Configuring backups on your appliance

In this article

As part of a disaster recovery plan, you can protect production data on your GitHub Enterprise Server instance by configuring automated backups.

About GitHub Enterprise Server Backup Utilities

GitHub Enterprise Server Backup Utilities is a backup system you install on a separate host, which takes backup snapshots of your GitHub Enterprise Server instance at regular intervals over a secure SSH network connection. You can use a snapshot to restore an existing GitHub Enterprise Server instance to a previous state from the backup host.

Only data added since the last snapshot will transfer over the network and occupy additional physical storage space. To minimize performance impact, backups are performed online under the lowest CPU/IO priority. You do not need to schedule a maintenance window to perform a backup.

Major releases and version numbers for GitHub Enterprise Server Backup Utilities align with feature releases of GitHub Enterprise Server. We support the four most recent versions of both products. For more information, see "GitHub Enterprise Server releases."

For more detailed information on features, requirements, and advanced usage, see the GitHub Enterprise Server Backup Utilities README in the GitHub Enterprise Server Backup Utilities project documentation.

Prerequisites

To use GitHub Enterprise Server Backup Utilities, you must have a host system separate from your GitHub Enterprise Server instance. For details about how the system should be configured, see Requirements in the github/backup-utils repository.

You can also integrate GitHub Enterprise Server Backup Utilities into an existing environment for long-term permanent storage of critical data.

We recommend that the backup host and your GitHub Enterprise Server instance be

geographically distant from each other. This ensures that backups are available for recovery in the event of a major disaster or network outage at the primary site.

Physical storage requirements will vary based on Git repository disk usage and expected growth patterns:

Hardware	Recommendation
vCPUs	2
Memory	2 GB
Storage	Five times the primary instance's allocated storage

More resources may be required depending on your usage, such as user activity and selected integrations.

For more information, see GitHub Enterprise Server Backup Utilities requirements in the GitHub Enterprise Server Backup Utilities project documentation.

Installing GitHub Enterprise Server Backup Utilities

To install GitHub Enterprise Server Backup Utilities on your backup host, we recommend cloning the project's Git repository. This approach allows you to fetch new releases directly using Git, and your existing backup configuration file, `backup.config`, will be preserved when installing a new version.

Alternatively, if the host machine can't access the internet, you can download each GitHub Enterprise Server Backup Utilities release as a compressed archive, then extract and install the contents. For more information, see Getting started in the GitHub Enterprise Server Backup Utilities project documentation.

Backup snapshots are written to the disk path set by the `GHE_DATA_DIR` data directory variable in your `backup.config` file. Snapshots need to be stored on a filesystem which supports symbolic and hard links.

> **Note:** We recommend ensuring your snapshots are not kept in a subdirectory of the GitHub Enterprise Server Backup Utilities installation directory, to avoid inadvertently overwriting your data directory when upgrading GitHub Enterprise Server Backup Utilities versions.

1. To clone the GitHub Enterprise Server Backup Utilities project repository to a local directory on your backup host, run the following command.

   ```
   $ git clone https://github.com/github/backup-utils.git /path/to/target/director
   ```

2. To change into the local repository directory, run the following command.

   ```
   cd backup-utils
   ```

3. To update to the latest project release version, use the `stable` branch by running the `git checkout stable` command.

   ```
   git checkout stable
   ```

Alternatively, to use a specific project version, run the following command, replacing `X.Y.Z` with the desired release version.

```
$ git checkout vX.Y.Z
```

4. To copy the included `backup.config-example` file to `backup.config`, run the following command.

   ```
   cp backup.config-example backup.config
   ```

5. To customize your configuration, edit `backup.config` in a text editor.

   a. Set the `GHE_HOSTNAME` value to your primary GitHub Enterprise Server instance's hostname or IP address.

   > **Note:** If your GitHub Enterprise Server instance is deployed as a cluster or in a high availability configuration using a load balancer, the `GHE_HOSTNAME` can be the load balancer hostname, as long as it allows SSH access (on port 122) to your GitHub Enterprise Server instance.
   >
   > To ensure a recovered appliance is immediately available, perform backups targeting the primary instance even in a geo-replication configuration.

   b. Set the `GHE_DATA_DIR` value to the filesystem location where you want to store backup snapshots. We recommend choosing a location on the same filesystem as your backup host, but outside of where you cloned the Git repository in step 1.

6. To grant your backup host access to your instance, open your primary instance's settings page at `http(s)://HOSTNAME/setup/settings` and add the backup host's SSH key to the list of authorized SSH keys. For more information, see "Accessing the administrative shell (SSH)."

7. On your backup host, verify SSH connectivity with your GitHub Enterprise Server instance with the `ghe-host-check` command.

   ```
   ./bin/ghe-host-check
   ```

8. To create an initial full backup, run the following command.

   ```
   ./bin/ghe-backup
   ```

For more information on advanced usage, see the GitHub Enterprise Server Backup Utilities README in the GitHub Enterprise Server Backup Utilities project documentation.

Upgrading GitHub Enterprise Server Backup Utilities

When upgrading GitHub Enterprise Server Backup Utilities, you must choose a release that will work with your current version of GitHub Enterprise Server. Your installation of GitHub Enterprise Server Backup Utilities must be at least the same version as your

GitHub Enterprise Server instance, and cannot be more than two versions ahead. For more information, see GitHub Enterprise Server version requirements in the GitHub Enterprise Server Backup Utilities project documentation. You can upgrade GitHub Enterprise Server Backup Utilities in a Git repository by fetching and checking out the latest changes.

Alternatively, if you don't use a Git repository for your installation, you can extract a new archive into place, or you can change your approach to use a Git repository instead.

Verifying the installation type

You can verify the installation method for GitHub Enterprise Server Backup Utilities and determine the best way to upgrade your installation.

1. On your backup host, navigate to your GitHub Enterprise Server Backup Utilities directory, usually `backup-utils`.
2. To check if a valid working directory exists inside a Git repository, run the following command.

   ```
   git rev-parse --is-inside-work-tree
   ```

   If the output is `true`, GitHub Enterprise Server Backup Utilities was installed by cloning the project's Git repository. If the output includes `fatal: not a git repository (or any of the parent directories)`, GitHub Enterprise Server Backup Utilities was likely installed by extracting a compressed archive file. If your installation is in a Git repository, you can install the latest version using Git. If the installation is from a compressed archive file, you can either download and extract the latest version, or you can reinstall GitHub Enterprise Server Backup Utilities using Git to simplify future upgrades.

- Upgrading an installation in a Git repository
- Using Git instead of compressed archives for upgrades

Upgrading an installation in a Git repository

1. On your backup host, navigate to your GitHub Enterprise Server Backup Utilities directory, usually `backup-utils`.

   > **Note:** We recommend creating a copy of your existing `backup.config` file in a temporary location, like `$HOME/backup.config`, before upgrading GitHub Enterprise Server Backup Utilities.

2. Download the latest project updates by running the `git fetch` command.

   ```
   git fetch
   ```

3. To update to the latest project release version, use the `stable` branch by running the `git checkout stable` command.

   ```
   git checkout stable
   ```

   Alternatively, to use a specific project version, run the following command, replacing `X.Y.Z` with the desired release version.

```
$ git checkout vX.Y.Z
```

4. To verify you have successfully upgraded, run the following command.

```
./bin/ghe-backup --version
```

5. To verify SSH connectivity between your configured GitHub Enterprise Server, run the following command.

```
./bin/ghe-host-check
```

Using Git instead of compressed archives for upgrades

If your backup host has internet connectivity and you previously used a compressed archive (`.tar.gz`) to install or upgrade GitHub Enterprise Server Backup Utilities, we recommend using a Git repository for your installation instead. Upgrading using Git requires less work and preserves your backup configuration.

1. On your backup host, navigate to your GitHub Enterprise Server Backup Utilities directory, usually `backup-utils`.
2. To back up your existing GitHub Enterprise Server Backup Utilities configuration, copy your current `backup.config` file to a safe location, such as your home directory.

```
$ cp backup.config $HOME/backup.config.saved-$(date +%Y%m%d-%H%M%S)
```

3. Change to the local directory on your backup host where you want to install the GitHub Enterprise Server Backup Utilities Git repository.
4. To clone the project repository to the directory on your backup host, run the following command.

```
git clone https://github.com/github/backup-utils.git
```

5. To change into the cloned repository, run the following command.

```
cd backup-utils
```

6. To update to the latest project release version, use the `stable` branch by running the `git checkout stable` command.

```
git checkout stable
```

Alternatively, to use a specific project version, run the following command, replacing `X.Y.Z` with the desired release version.

```
$ git checkout vX.Y.Z
```

7. To restore your backup configuration from earlier, copy your existing backup configuration file to the local repository directory. Replace the path in the command with the location of the file saved in step 2.

```
$ cp PATH/TO/BACKUP/FROM/STEP/2 backup.config
```

> **Note:** You can choose where to restore your backup configuration file to after cloning. For more information about where configuration files can be located, see Getting started in the GitHub Enterprise Server Backup Utilities project documentation.

8. To confirm that the paths to directories or scripts in your backup configuration file are correct, review the file in a text editor.

9. To verify you have successfully upgraded, run the following command.

```
./bin/ghe-backup --version
```

10. To verify SSH connectivity between your configured GitHub Enterprise Server, run the following command.

```
./bin/ghe-host-check
```

11. Delete your old GitHub Enterprise Server Backup Utilities directory from step 1 (where the compressed archive installation was located).

Scheduling a backup

You can schedule regular backups on the backup host using the `cron(8)` command or a similar command scheduling service. The configured backup frequency will dictate the worst case recovery point objective (RPO) in your recovery plan. For example, if you have scheduled the backup to run every day at midnight, you could lose up to 24 hours of data in a disaster scenario. We recommend starting with an hourly backup schedule, guaranteeing a worst case maximum of one hour of data loss if the primary site data is destroyed.

If backup attempts overlap, the `ghe-backup` command will abort with an error message, indicating the existence of a simultaneous backup. If this occurs, we recommended decreasing the frequency of your scheduled backups. For more information, see the "Scheduling backups" section of the GitHub Enterprise Server Backup Utilities README in the GitHub Enterprise Server Backup Utilities project documentation.

Restoring a backup

In the event of prolonged outage or catastrophic event at the primary site, you can restore your GitHub Enterprise Server instance by provisioning another GitHub Enterprise appliance and performing a restore from the backup host. You must add the backup host's SSH key to the target GitHub Enterprise appliance as an authorized SSH key before restoring an appliance.

> **Note:** When performing backup restores to your GitHub Enterprise Server instance, the same version supportability rules apply. You can only restore data from at most two feature releases behind.

For example, if you take a backup from GitHub Enterprise Server 3.0.x, you can restore the backup to a GitHub Enterprise Server 3.2.x instance. You cannot restore data from a backup of GitHub Enterprise Server 2.22.x to an instance running 3.2.x, because that would be three jumps between versions (2.22 to 3.0 to 3.1 to 3.2). You would first need to restore to an instance running 3.1.x, and then upgrade to 3.2.x.

To restore your GitHub Enterprise Server instance from the last successful snapshot, use the `ghe-restore` command.

Note: Prior to restoring a backup, ensure:

- Maintenance mode is enabled on the primary instance and all active processes have completed. For more information, see "Enabling and scheduling maintenance mode."
- Replication is stopped on all replicas in high availability configurations. For more information, see the `ghe-repl-stop` command in "About high availability configuration."
- If your GitHub Enterprise Server instance has GitHub Actions enabled, you must first configure the GitHub Actions external storage provider on the replacement appliance. For more information, see "Backing up and restoring GitHub Enterprise Server with GitHub Actions enabled."

When running the `ghe-restore` command, you should see output similar to this:

```
$ ghe-restore -c 169.154.1.1
> Checking for leaked keys in the backup snapshot that is being restored ...
> * No leaked keys found
> Connect 169.154.1.1:122 OK (v2.9.0)

> WARNING: All data on GitHub Enterprise appliance 169.154.1.1 (v2.9.0)
>          will be overwritten with data from snapshot 20170329T150710.
> Please verify that this is the correct restore host before continuing.
> Type 'yes' to continue: yes

> Starting restore of 169.154.1.1:122 from snapshot 20170329T150710
# ...output truncated
> Completed restore of 169.154.1.1:122 from snapshot 20170329T150710
> Visit https://169.154.1.1/setup/settings to review appliance configuration.
```

Optionally, to validate the restore, configure an IP exception list to allow access to a specified list of IP addresses. For more information, see "Enabling and scheduling maintenance mode."

Note:

- The network settings are excluded from the backup snapshot. You must manually configure the network on the target GitHub Enterprise Server appliance as required for your environment.
- When restoring to new disks on an existing or empty GitHub Enterprise Server instance, stale UUIDs may be present, resulting in Git and/or Alambic replication reporting as out of sync. Stale server entry IDs can be the result of a retired node in a high availability configuration still being present in the application database, but not in the restored replication configuration. To remediate, stale UUIDs can be torn down using `ghe-repl-teardown` once the restore has completed and prior to starting replication. In this scenario, contact GitHub Enterprise Support for further assistance.

You can use these additional options with `ghe-restore` command:

- The `-c` flag overwrites the settings, certificate, and license data on the target host even if it is already configured. Omit this flag if you are setting up a staging instance for testing purposes and you wish to retain the existing configuration on the target. For more information, see the "Using backup and restore commands" section of the GitHub Enterprise Server Backup Utilities README in the GitHub Enterprise Server

Backup Utilities project documentation.

- The `-s` flag allows you to select a different backup snapshot.