Streaming the audit log for your enterprise

In this article

You can stream audit and Git events data from GitHub to an external data management system.

> **Who can use this feature**
> Enterprise owners can configure audit log streaming.

Note: Audit log streaming is currently in beta for GitHub Enterprise Server and is subject to change.

About audit log streaming

To help protect your intellectual property and maintain compliance for your organization, you can use streaming to keep copies of your audit log data and monitor:

- Access to your organization or repository settings
- Changes in permissions
- Added or removed users in an organization, repository, or team
- Users being promoted to admin
- Changes to permissions of a GitHub App
- Git events, such as cloning, fetching, and pushing (must be enabled, see "Configuring the audit log for your enterprise")

The benefits of streaming audit data include:

- **Data exploration**. You can examine streamed events using your preferred tool for querying large quantities of data. The stream contains both audit events and Git events across the entire enterprise account.
- **Data retention**. You can keep your exported audit logs and Git events data as long as you need to.

Enterprise owners can set up or delete a stream at any time. The stream exports audit and Git events data for all of the organizations in your enterprise, for activity from the time the stream is enabled onwards.

All streamed audit logs are sent as compressed JSON files. The filename format is in `YYYY/MM/HH/MM/<uuid>.json.gz`.

Note: GitHub uses an at-least-once delivery method. Due to certain network or system issues, some events may be duplicated.

Enabling audit log streaming can cause a minor impact on the performance of your GitHub Enterprise Server instance. For more information about increasing resources to mitigate this performance impact, see "Increasing CPU or memory resources."

Setting up audit log streaming

You set up the audit log stream on GitHub Enterprise Server by following the instructions for your provider.

- Amazon S3
- Azure Blob Storage
- Azure Event Hubs
- Datadog
- Google Cloud Storage
- Splunk

Setting up streaming to Amazon S3

To set up audit log streaming from GitHub you will need:

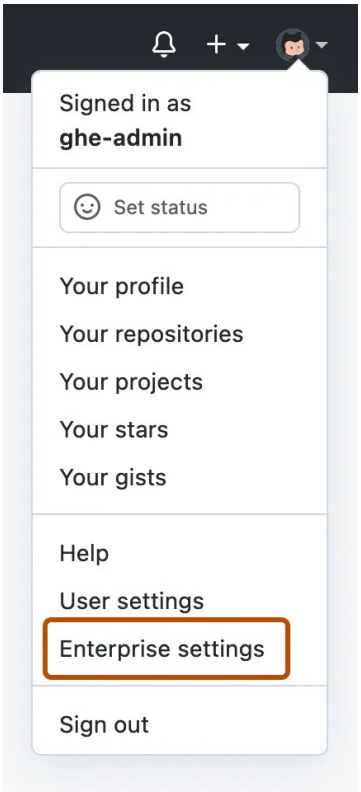
- Your AWS access key ID
- Your AWS secret key

For information on creating or accessing your access key ID and secret key, see Understanding and getting your AWS credentials in the AWS documentation.

1. In AWS, create a bucket, and block public access to the bucket. For more information, see Creating, configuring, and working with Amazon S3 buckets in the AWS documentation.
2. In AWS, create a policy that allows GitHub to write to the bucket by copying the following JSON and replacing `EXAMPLE-BUCKET` with the name of your bucket. GitHub requires only the permissions in this JSON.

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
         {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
               "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::EXAMPLE-BUCKET/*"
        }
      ]
   }
   ```

   For more information, see Creating IAM policies in the AWS documentation.
3. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



4. In the enterprise account sidebar, click ⚙ **Settings**.
5. Under "⚙ Settings", click **Audit log**.
6. Under "Audit log", click **Log streaming**.
7. Select the **Configure stream** dropdown menu and click **Amazon S3**.
8. Configure the stream settings.
   - Under "Bucket", type the name of the bucket you want to stream to. For example, `auditlog-streaming-test`.
   - Under "Access Key ID", type your access key ID. For example, `ABCAIOSFODNN7EXAMPLE1`.
   - Under "Secret Key", type your secret key. For example, `aBcJalrXUtnWXYZ/A1MDENG/zPxRfiCYEXAMPLEKEY`.
9. To verify that GitHub can connect and write to the Amazon S3 endpoint, click **Check endpoint**.
10. After you have successfully verified the endpoint, click **Save**.

Integrating with AWS CloudTrail Lake

You can consolidate your audit logs from GitHub Enterprise Server with AWS activity logs by integrating audit log streaming to S3 with AWS CloudTrail Lake. For additional information, see the AWS CloudTrail Documentation or the GitHub Audit Log to CloudTrail Open Audit in the `aws-samples/aws-cloudtrail-lake-github-audit-log` repository.

Setting up streaming to Azure Blob Storage

Before setting up a stream in GitHub, you must first have created a storage account and a container in Microsoft Azure. For details, see the Microsoft documentation, "Introduction to Azure Blob Storage."

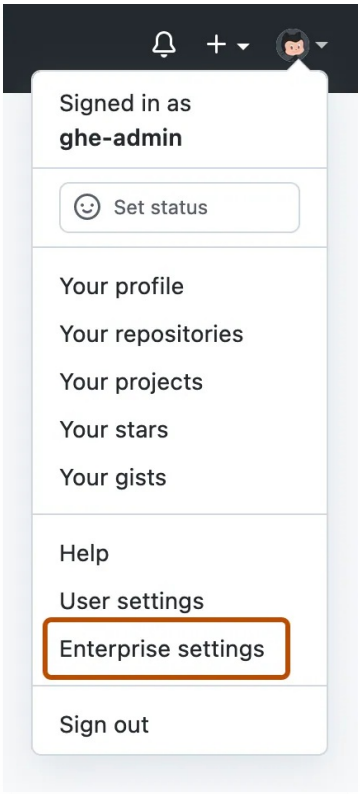
To configure the stream in GitHub you need the URL of a SAS token.

On Microsoft Azure portal:

1. On the Home page, click **Storage Accounts**.
2. Under "Name", click the name of the storage account you want to use.
3. Under "Data storage", click **Containers**.
4. Click the name of the container you want to use.
5. In the left sidebar, under "Settings", click **Shared access tokens**.
6. Select the **Permissions** dropdown menu, then select `Create` and `Write` and deselect all other options.
7. Set an expiry date that complies with your secret rotation policy.
8. Click **Generate SAS token and URL**.
9. Copy the value of the **Blob SAS URL** field that's displayed. You will use this URL in GitHub.

On GitHub:

1. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



2. In the enterprise account sidebar, click ⚙ **Settings**.
3. Under "⚙ Settings", click **Audit log**.
4. Under "Audit log", click **Log streaming**.
5. Select the **Configure stream** dropdown menu and click **Azure Blob Storage**.

6. On the configuration page, enter the blob SAS URL that you copied in Azure. The **Container** field is auto-filled based on the URL.
7. Click **Check endpoint** to verify that GitHub can connect and write to the Azure Blob Storage endpoint.
8. After you have successfully verified the endpoint, click **Save**.

Setting up streaming to Azure Event Hubs

Before setting up a stream in GitHub, you must first have an event hub namespace in Microsoft Azure. Next, you must create an event hub instance within the namespace. You'll need the details of this event hub instance when you set up the stream. For details, see the Microsoft documentation, "Quickstart: Create an event hub using Azure portal."

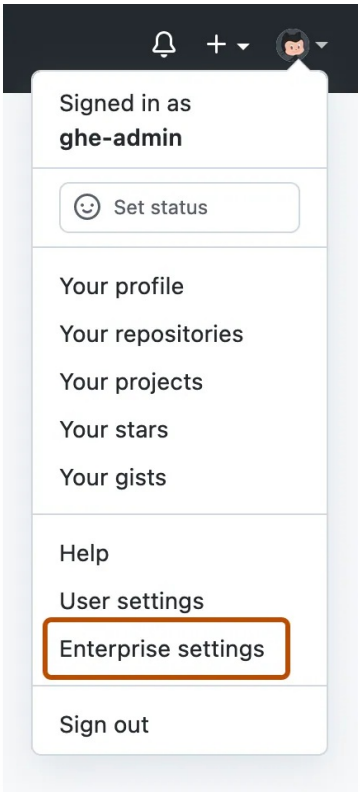
You need two pieces of information about your event hub: its instance name and the connection string.

On Microsoft Azure portal:

1. At the top of the page, next to "Microsoft Azure", use the search box to search for "Event Hubs".
2. Select **Event Hubs**. The names of your event hubs are listed.
3. Make a note of the name of the event hub to which you want to stream. Click the event hub.
4. In the left menu, click **Shared Access Policies**.
5. Select a shared access policy from the list of policies, or create a new policy.
6. Copy the connection string from the **Connection string-primary key** field.

On GitHub:

1. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



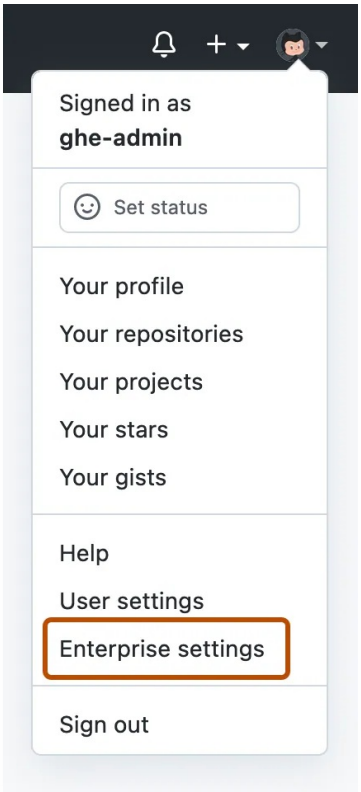
2. In the enterprise account sidebar, click ⚙ **Settings**.
3. Under "⚙ Settings", click **Audit log**.
4. Under "Audit log", click **Log streaming**.
5. Select the **Configure stream** dropdown menu and click **Azure Event Hubs**.
6. On the configuration page, enter:
   - The name of the Azure Event Hubs instance.
   - The connection string.
7. Click **Check endpoint** to verify that GitHub can connect and write to the Azure Events Hub endpoint.
8. After you have successfully verified the endpoint, click **Save**.

Setting up streaming to Datadog

To set up streaming to Datadog, you must create a client token or an API key in Datadog, then configure audit log streaming in GitHub Enterprise Server using the token for authentication. You do not need to create a bucket or other storage container in Datadog.

After you set up streaming to Datadog, you can see your audit log data by filtering by "github.audit.streaming." For more information, see Log Management.

1. If you don't already have a Datadog account, create one.
2. In Datadog, generate a client token or an API key and then click **Copy key**. For more information, see API and Application Keys in Datadog Docs.
3. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



4. In the enterprise account sidebar, click ⚙ **Settings**.
5. Under "⚙ Settings", click **Audit log**.
6. Under "Audit log", click **Log streaming**.
7. Select the **Configure stream** dropdown menu and click **Datadog**.
8. In the **Token** field, paste the token you copied earlier.
9. Select the **Site** dropdown menu and click your Datadog site. To determine your Datadog site, compare your Datadog URL to the table in Datadog sites in Datadog Docs.
10. To verify that GitHub can connect and write to the Datadog endpoint, click **Check endpoint**.
11. After you have successfully verified the endpoint, click **Save**.
12. After a few minutes, confirm that audit log data is appearing on the **Logs** tab in Datadog. If audit log data is not appearing, confirm that your token and site are correct in GitHub.

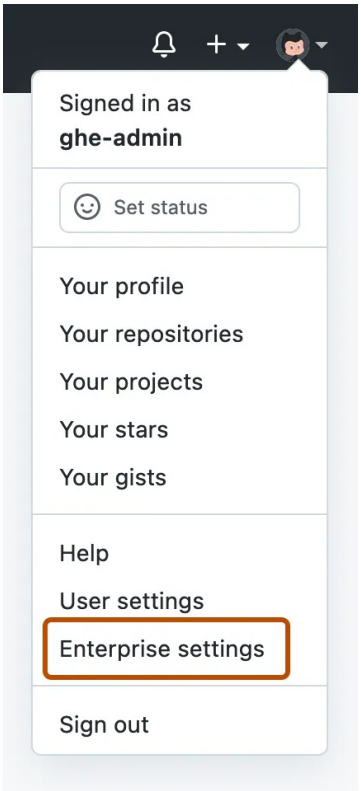
Setting up streaming to Google Cloud Storage

To set up streaming to Google Cloud Storage, you must create a service account in Google Cloud with the appropriate credentials and permissions, then configure audit log streaming in GitHub Enterprise Server using the service account's credentials for authentication.

1. Create a service account for Google Cloud. You do not need to set access controls or IAM roles for the service account. For more information, see Creating and managing service accounts in the Google Cloud documentation.
2. Create a JSON key for the service account, and store the key securely. For more

information, see Creating and managing service account keys in the Google Cloud documentation.

3. If you haven't created a bucket yet, create the bucket. For more information, see Creating storage buckets in the Google Cloud documentation.
4. Give the service account the Storage Object Creator role for the bucket. For more information, see Using Cloud IAM permissions in the Google Cloud documentation.
5. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.

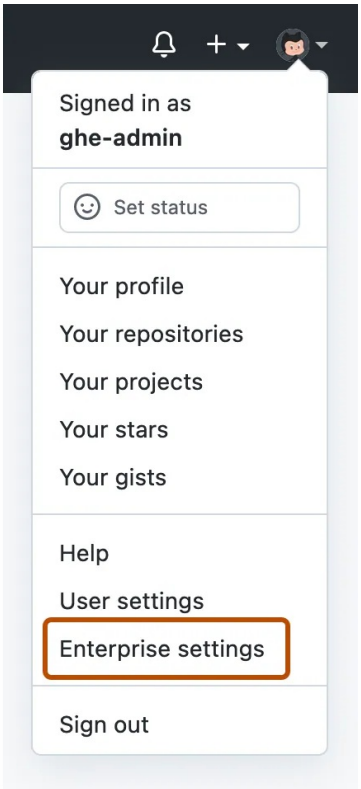


6. In the enterprise account sidebar, click ⚙ **Settings**.
7. Under "⚙ Settings", click **Audit log**.
8. Under "Audit log", click **Log streaming**.
9. Select the **Configure stream** dropdown menu and click **Google Cloud Storage**.
10. Under "Bucket", type the name of your Google Cloud Storage bucket.
11. Under "JSON Credentials", paste the entire contents of the file for your service account's JSON key.
12. To verify that GitHub can connect and write to the Google Cloud Storage bucket, click **Check endpoint**.
13. After you have successfully verified the endpoint, click **Save**.

Setting up streaming to Splunk

To stream audit logs to Splunk's HTTP Event Collector (HEC) endpoint you must make sure that the endpoint is configured to accept HTTPS connections. For more information, see Set up and use HTTP Event Collector in Splunk Web in the Splunk documentation.

1. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



2. In the enterprise account sidebar, click ⚙ **Settings**.
3. Under "⚙ Settings", click **Audit log**.
4. Under "Audit log", click **Log streaming**.
5. Select the **Configure stream** dropdown menu and click **Splunk**.
6. On the configuration page, enter:
   - The domain on which the application you want to stream to is hosted.

     If you're using Splunk Cloud, `Domain` should be `http-inputs-<host>`, where `host` is the domain you use in Splunk Cloud. For example, `http-inputs-mycompany.splunkcloud.com`.

     If you're using the free trial version of Splunk Cloud, `Domain` should be `inputs.<host>`, where `host` is the domain you use in Splunk Cloud. For example, `inputs.mycompany.splunkcloud.com`.
   - The port on which the application accepts data.

     If you're using Splunk Cloud and haven't changed the port configration, `Port` should be `443`.

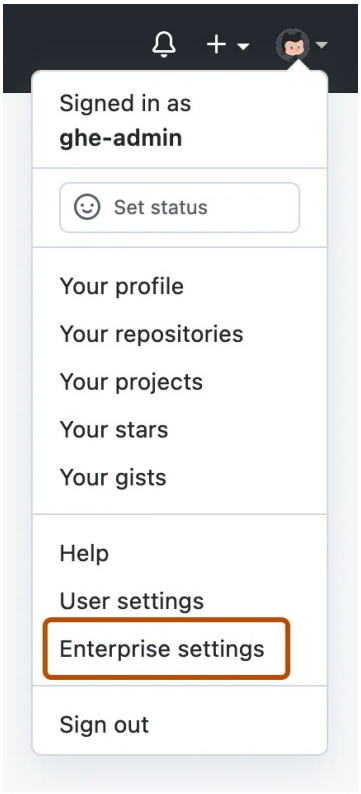
     If you're using the free trial version of Splunk Cloud, `Port` should be `8088`.
   - A token that GitHub can use to authenticate to the third-party application.
7. Leave the **Enable SSL verification** check box selected.

   Audit logs are always streamed as encrypted data, however, with this option selected, GitHub verifies the SSL certificate of your Splunk instance when delivering events. SSL verification helps ensure that events are delivered to your URL endpoint securely. You can clear the selection of this option, but we recommend you leave SSL verification enabled.

8. Click **Check endpoint** to verify that GitHub can connect and write to the Splunk endpoint.
9. After you have successfully verified the endpoint, click **Save**.

Deleting the audit log stream

1. In the top-right corner of GitHub Enterprise Server, click your profile photo, then click **Enterprise settings**.



2. In the enterprise account sidebar, click ⚙ **Settings**.
3. Under "⚙ Settings", click **Audit log**.
4. Under "Audit log", click **Log streaming**.
5. Under "Danger zone", click **Delete stream**.
6. A confirmation message is displayed. Click **Delete stream** to confirm.