropriate service. You can use the search box to search by ID, name, or domain. You can also go to **CDN** > **CDN Services** or **Compute** > **Compute Services** to access a list of services by type.
3. Click **Edit configuration** and then select the option to clone the active version.
4. Click **Origins**.
5. Click the pencil to edit the affected host.
6. From the **Connect to backend using TLS** menu, select **No**.
7. Click **Update**.
8. Click **Activate** to deploy your configuration changes.

RC4 cipher error

- `Error: Using RC4 Cipher`

Why the error appears

When Fastly connects to your origin server using TLS, the only cipher suite your server supports for establishing a connection is the RC4 cipher. This cipher is considered to be unsafe for general use and should be deprecated.

How to fix it

You can fix this on your origin by using the latest version of both the server and the TLS library (e.g., OpenSSL) and ensuring the cipher suites offered are tuned to best practices. You may need to explicitly blocklist the RC4 cipher.

* * *

Fastly TLS prerequisites and limitations

🔗 /en/guides/tls-prerequisites-and-limitations

Before getting started with Fastly-managed TLS subscriptions (managed TLS) or self-managed TLS certificates (Bring Your Own Certificates), be sure to review the following prerequisites and limitations.

Prerequisites for using Fastly TLS

In order to use Fastly TLS, you must have the following in place:

- a Fastly user account assigned the role of superuser or assigned a user role with added TLS management permission
- permission to modify the DNS records on the relevant domains that appear as SAN entries on the TLS certificate
- the relevant domains added to a properly configured Fastly service

Fastly-managed certificate limitations

Fastly-managed certificates are an option for both paid accounts and trial accounts. When you set up TLS using Fastly-managed certificates, Fastly uses the ACME protocol to procure and renew TLS certificates. You have several options for certification authorities:

> NOTE
>
> Your charges may vary based on the certification authority you select.

- Certainly, Fastly's publicly-trusted certification authority
- Let’s Encrypt, a third-party non-profit certification authority
- GlobalSign, a third-party commercial certification authority (only available for paid accounts)

No matter which certification authority you select, the following limitations apply:

- Fastly managed certificates require clients to support TLS v1.2 and Server Name Indication (SNI) by default. To discuss how you can use settings other than these defaults, contact sales@fastly.com. The ability to use custom settings may require you to use a dedicated Fastly IP address pool, which must be purchased separately.
- Fastly TLS does not support the Triple DES (3des) cipher suite.

Trial accounts are subject to the following limitations:

- Trial accounts include up to two TLS domains for free using the Certainly or Let's Encrypt certification authorities.
- Wildcard certificates are not supported on trial accounts.

Self-managed certificate prerequisites and limitations

Self-managed certificates are an option for paid accounts. When you set up TLS using self-managed TLS certificates, you upload and deploy your own TLS certificates and private keys using the Fastly web interface or API.

To use Fastly TLS with self-managed certificates, be sure you have the following prerequisites in place:

- a paid Fastly user account (not a developer’s trial)
- a valid X.509 TLS certificate from a trusted certification authority (CA) and a matching 256-bit ECDSA private key (recommended) or 2048-bit RSA private key
- the relevant domains added as Subject Alternative Name (SAN) entries on the TLS certificate

In addition to these prerequisites, be sure you understand the following limitations about the certificate you upload and the CA you choose:

- Uploaded certificates require clients to support TLS v1.2 and Server Name Indication (SNI) by default. To discuss how you can use settings other than these defaults, contact sales@fastly.com. The ability to use custom settings may require you to use a dedicated Fastly IP address pool, which must be purchased separately.

- If you're a DigiCert customer, be aware that upon making certificate changes, DigiCert will revoke your original certificate 72 hours after re-issuance. Be sure you upload your new certificate and switch all hostnames as soon as possible.

Each certificate you upload must have a matching private key. Private keys use cipher suites to encrypt communications through a set of algorithms and protocols, making them secure. Be sure you understand the following limitations about the private keys and cipher suites you use for Fastly TLS:

- Fastly TLS does not support the Triple DES (3des) cipher suite.

Supported cipher suites

Each certificate you upload must have a matching private key. Private keys use ciphers to encrypt communications through a set of algorithms and protocols, making them secure. Fastly supports the following cipher suites.

TLS 1.3

The following ciphers are supported on TLS 1.3, the default version of TLS used when setting up TLS for the first time.

For the highest level of security, we recommend using these ciphers.

RFC cipher name	`openssl` cipher name
`TLS_AES_256_GCM_SHA384`	`TLS13-AES-256-GCM-SHA384`
`TLS_CHACHA20_POLY1305_SHA256`	`TLS13-CHACHA20-POLY1305-SHA256`
`TLS_AES_128_GCM_SHA256`	`TLS13-AES-128-GCM-SHA256`

TLS 1.2

The following ciphers are supported on TLS 1.2, the minimum standard version of TLS supported by Fastly.

RFC cipher name	`openssl` cipher name
`TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`	`ECDHE-RSA-AES128-GCM-SHA256`
`TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`	`ECDHE-ECDSA-AES128-GCM-SHA256`
`TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384`	`ECDHE-RSA-AES256-GCM-SHA384`
`TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384`	`ECDHE-ECDSA-AES256-GCM-SHA384`
`TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256`	`ECDHE-RSA-CHACHA20-POLY1305`
`TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256`	`ECDHE-ECDSA-CHACHA20-POLY1305`
`TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256`	`ECDHE-RSA-AES128-SHA256`
`TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256`	`ECDHE-ECDSA-AES128-SHA256`

Legacy ciphers

The following are legacy ciphers supported only on TLS versions 1.0 - 1.1 and require a dedicated IP address to create custom cipher suites. These ciphers should only be used in edge cases, such as for compatibility with older devices. Where possible, we recommend upgrading to TLS 1.3 for the highest level of security.

RFC cipher name	`openssl` cipher name
`TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA`	`ECDHE-RSA-AES128-SHA`
`TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA`	`ECDHE-ECDSA-AES128-SHA`
`TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA`	`ECDHE-RSA-AES256-SHA`
`TLS_RSA_WITH_AES_128_GCM_SHA256`	`AES128-GCM-SHA256`
`TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA`	`ECDHE-ECDSA-AES256-SHA`
`TLS_RSA_WITH_AES_128_CBC_SHA`	`AES128-SHA`
`TLS_RSA_WITH_AES_256_CBC_SHA`	`AES256-SHA`
`TLS_RSA_WITH_3DES_EDE_CBC_SHA`	`DES-CBC3-SHA`

Legacy Customer-Provided TLS Certificate Hosting Service limitations

NOTE

Fastly maintains support for its original Customer-Provided TLS Certificate Hosting Service. For information on migrating certificates from the Customer-Provided TLS Certificate Hosting Service to our current Fastly TLS offerings, contact Support.

The Fastly TLS web interface is compatible with certificates that have been uploaded as part of the Customer-Provided TLS Certificate Hosting Service with the following limitations:

- If you update previously uploaded certificates, you can continue to use the Customer-Provided TLS Certificate Hosting Service with no changes to your bill.
- Removing a previously uploaded certificate from the Customer-Provided TLS Certificate Hosting Service and uploading a new one using Fastly TLS will result in the new certificate being counted in your bill for Fastly TLS. The old certificate will continue to be billed per any contracted term for Customer-Provided TLS Certificate Hosting Service.

Next steps

Once you've reviewed these prerequisites and limitations, you are ready to get started with TLS. Refer to our TLS quick start guide to quickly get set up using Certainly. You can also check out our guides on Setting up TLS with certificates Fastly manages and Setting up TLS with your own certificates for more in-depth instructions.

---

* * *

TLS quick start

/en/guides/tls-quick-start

Welcome! This guide helps new TLS users like you get started with Fastly TLS as quickly as possible using Certainly, Fastly's publicly-trusted certification authority. By setting up TLS for a domain you've added to a Fastly service, you can deliver your origin website to your users through the Fastly CDN.

Before you begin

Keep in mind the following:

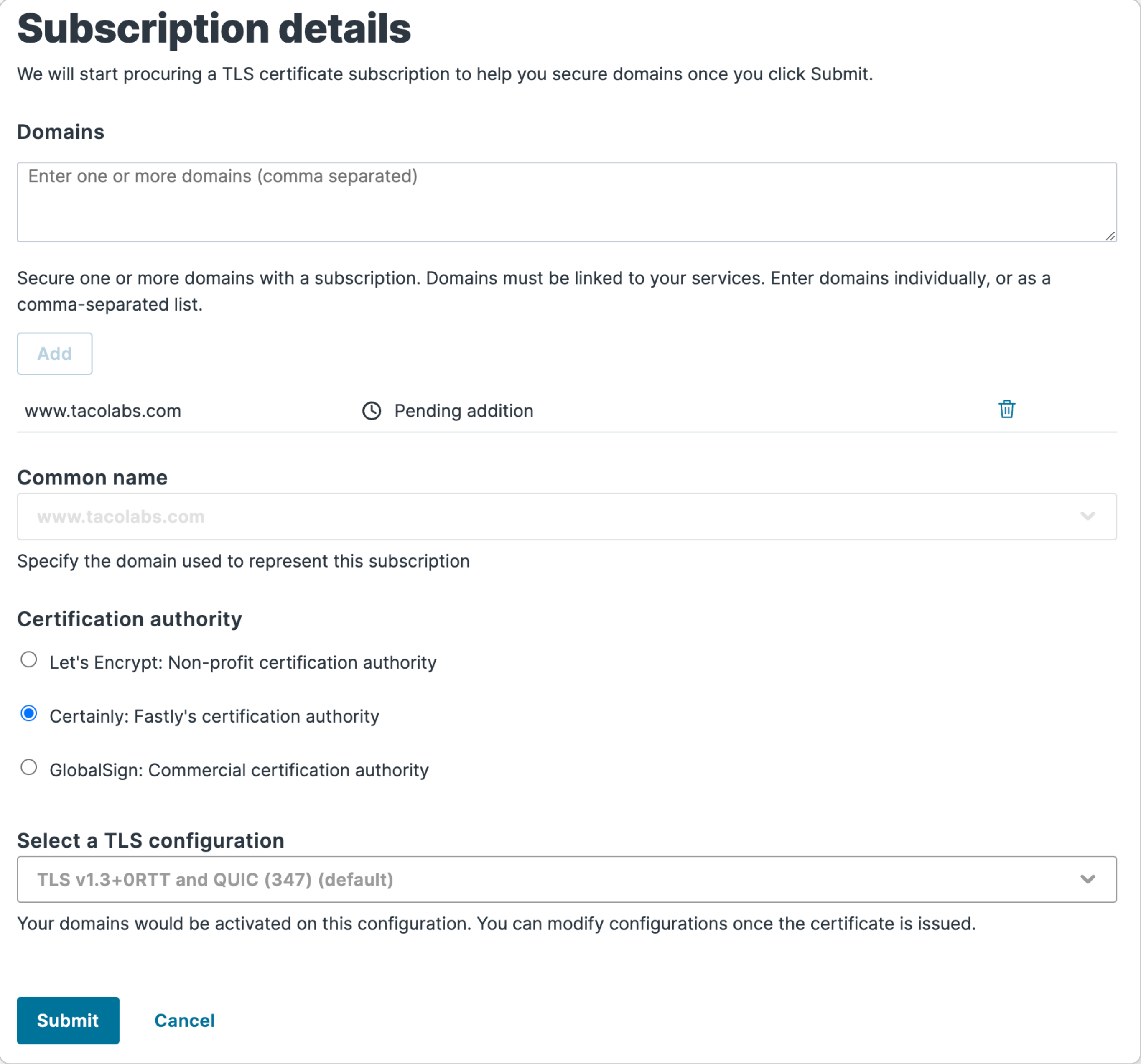
- Be sure to review the TLS prerequisites and limitations.
- This guide assumes you have a Fastly account and a service with at least one domain added. If you don't, refer to our Start here guide to quickly get set up.
- This guide includes instructions for working with subdomains, not apex or second-level domains. To work with those types of domains, refer to our guide on using Fastly with apex domains.

1. Create a TLS certificate

We’ll use Fastly-managed TLS to create a TLS certificate, which means Fastly will procure the certificate for us and manage the certificate renewals. We'll have Fastly issue a certificate using Certainly.

> ✓ TIP
>
> Trial accounts include up to two TLS domains for free with Certainly, while paid accounts include five free domains using Certainly. If you want to bring your own certificates or use Fastly-managed TLS with another certification authority, refer to those specific guides.

1. Log in to the Fastly web interface and select **Security** from the navigation sidebar. The Overview page appears displaying an overview of Fastly's security offerings.
2. Click **Manage certificates**.
3. If this is your first time setting up TLS, click **Get started**. If you have a TLS certificate in your account already, click **Secure another domain** instead.
4. Enter the name of the domain you want to secure with HTTPS (e.g., `www.tacolabs.com`), and then click **Add**. Be sure this domain is present on one of your Fastly services.
5. Fastly adds your domain as the **Common name**. Select Certainly as the certificate authority and keep the default TLS configuration, then click **Submit**.



Subscription details

We will start procuring a TLS certificate subscription to help you secure domains once you click Submit.

Domains

Enter one or more domains (comma separated)

Secure one or more domains with a subscription. Domains must be linked to your services. Enter domains individually, or as a comma-separated list.

www.tacolabs.com Pending addition

Common name

www.tacolabs.com

Specify the domain used to represent this subscription

Certification authority

○ Let's Encrypt: Non-profit certification authority

● Certainly: Fastly's certification authority

○ GlobalSign: Commercial certification authority

Select a TLS configuration

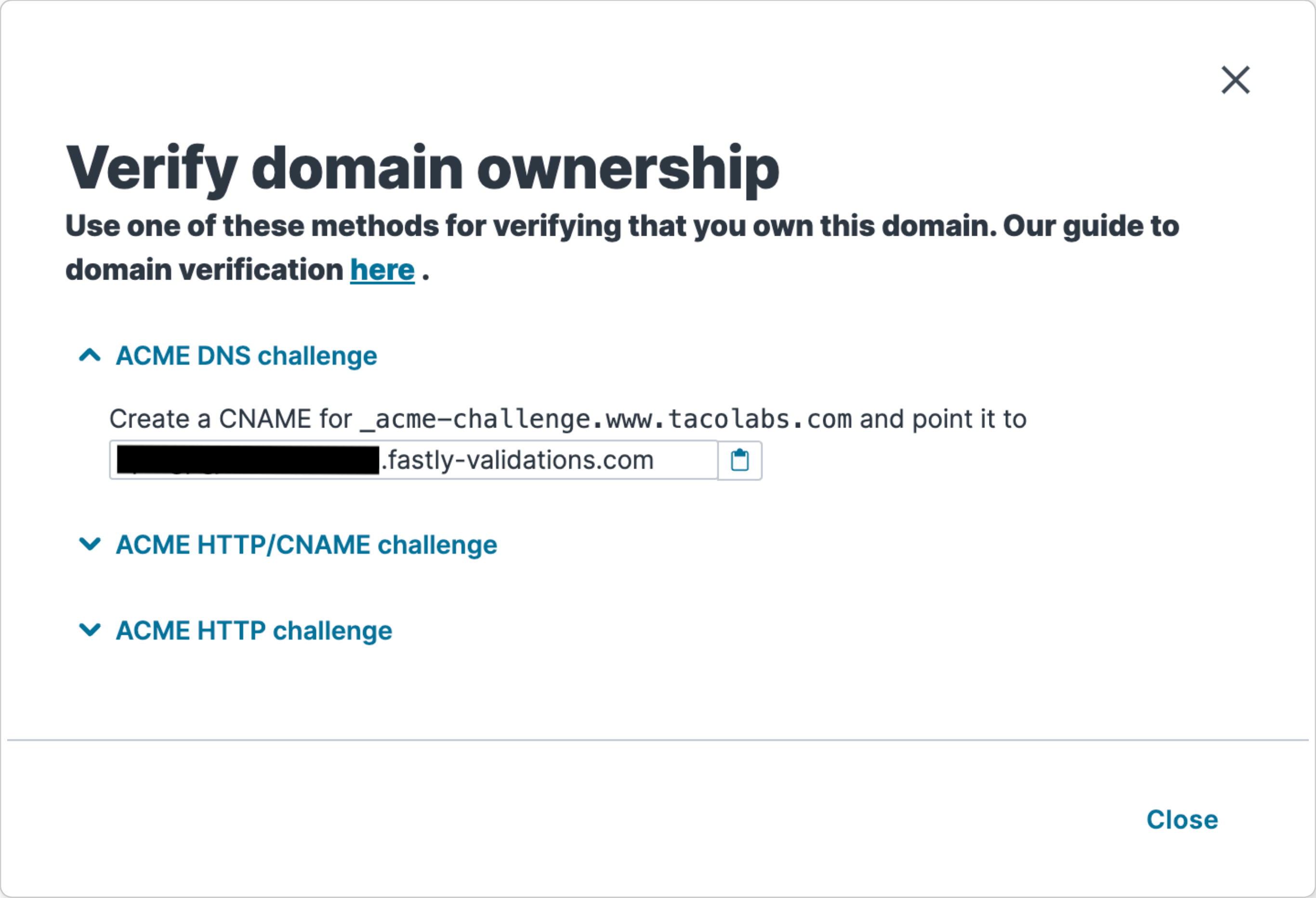
TLS v1.3+0RTT and QUIC (347) (default)

Your domains would be activated on this configuration. You can modify configurations once the certificate is issued.

2. Verify your domain

Once you click Submit, Fastly provides the necessary information to use to confirm you own the domain. You'll need access to the DNS records for your domain - in most cases this will be wherever you purchased the domain. While the exact process may vary, the instructions below should work for most providers.

1. Click the **Verification options** for the domain and expand the menu for **ACME DNS challenge**.
2. Note the CNAME address (`_acme-challenge` followed by your domain name), which is the address to point your DNS record to. The CNAME value is a string of characters followed by `fastly-validations.com`. You'll need these later.



Verify domain ownership

Use one of these methods for verifying that you own this domain. Our guide to domain verification here .

Create a CNAME for `_acme-challenge.www.tacolabs.com` and point it to

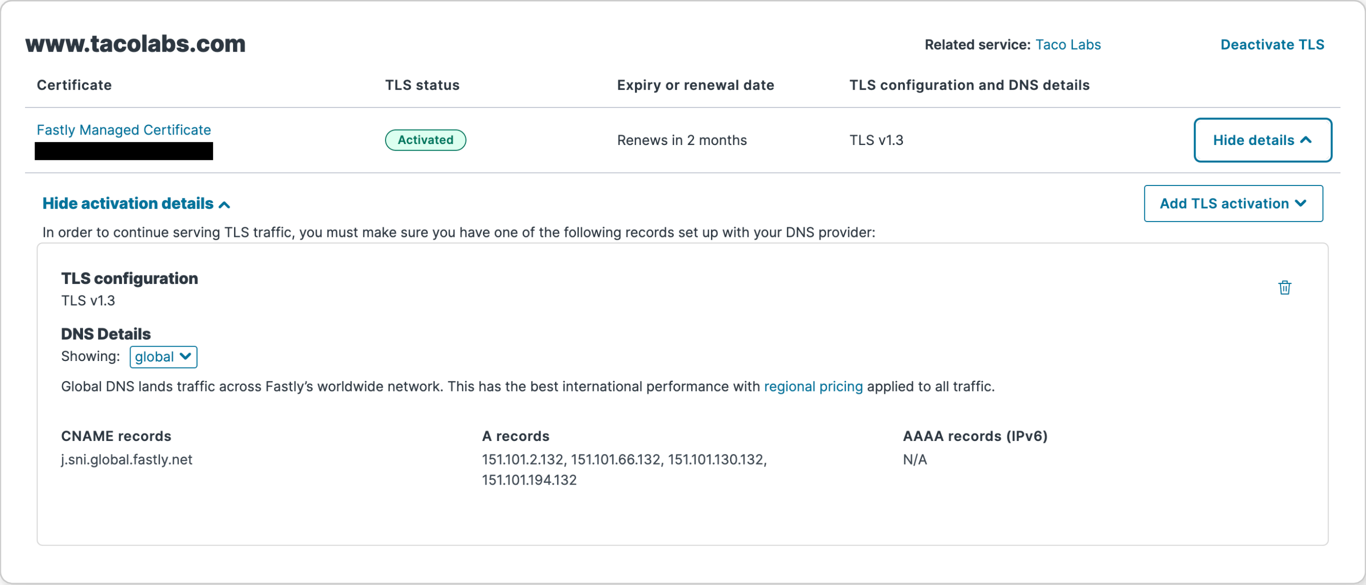
[redacted].fastly-validations.com

3. Access your DNS provider or domain registrar (wherever you registered or purchased the domain) and navigate to the Custom DNS settings.
4. Add a new CNAME record to verify the domain for the TLS certificate.
    - Set the **hostname** to the CNAME address from the TLS info that's something like `_acme-challenge` followed by your domain.
    - Set the **address**, sometimes called data or content, to the CNAME value which ends in `fastly-validations.com`.
5. Save your new record then navigate back to your Fastly account. Fastly will work to verify your domain and issue the certificate, a process that typically takes 20 minutes to an hour, but can extend up to 72 hours.

3. Point your domain at Fastly

Now that the certificate has been issued, you'll add another CNAME record, this time to update your DNS records to point your domain at Fastly and begin serving traffic.

1. Locate the value for the CNAME record for your domain in Fastly by clicking **See DNS details** on the **TLS domains page**. The CNAME record value is a single letter followed by `.sni.global.fastly.net`.



www.tacolabs.com

Related service: Taco Labs

Certificate	TLS status	Expiry or renewal date	TLS configuration and DNS details
Fastly Managed Certificate [redacted]	Activated	Renews in 2 months	TLS v1.3

In order to continue serving TLS traffic, you must make sure you have one of the following records set up with your DNS provider:

TLS configuration
TLS v1.3

DNS Details
Showing: global

Global DNS lands traffic across Fastly's worldwide network. This has the best international performance with regional pricing applied to all traffic.

CNAME records	A records	AAAA records (IPv6)
j.sni.global.fastly.net	151.101.2.132, 151.101.66.132, 151.101.130.132, 151.101.194.132	N/A

In this example, the Fastly web interface says that the value for our CNAME record is `j.sni.global.fastly.net`. This value doesn't support IPv6 addresses, but you can preface `dualstack` to enable IPv6 support (i.e., `dualstack.j.sni.global.fastly.net`).

2. Navigate back to your DNS records and add another CNAME record, this time for the domain you want to point at Fastly. If you have an existing DNS record for this domain, you'll need to edit or remove it.
   - Set the **hostname** to the domain you want to point at Fastly, for example, `www.tacolabs.com`.

     > TIP
     >
     > If using an apex domain, refer to Using Fastly with apex domains for instruction.

   - Set the **address** to `dualstack.<letter>.sni.global.fastly.net` (adjusting the single letter to match your certificate).

Give it some time for your changes to propagate. While most newly issued certificates take between 20 minutes to an hour to fully deploy across Fastly's global network, it may take up to 72 hours depending on your DNS provider and Time To Live (TTL) value, which is in seconds. Once the changes propagate, you'll see the origin site you pointed your Fastly service at when you visit the domain you registered! You can also use the following command to check the status:

```
$ dig www.tacolabs.com +short
dualstack.t.sni.global.fastly.net.
151.101.198.132
```

4. Certificate renewals

Fastly automatically handles TLS certificate renewals prior to expiration, assuming your CNAME records haven't been modified. Ten days before your certificates are due to expire, Fastly automatically runs a DNS check for certificate renewals. However, if your DNS records no longer point at Fastly or if a CAA record blocks Certainly, the certificate will lapse at the end of the 30-day period. For more information, refer to Certificate management and renewals.

* * *

TLS termination

/en/guides/tls-termination

Identifying TLS terminated requests

To maintain optimal caching performance, Fastly uses a TLS terminator separate from the caching engine. This means, however, that the caching engine doesn't know that it was originally a TLS request. As a result, we set the `Fastly-SSL` header when fetching the content from your servers.

Because we set this header, you can check for its presence on your backend by including something like this in your `vcl_recv` subroutine:

```
if (req.http.Fastly-SSL) {
    set resp.http.X-Is-SSL = "yes";
  }
```

and that should tell you if the request was a TLS request or not.

When using WordPress

If you're using Fastly TLS services with WordPress, you'll want to add a check for the `HTTP_FASTLY_SSL` header so that WordPress can build URLs to your CSS or JS assets correctly. Do this by placing a check in your `wp-config.php` file to override the SSL flag that is checked later:

```
if (!empty( $_SERVER['HTTP_FASTLY_SSL'])) {
    $_SERVER['HTTPS'] = 'on';
  }
```

As usual, this must be placed anywhere before the `require_once` line with `wp-settings.php`.

Finding the original IP when using TLS termination

Because Fastly uses a TLS terminator, separate from the caching engine for performance, the engine overwrites the original IP address briefly due to the re-request to your origin servers once decrypted and causes anything that references the original IP to show up as 127.0.0.0/8 IPs. To find the original IP via VCL:

- use `req.http.Fastly-Client-IP` if you're using shielding
- use `client.ip` if you're not using shielding or if you're building an ACL

Fastly also sends along the client IP to the origin in a HTTP header, `Fastly-Client-IP`, which can be used by server software to adjust as needed.

* * *

© Fastly 2024