Enabling GitHub Actions with Amazon S3 storage

In this article

About external storage for GitHub Actions
Prerequisites
Enabling GitHub Actions with Amazon S3 using OIDC (recommended)
Enabling GitHub Actions with Amazon S3 storage using access keys
Next steps

You can enable GitHub Actions on GitHub Enterprise Server and use Amazon S3 storage to store data generated by workflow runs.

> **Who can use this feature**
> Site administrators can enable GitHub Actions and configure enterprise settings.

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

> **Note:** The only GitHub-supported S3 storage providers are Amazon S3 and MinIO Gateway for NAS.
>
> There are other S3 API-compatible storage products that GitHub partners have self-validated as working with GitHub Actions on GitHub Enterprise Server. For more information, see the GHES Storage Partners repository.
>
> For storage products validated through the GitHub Technology Partnership program, the storage

provider is responsible for support and documentation for using the storage product with GitHub Actions.

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

- Create your Amazon S3 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 Amazon S3 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 an Amazon S3 bucket, you must first create an Amazon OIDC provider, then create an Identity and Access Management (IAM) role, and finally configure GitHub Enterprise Server to use the provider and role to access your S3 bucket.

1. Create an Amazon OIDC provider

1. Get the thumbprint for your GitHub Enterprise Server instance.

   a. Use the following OpenSSL command to get the SHA1 thumbprint for your GitHub Enterprise Server instance, replacing `HOSTNAME` with the public hostname for your GitHub Enterprise Server instance

   Shell

```
openssl s_client -connect HOSTNAME:443 < /dev/null 2>/dev/null |
openssl x509 -fingerprint -noout -sha1 -in /dev/stdin
```

For example:

```
openssl s_client -connect my-ghes-host.example.com:443 < /dev/null
2>/dev/null | openssl x509 -fingerprint -noout -sha1 -in /dev/stdin
```

The command returns a thumbprint in the following format:

```
SHA1
Fingerprint=AB:12:34:56:78:90:AB:CD:EF:12:34:56:78:90:AB:CD:EF:12:34:56
```

b. Remove the colons (`:`) from the thumbprint value, and save the value to use later.

For example, the thumbprint for the value returned in the previous step is:

```
AB1234567890ABCDEF1234567890ABCDEF123456
```

2 Using the AWS CLI, use the following command to create an OIDC provider for your GitHub Enterprise Server instance. Replace `HOSTNAME` with the public hostname for your GitHub Enterprise Server instance, and `THUMBPRINT` with the thumbprint value from the previous step.

Shell

```
aws iam create-open-id-connect-provider \
  --url https://HOSTNAME/_services/token \
  --client-id-list "sts.amazonaws.com" \
  --thumbprint-list "THUMBPRINT"
```

For example:

Shell

```
aws iam create-open-id-connect-provider \
  --url https://my-ghes-host.example.com/_services/token \
  --client-id-list "sts.amazonaws.com" \
  --thumbprint-list "AB1234567890ABCDEF1234567890ABCDEF123456"
```

For more information on installing the AWS CLI, see the Amazon documentation.

Warning: If the certificate for your GitHub Enterprise Server instance changes in the future, you must update the thumbprint value in the Amazon OIDC provider for the OIDC trust to continue to work.

2. Create an IAM role

1 Open the AWS Console, and navigate to the Identity and Access Management (IAM) service.

2. In the left menu, under "Access management", click **Roles**, then click **Create Role**.

3. On the "Select trusted entity" page, enter the following options:

   - For "Trusted entity type", click **Web identity**.
   - For "Identity provider", use the **Choose provider** drop-down menu and select the OIDC provider you created in the previous steps. It should be named `HOSTNAME/_services/token`, where `HOSTNAME` is the public hostname for your GitHub Enterprise Server instance.
   - For "Audience", select `sts.amazonaws.com`.

4. Click **Next**.

5. On the "Add permissions" page, use the filter to find and select the `AmazonS3FullAccess` policy.

6. Click **Next**.

7. On the "Name, review, and create" page, enter a name for the role, and click **Create role**.

8. On the IAM "Roles" page, select the role you just created.

9. Under "Summary", note the ARN value for the role, as this is needed later.

10. Click the **Trust relationships** tab, then click **Edit trust policy**.

11. Edit the trust policy to add a new `sub` claim. The value for `Condition` must match the following example, replacing `HOSTNAME` with the public hostname for your GitHub Enterprise Server instance:

```
...
"Condition": {
  "StringEquals": {
    "HOSTNAME/_services/token:aud": "sts.amazonaws.com",
    "HOSTNAME/_services/token:sub": "HOSTNAME"
  }
}
...
```

For example:

```
...
"Condition": {
  "StringEquals": {
    "my-ghes-host.example.com/_services/token:aud": "sts.amazonaws.com",
    "my-ghes-host.example.com/_services/token:sub": "my-ghes-
host.example.com"
  }
}
...
```

12. Click **Update policy**.

3. Configure GitHub Enterprise Server to connect to Amazon S3 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 "Amazon S3", click **Setup**.

7. Under "Authentication", select **OpenID Connect (OIDC)**, and enter the values for your storage:

   - **AWS S3 Bucket**: The name of your S3 bucket.
   - **AWS Role**: The ARN for the role you created in the previous procedures. For example, `arn:aws:iam::123456789:role/my-role-name`.
   - **AWS Region**: The AWS region for your bucket. For example, `us-east-1`.

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 Amazon S3 storage using access keys

1. Using the AWS Console or CLI, create an access key for your storage bucket. GitHub Actions requires the following permissions for the access key that will access the bucket:

   - `s3:PutObject`
   - `s3:GetObject`
   - `s3:ListBucketMultipartUploads`
   - `s3:ListMultipartUploadParts`
   - `s3:AbortMultipartUpload`
   - `s3:DeleteObject`
   - `s3:ListBucket`
   - `kms:GenerateDataKey` (if Key Management Service (KMS) encryption has been enabled)

   For more information on managing AWS access keys, see the "AWS Identity and Access Management Documentation."

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 "Amazon S3", click **Setup**.
8. Under "Authentication", select **Credentials-based**, and enter your storage bucket's details:
    - **AWS Service URL**: The service URL for your bucket. For example, if your S3 bucket was created in the `us-west-2` region, this value should be `https://s3.us-west-2.amazonaws.com`.

      For more information, see "AWS service endpoints" in the AWS documentation.
    - **AWS S3 Bucket**: The name of your S3 bucket.
    - **AWS S3 Access Key** and **AWS S3 Secret Key**: The AWS access key ID and secret key for your bucket.
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."