This version of GitHub Enterprise was discontinued on 2023-03-15. No patch releases will be made, even for critical security issues. For better performance, improved security, and new features, upgrade to the latest version of GitHub Enterprise. For help with the upgrade, contact GitHub Enterprise support.

Troubleshooting GitHub Actions for your enterprise

In this article

Troubleshooting common issues that occur when using GitHub Actions on GitHub Enterprise Server.

Who can use this feature
Site administrators can troubleshoot GitHub Actions issues and modify GitHub Enterprise Server configurations.

Checking the health of GitHub Actions

You can check the health of GitHub Actions on your GitHub Enterprise Server instance with the `ghe-actions-check` command-line utility. For more information, see "Command-line utilities" and "Accessing the administrative shell (SSH)."

Configuring self-hosted runners when using a self-signed certificate for GitHub Enterprise Server

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. For more information, see "Configuring TLS."

Installing the certificate on the runner machine

For a self-hosted runner to connect to a GitHub Enterprise Server using a self-signed certificate, you must install the certificate on the runner machine so that the connection is security hardened.

For the steps required to install a certificate, refer to the documentation for your runner's operating system.

Configuring Node.JS to use the certificate

Most actions are written in JavaScript and run using Node.js, which does not use the operating system certificate store. For the self-hosted runner application to use the certificate, you must set the `NODE_EXTRA_CA_CERTS` environment variable on the runner machine.

You can set the environment variable as a system environment variable, or declare it in a file named *.env* in the self-hosted runner application directory.

For example:

```
NODE_EXTRA_CA_CERTS=/usr/share/ca-certificates/extra/mycertfile.crt
```

Environment variables are read when the self-hosted runner application starts, so you must set the environment variable before configuring or starting the self-hosted runner application. If your certificate configuration changes, you must restart the self-hosted runner application.

Configuring Docker containers to use the certificate

If you use Docker container actions or service containers in your workflows, you might also need to install the certificate in your Docker image in addition to setting the above environment variable.

Configuring HTTP proxy settings for GitHub Actions

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 these settings aren't correctly configured, you might receive errors like `Resource unexpectedly moved to https://<IP_ADDRESS>` when setting or changing your GitHub Actions configuration.

Runners not connecting to GitHub Enterprise Server with a new hostname

Warning: Do not change the hostname for GitHub Enterprise Server after initial setup. Changing the hostname will cause unexpected behavior, up to and including instance outages. If you have changed the hostname for your instance and are experiencing problems, contact GitHub Enterprise Support or GitHub Premium Support.

If you deploy GitHub Enterprise Server in your environment with a new hostname and the old hostname no longer resolves to your instance, self-hosted runners will be unable to connect to the old hostname, and will not execute any jobs.

You will need to update the configuration of your self-hosted runners to use the new hostname for your GitHub Enterprise Server instance. Each self-hosted runner will require one of the following procedures:

- In the self-hosted runner application directory, edit the `.runner` and `.credentials` files to replace all mentions of the old hostname with the new hostname, then restart the self-hosted runner application.
- Remove the runner from GitHub Enterprise Server using the UI, and re-add it. For more information, see "Removing self-hosted runners" and "Adding self-hosted runners."

Stuck jobs and GitHub Actions memory and CPU limits

GitHub Actions is composed of multiple services running on your GitHub Enterprise Server instance. By default, these services are set up with default CPU and memory limits that should work for most instances. However, heavy users of GitHub Actions might need to adjust these settings.

You may be hitting the CPU or memory limits if you notice that jobs are not starting (even though there are idle runners), or if the job's progress is not updating or changing in the UI.

1. Check the overall CPU and memory usage in the management console

Access the management console and use the monitor dashboard to inspect the overall CPU and memory graphs under "System Health". For more information, see "Accessing the monitor dashboard."

If the overall "System Health" CPU usage is close to 100%, or there is no free memory left, then your GitHub Enterprise Server instance is running at capacity and needs to be scaled up. For more information, see "Increasing CPU or memory resources."

2. Check the Nomad Jobs CPU and memory usage in the management console

If the overall "System Health" CPU and memory usage is OK, scroll down the monitor dashboard page to the "Nomad Jobs" section, and look at the "CPU Percent Value" and "Memory Usage" graphs.

Each plot in these graphs corresponds to one service. For GitHub Actions services, look for:

- `mps_frontend`
- `mps_backend`
- `token_frontend`
- `token_backend`
- `actions_frontend`
- `actions_backend`

If any of these services are at or near 100% CPU utilization, or the memory is near their limit (2 GB by default), then the resource allocation for these services might need increasing. Take note of which of the above services are at or near their limit.

3. Increase the resource allocation for services at their limit

1. Log in to the administrative shell using SSH. For more information, see "Accessing the administrative shell (SSH)."

2. Run the following command to see what resources are available for allocation:

```
nomad node status -self
```

In the output, find the "Allocated Resources" section. It looks similar to the following example:

```
Allocated Resources
CPU              Memory          Disk
7740/49600 MHZ   23 GiB/32 GiB   4.4 GiB/7.9 GiB
```

For CPU and memory, this shows how much is allocated to the **total** of **all** services (the left value) and how much is available (the right value). In the example above, there is 23 GiB of memory allocated out of 32 GiB total. This means there is 9 GiB of memory available for allocation.

Warning: Be careful not to allocate more than the total available resources, or services will fail to start.

3. Change directory to `/etc/consul-templates/etc/nomad-jobs/actions`:

```
cd /etc/consul-templates/etc/nomad-jobs/actions
```

In this directory there are three files that correspond to the GitHub Actions services from above:

- `mps.hcl.ctmpl`
- `token.hcl.ctmpl`
- `actions.hcl.ctmpl`

4. For the services that you identified that need adjustment, open the corresponding file and locate the `resources` group that looks like the following:

```
resources {
  cpu = 512
  memory = 2048
  network {
    port "http" { }
  }
}
```

The values are in MHz for CPU resources, and MB for memory resources.

For example, to increase the resource limits in the above example to 1 GHz for the CPU and 4 GB of memory, change it to:

```
resources {
  cpu = 1024
  memory = 4096
  network {
    port "http" { }
  }
}
```

5. Save and exit the file.

6. Run `ghe-config-apply` to apply the changes.

When running `ghe-config-apply`, if you see output like `Failed to run nomad job`

`'/etc/nomad-jobs/<name>.hcl'`, then the change has likely over-allocated CPU or memory resources. If this happens, edit the configuration files again and lower the allocated CPU or memory, then re-run `ghe-config-apply`.

7. After the configuration is applied, run `ghe-actions-check` to verify that the GitHub Actions services are operational.

Troubleshooting failures when Dependabot triggers existing workflows

Note: Dependabot security and version updates are currently in public beta and subject to change.

After you set up Dependabot updates for your GitHub Enterprise Server instance, you may see failures when existing workflows are triggered by Dependabot events.

By default, GitHub Actions workflow runs that are triggered by Dependabot from `push`, `pull_request`, `pull_request_review`, or `pull_request_review_comment` events are treated as if they were opened from a repository fork. Unlike workflows triggered by other actors, this means they receive a read-only `GITHUB_TOKEN` and do not have access to any secrets that are normally available. This will cause any workflows that attempt to write to the repository to fail when they are triggered by Dependabot.

There are three ways to resolve this problem:

1. You can update your workflows so that they are no longer triggered by Dependabot using an expression like: `if: github.actor != 'dependabot[bot]'`. For more information, see "Expressions."

2. You can modify your workflows to use a two-step process that includes `pull_request_target` which does not have these limitations. For more information, see "Automating Dependabot with GitHub Actions."

3. You can provide workflows triggered by Dependabot access to secrets and allow the `permissions` term to increase the default scope of the `GITHUB_TOKEN`. For more information, see "Providing workflows triggered byDependabot access to secrets and increased permissions" below.

Providing workflows triggered by Dependabot access to secrets and increased permissions

1. Log in to the administrative shell using SSH. For more information, see "Accessing the administrative shell (SSH)."

2. To remove the limitations on workflows triggered by Dependabot on your GitHub Enterprise Server instance, use the following command.

```
$ ghe-config app.actions.disable-dependabot-enforcement true
```

3. Apply the configuration.

```
$ ghe-config-apply
```

4. Return to GitHub Enterprise Server.

Troubleshooting bundled actions in GitHub Actions

If you receive the following error when installing GitHub Actions in GitHub Enterprise Server, you can resolve the problem by installing the official bundled actions and starter workflows.

```
A part of the Actions setup had problems and needs an administrator to resolve.
```

To install the official bundled actions and starter workflows within a designated organization in GitHub Enterprise Server, follow this procedure.

1. Identify an organization that will store the official bundled actions and starter workflows. You can create a new organization or reuse an existing one.
   - To create a new organization, see "Creating a new organization from scratch."
   - For assistance with choosing a name for this organization, see "Getting started with GitHub Actions for GitHub Enterprise Server."
2. Log in to the administrative shell using SSH. For more information, see "Accessing the administrative shell (SSH)."
3. To designate your organization as the location to store the bundled actions, use the `ghe-config` command, replacing `ORGANIZATION` with the name of your organization.

   ```
   $ ghe-config app.actions.actions-org ORGANIZATION
   ```

   and:

   ```
   $ ghe-config app.actions.github-org ORGANIZATION
   ```

4. To add the bundled actions to your organization, unset the SHA.

   ```
   $ ghe-config --unset 'app.actions.actions-repos-sha1sum'
   ```

5. Apply the configuration.

   ```
   $ ghe-config-apply
   ```

After you've completed these steps, you can resume configuring GitHub Actions at "Getting started with GitHub Actions for GitHub Enterprise Server."