

Enabling GitHub Actions with Google Cloud Storage

In this article

About external storage for GitHub Actions

Prerequisites

Enabling GitHub Actions with Google Cloud Storage using OIDC (recommended)

Enabling GitHub Actions with Google Cloud Storage using a HMAC key

Next steps

You can enable GitHub Actions on GitHub Enterprise Server and use Google Cloud Storage to store data generated by workflow runs.

Who can use this feature

Site administrators can enable GitHub Actions and configure enterprise settings.

Note: GitHub Actions support for Google Cloud Storage is currently in beta and subject to change.

About external storage for GitHub Actions

GitHub Actions uses external blob storage to store data generated by workflow runs. Stored data includes workflow logs, caches, and user-uploaded build artifacts. For more information, see "[Getting started with GitHub Actions for GitHub Enterprise Server](#)."

There are two options for configuring GitHub Enterprise Server to connect to your external storage provider:

- OpenID Connect (OIDC)
- Traditional credentials-based authentication using secrets

We recommend using OIDC where possible, as you won't need create or manage sensitive and long-lived credential secrets for your storage provider, and risk them being exposed. After defining a trust with OIDC, your cloud storage provider automatically issues short-lived access tokens to your GitHub Enterprise Server instance, which automatically expire.

Note: Using OIDC to connect to an external storage provider is in beta and subject to change.

Prerequisites

Before enabling GitHub Actions, make sure you have completed the following steps:

- Create your Google Cloud Storage bucket for storing data generated by workflow runs.

- Review the hardware requirements for GitHub Actions. For more information, see "[Getting started with GitHub Actions for GitHub Enterprise Server](#)."
- TLS must be configured for your GitHub Enterprise Server instance's domain. For more information, see "[Configuring TLS](#)."

Note: We strongly recommend that you configure TLS on GitHub Enterprise Server with a certificate signed by a trusted authority. Although a self-signed certificate can work, extra configuration is required for your self-hosted runners, and it is not recommended for production environments.

- If you have an **HTTP Proxy Server** configured on your GitHub Enterprise Server instance:
- You must add `.localhost` and `127.0.0.1` to the **HTTP Proxy Exclusion** list.
- If your external storage location is not routable, then you must also add your external storage URL to the exclusion list.

For more information on changing your proxy settings, see "[Configuring an outbound web proxy server](#)."

- If you are using OIDC for the connection to your storage provider, you must expose the following OIDC token service URLs on your GitHub Enterprise Server instance to the public internet:

```
https://HOSTNAME/_services/token/.well-known/openid-configuration
https://HOSTNAME/_services/token/.well-known/jwks
```

This ensures that the storage provider can contact your GitHub Enterprise Server instance for authentication.

Enabling GitHub Actions with Google Cloud Storage using OIDC (recommended)

Note: Using OIDC to connect to an external storage provider is in beta and subject to change.

To configure GitHub Enterprise Server to use OIDC with Google Cloud Storage, you must first create a Google Cloud service account, then create a Google Cloud identity pool and identity provider, and finally configure GitHub Enterprise Server to use the provider and service account to access your Google Cloud Storage bucket.

1. Create a service account

- 1 Create a service account that can access your bucket using OIDC. For more information, see [Creating and managing service accounts](#) in the Google Cloud documentation.

When creating the service account, ensure that you do the following:

- Enable the IAM API as described at the start of [Creating and managing service accounts](#).
- Add the following roles to the service account:
 - Service Account Token Creator
 - Storage Object Admin

- 2 After creating the service account, note its email address, as it is need later. The

service account email address is in the format `SERVICE-ACCOUNT-NAME@PROJECT-NAME.iam.gserviceaccount.com`.

2. Create an identity pool and identity provider

1 In the Google Cloud console, go to the [New workload provider and pool](#) page.

2 Under "Create an identity pool", enter a name for the identity pool, and click **Continue**.

3 Under "Add a provider to pool":

- For "Select a provider", select **OpenID Connect (OIDC)**.
- For "Provider name", enter a name for the provider.
- For "Issuer (URL)", enter the following URL, replacing `HOSTNAME` with the public hostname for your GitHub Enterprise Server instance:

```
https://HOSTNAME/_services/token
```

For example:

```
https://my-ghes-host.example.com/_services/token
```

- Under "Audiences", leave **Default audience** selected, but note the identity provider URL, as it is needed later. The identity provider URL is in the format `https://iam.googleapis.com/projects/PROJECT-NUMBER/locations/global/workloadIdentityPools/POOL-NAME/providers/PROVIDER-NAME`.
- Click **Continue**.

4 Under "Configure provider attributes":

- For the "OIDC 1" mapping, enter `assertion.sub`.
- Under "Attribute Conditions", click **Add condition**.
- For "Condition CEL", enter the following condition, replacing `HOSTNAME` with the public hostname for your GitHub Enterprise Server instance:

```
google.subject == "HOSTNAME"
```

For example:

```
google.subject == "my-ghes-host.example.com"
```

Note: The hostname of your GitHub Enterprise Server instance used here *must not* include the protocol.



- Click **Save**.

5 After creating the identity pool, at the top of the identity pool's page, click **Grant access**.

- Under "Select service account", select the service account that you created in the previous procedure.
- Under "Select principals (identities that can access the service account)", select **Only identities matching the filter**.

- For "Attribute name", select **subject**.
- For "Attribute value", enter your GitHub Enterprise Server hostname, without the protocol. For example, `my-ghes-host.example.com`.
- Click **Save**.
- You can dismiss the "Configure your application" dialog, as the configuration file is not needed.

3. Configure GitHub Enterprise Server to connect to Google Cloud Storage using OIDC

- 1 From an administrative account on GitHub Enterprise Server, in the upper-right corner of any page, click .
- 2 If you're not already on the "Site admin" page, in the upper-left corner, click **Site admin**.
- 3 In the " Site admin" sidebar, click **Management Console**.
- 4 In the "Settings" sidebar, click **Actions**.
- 5 Under "GitHub Actions", select **Enable GitHub Actions**.
- 6 Under "Artifact & Log Storage", next to "Google Cloud Storage", click **Setup**.
- 7 Under "Authentication", select **OpenID Connect (OIDC)**, and enter the values for your storage:

- **Service URL:** The service URL for your bucket. This is usually `https://storage.googleapis.com`.
- **Bucket name:** The name of your bucket.
- **Workload Identity Provider ID:** The identity provider ID for your identity pool.

This is in the format `projects/PROJECT-NUMBER/locations/global/workloadIdentityPools/POOL-NAME/providers/PROVIDER-NAME`. Note that you must remove the `https://iam.googleapis.com/` prefix from the value noted in the previous procedure.

For example, `projects/1234567890/locations/global/workloadIdentityPools/my-pool/providers/my-provider`.

- **Service account:** The service account email address that you noted in the previous procedure. For example, `ghes-oidc-service-account@my-project.iam.gserviceaccount.com`.

- 8 Click the **Test storage settings** button to validate your storage settings.

If there are any errors validating the storage settings, check the settings with your storage provider and try again.

- 9 Under the "Settings" sidebar, click **Save settings**.

Note: Saving settings in the Management Console restarts system services, which could result in user-visible downtime.



- 10 Wait for the configuration run to complete.

Enabling GitHub Actions with Google Cloud Storage using a HMAC key

- 1 Create a Google Cloud service account that can access the bucket, and create a Hash-based Message Authentication Code (HMAC) key for the service account. For more information, see "[Manage HMAC keys for service accounts](#)" in the Google Cloud documentation.

The service account must have the following [Identity and Access Management \(IAM\) permissions](#) for the bucket:

- `storage.objects.create`
- `storage.objects.get`
- `storage.objects.list`
- `storage.objects.update`
- `storage.objects.delete`
- `storage.multipartUploads.create`
- `storage.multipartUploads.abort`
- `storage.multipartUploads.listParts`
- `storage.multipartUploads.list`

- 2 From an administrative account on GitHub Enterprise Server, in the upper-right corner of any page, click .
- 3 If you're not already on the "Site admin" page, in the upper-left corner, click **Site admin**.
- 4 In the " Site admin" sidebar, click **Management Console**.
- 5 In the "Settings" sidebar, click **Actions**.
- 6 Under "GitHub Actions", select **Enable GitHub Actions**.
- 7 Under "Artifact & Log Storage", next to "Google Cloud Storage", click **Setup**.
- 8 Under "Authentication", select **Credentials-based**, and enter your storage bucket's details:
 - **Service URL**: The service URL for your bucket. This is usually `https://storage.googleapis.com`.
 - **Bucket Name**: The name of your bucket.
 - **HMAC Access Id** and **HMAC Secret**: The Google Cloud access ID and secret for your storage account. For more information, see "[Manage HMAC keys for service accounts](#)" in the Google Cloud documentation.

- 9 Click the **Test storage settings** button to validate your storage settings.

If there are any errors validating the storage settings, check the settings with your storage provider and try again.

- 10 Under the "Settings" sidebar, click **Save settings**.

Note: Saving settings in the Management Console restarts system services, which could result in user-visible downtime.

- 11 Wait for the configuration run to complete.

Next steps

After the configuration run has successfully completed, GitHub Actions will be enabled on your GitHub Enterprise Server instance. For your next steps, such as managing GitHub Actions access permissions and adding self-hosted runners, return to "[Getting started with GitHub Actions for GitHub Enterprise Server](#)."

Legal

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Status](#) [Pricing](#) [Expert services](#) [Blog](#)