Upgrading GitHub Enterprise Server

In this article

Upgrade GitHub Enterprise Server to get the latest features and security updates.

Preparing to upgrade

1. Determine an upgrade strategy and choose a version to upgrade to. For more information, see "Upgrade requirements" and refer to the Upgrade assistant to find the upgrade path from your current release version.
2. Create a fresh backup of your primary instance with the GitHub Enterprise Server Backup Utilities. For more information, see the README.md file in the GitHub Enterprise Server Backup Utilities project documentation.

   Note: Your GitHub Enterprise Server Backup Utilities version needs to be the same version as, or at most two versions ahead of, your GitHub Enterprise Server instance. For more information, see "Upgrading GitHub Enterprise Server Backup Utilities."

3. If your GitHub Enterprise Server instance uses ephemeral self-hosted runners for GitHub Actions and you've disabled automatic updates, upgrade your runners to the version of the runner application that your upgraded instance will run.
4. If you are upgrading using an upgrade package, schedule a maintenance window for GitHub Enterprise Server end users. If you are using a hotpatch, maintenance mode is not required.

   Note: The maintenance window depends on the type of upgrade you perform. Upgrades using a hotpatch usually don't require a maintenance window. Sometimes a reboot is required, which you can perform at a later time. Following the versioning scheme of MAJOR.FEATURE.PATCH, patch releases using an upgrade package typically require less than five minutes of downtime. Feature releases that include data migrations take longer depending on storage performance and the amount of data that's migrated. For more information, see "Enabling and scheduling maintenance mode."

Taking a snapshot

A snapshot is a checkpoint of a virtual machine (VM) at a point in time. We highly recommend taking a snapshot before upgrading your virtual machine so that if an upgrade fails, you can revert your VM back to the snapshot. We only recommend taking a VM snapshot when the appliance is powered down or in maintenance mode and all background jobs have finished.

If you're upgrading to a new feature release, you must take a VM snapshot. If you're upgrading to a patch release, you can attach the existing data disk.

There are two types of snapshots:

- **VM snapshots** save your entire VM state, including user data and configuration data. This snapshot method requires a large amount of disk space and is time consuming.
- **Data disk snapshots** only save your user data.

  > **Notes:**
  >
  > - Some platforms don't allow you to take a snapshot of just your data disk. For these platforms, you'll need to take a snapshot of the entire VM.
  > - If your hypervisor does not support full VM snapshots, you should take a snapshot of the root disk and data disk in quick succession.

Platform	Snapshot method	Snapshot documentation URL
Amazon AWS	Disk	https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html
Azure	VM	https://docs.microsoft.com/azure/backup/backup-azure-vms-first-look-arm
Hyper-V	VM	https://docs.microsoft.com/windows-server/virtualization/hyper-v/manage/enable-or-disable-checkpoints-in-hyper-v
Google Compute Engine	Disk	https://cloud.google.com/compute/docs/disks/create-snapshots
VMware	VM	https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.hostclient.doc/GUID-64B866EF-7636-401C-A8FF-2B4584D9CA72.html

Upgrading with a hotpatch

You can upgrade GitHub Enterprise Server to the latest patch release using a hotpatch.

You can use hotpatching to upgrade to a newer patch release, but not a feature release. For example, you can upgrade from `2.10.1` to `2.10.5` because they are in the same feature series, but not from `2.10.9` to `2.11.0` because they are in a different feature series.

Hotpatches do not generally require a reboot. If a hotpatch does require a reboot, the GitHub Enterprise Server release notes will indicate the requirement.

Hotpatches require a configuration run, which can cause a brief period of errors or unresponsiveness for some or all services on your GitHub Enterprise Server instance. You are not required to enable maintenance mode during installation of a hotpatch, but doing

so will guarantee that users see a maintenance page instead of errors or timeouts. For more information, see "Enabling and scheduling maintenance mode."

Using the Management Console, you can install a hotpatch immediately or schedule it for later installation. You can use the administrative shell to install a hotpatch with the `ghe-upgrade` utility. For more information, see "Upgrade requirements."

Notes:

- If your GitHub Enterprise Server instance is running a release candidate build, you can't upgrade with a hotpatch.
- Installing a hotpatch using the Management Console is not available in clustered environments. To install a hotpatch in a clustered environment, see "Upgrading a cluster."

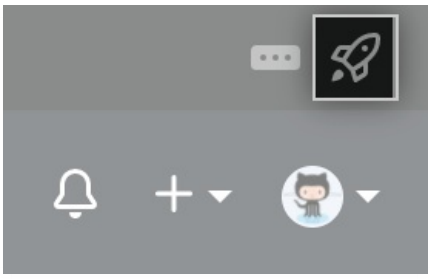
Upgrading a single appliance with a hotpatch

Installing a hotpatch using the Management Console

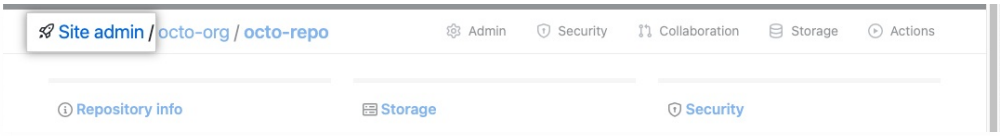
You can use the Management Console to upgrade with a hotpatch by enabling automatic updates. You will then be presented with the latest available version of GitHub Enterprise Server that you can upgrade to.

If the upgrade target you're presented with is a feature release instead of a patch release, you cannot use the Management Console to install a hotpatch. You must install the hotpatch using the administrative shell instead. For more information, see "Installing a hotpatch using the administrative shell."

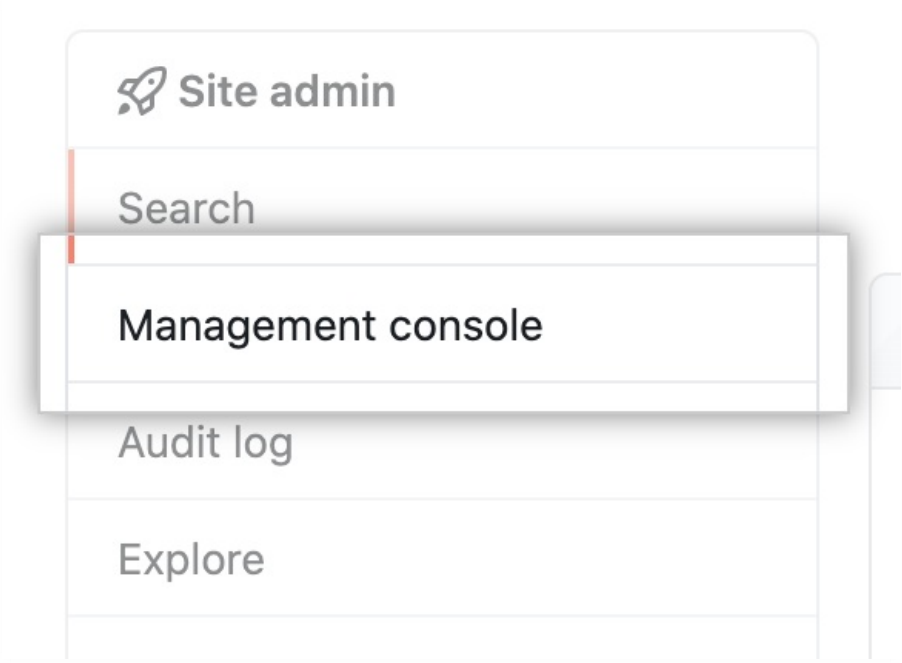
1. Enable automatic updates. For more information, see "Enabling automatic updates."
2. From an administrative account on GitHub Enterprise Server, in the upper-right corner of any page, click 🚀.



3. If you're not already on the "Site admin" page, in the upper-left corner, click **Site admin**.

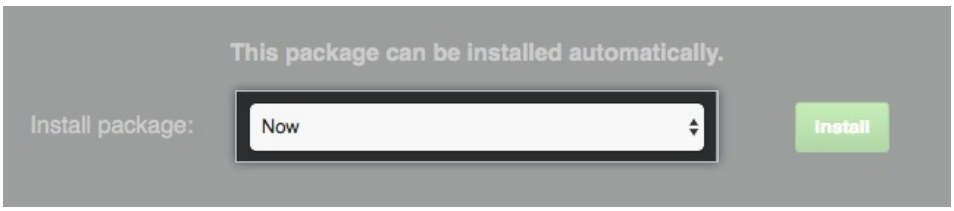


4. In the left sidebar, click **Management Console**.

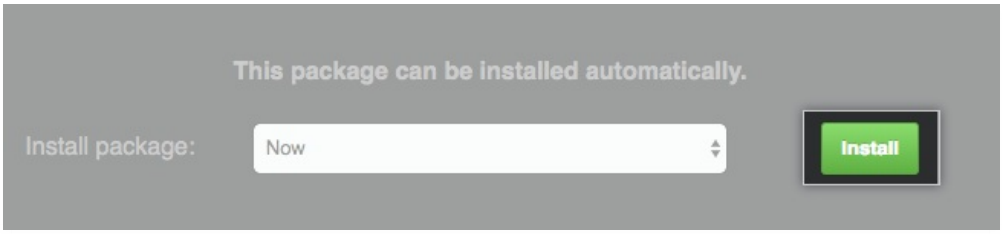


5. At the top of the Management Console, click **Updates**.

6. When a new hotpatch has been downloaded, use the Install package drop-down menu:
   - To install immediately, select **Now**:
   - To install later, select a later date.



7. Click **Install**.



Installing a hotpatch using the administrative shell

Note: If you've enabled automatic update checks, you don't need to download the upgrade package and can use the file that was automatically downloaded. For more information, see

"Enabling automatic update checks."

1. SSH into your GitHub Enterprise Server instance. If your instance comprises multiple nodes, for example if high availability or geo-replication are configured, SSH into the primary node. If you use a cluster, you can SSH into any node. For more information about SSH access, see "Accessing the administrative shell (SSH)."

   ```
   $ ssh -p 122 admin@HOSTNAME
   ```

2. Browse to the GitHub Enterprise Server Releases page. Next to the release you are upgrading to, click **Download**, then click the **Upgrading** tab. Copy the URL for the upgrade hotpackage (*.hpkg* file).

3. Download the upgrade package to your GitHub Enterprise Server instance using `curl`:

   ```
   admin@HOSTNAME:~$ curl -L -O UPGRADE-PKG-URL
   ```

4. Run the `ghe-upgrade` command using the package file name:

   ```
   admin@HOSTNAME:~$ ghe-upgrade GITHUB-UPGRADE.hpkg
   *** verifying upgrade package signature...
   ```

5. If a reboot is required for updates for kernel, MySQL, Elasticsearch or other programs, the hotpatch upgrade script notifies you.

Upgrading an appliance that has replica instances using a hotpatch

Note: If you are installing a hotpatch, you do not need to enter maintenance mode or stop replication.

Appliances configured for high-availability and geo-replication use replica instances in addition to primary instances. To upgrade these appliances, you'll need to upgrade both the primary instance and all replica instances, one at a time.

Upgrading the primary instance

1. Upgrade the primary instance by following the instructions in "Installing a hotpatch using the administrative shell."

Upgrading a replica instance

Note: If you're running multiple replica instances as part of geo-replication, repeat this procedure for each replica instance, one at a time.

1. Upgrade the replica instance by following the instructions in "Installing a hotpatch using the administrative shell." If you are using multiple replicas for Geo-replication, you must repeat this procedure to upgrade each replica one at a time.

2. Connect to the replica instance over SSH as the "admin" user on port 122:

```
$ ssh -p 122 admin@REPLICA_HOST
```

3. Verify the upgrade by running:

```
$ ghe-version
```

Upgrading with an upgrade package

While you can use a hotpatch to upgrade to the latest patch release within a feature series, you must use an upgrade package to upgrade to a newer feature release. For example to upgrade from `2.11.10` to `2.12.4` you must use an upgrade package since these are in different feature series. For more information, see "Upgrade requirements."

Upgrading a single appliance with an upgrade package

> **Note:** If you've enabled automatic update checks, you don't need to download the upgrade package and can use the file that was automatically downloaded. For more information, see "Enabling automatic update checks."

1. SSH into your GitHub Enterprise Server instance. If your instance comprises multiple nodes, for example if high availability or geo-replication are configured, SSH into the primary node. If you use a cluster, you can SSH into any node. For more information about SSH access, see "Accessing the administrative shell (SSH)."

```
$ ssh -p 122 admin@HOSTNAME
```

2. Browse to the GitHub Enterprise Server Releases page. Next to the release you are upgrading to, click **Download**, then click the **Upgrading** tab. Select the appropriate platform and copy the URL for the upgrade package (*.pkg* file).
3. Download the upgrade package to your GitHub Enterprise Server instance using `curl`:

```
admin@HOSTNAME:~$ curl -L -O UPGRADE-PKG-URL
```

4. Enable maintenance mode and wait for all active processes to complete on the GitHub Enterprise Server instance. For more information, see "Enabling and scheduling maintenance mode."

> **Note**: When upgrading the primary appliance in a High Availability configuration, the appliance should already be in maintenance mode if you are following the instructions in "Upgrading the primary instance."

5. Run the `ghe-upgrade` command using the package file name:

```
admin@HOSTNAME:~$ ghe-upgrade GITHUB-UPGRADE.pkg
*** verifying upgrade package signature...
```

6. Confirm that you'd like to continue with the upgrade and restart after the package signature verifies. The new root filesystem writes to the secondary partition and the instance automatically restarts in maintenance mode:

```
*** applying update...
This package will upgrade your installation to version VERSION-NUMBER
Current root partition: /dev/xvda1 [VERSION-NUMBER]
Target root partition:  /dev/xvda2
Proceed with installation? [y/N]
```

7. For single appliance upgrades, disable maintenance mode so users can use your GitHub Enterprise Server instance.

> **Note**: When upgrading appliances in a High Availability configuration you should remain in maintenance mode until you have upgraded all of the replicas and replication is current. For more information, see "Upgrading a replica instance."

Upgrading an appliance that has replica instances using an upgrade package

Appliances configured for high-availability and geo-replication use replica instances in addition to primary instances. To upgrade these appliances, you'll need to upgrade both the primary instance and all replica instances, one at a time.

Upgrading the primary instance

> **Warning:** When replication is stopped, if the primary fails, any work that is done before the replica is upgraded and the replication begins again will be lost.

1. On the primary instance, enable maintenance mode and wait for all active processes to complete. For more information, see "Enabling maintenance mode."
2. Connect to the replica instance over SSH as the "admin" user on port 122:

```
$ ssh -p 122 admin@REPLICA_HOST
```

3. On the replica instance, or on all replica instances if you're running multiple replica instances as part of geo-replication, run `ghe-repl-stop` to stop replication.
4. Upgrade the primary instance by following the instructions in "Upgrading a single appliance with an upgrade package."

Upgrading a replica instance

> **Note:** If you're running multiple replica instances as part of geo-replication, repeat this procedure for each replica instance, one at a time.

1. Upgrade the replica instance by following the instructions in "Upgrading a single appliance with an upgrade package." If you are using multiple replicas for Geo-

replication, you must repeat this procedure to upgrade each replica one at a time.

2. Connect to the replica instance over SSH as the "admin" user on port 122:

```
$ ssh -p 122 admin@REPLICA_HOST
```

3. Verify the upgrade by running:

```
$ ghe-version
```

4. On the replica instance, to start replication, run `ghe-repl-start`.

5. On the replica instance, to make sure replication services are running correctly, run `ghe-repl-status`. This command will return `OK` for all services when a successful replication is in progress and the replica has upgraded. If the command returns `Replication is not running`, the replication may still be starting. Wait about one minute before running `ghe-repl-status` again.

   > **Note:** While the resync is in progress `ghe-repl-status` may indicate that replication is behind. For example, you may see the following message.
   >
   > ```
   > CRITICAL: git replication is behind the primary by more than 1007 repositories and/or g
   > ```

   - If `ghe-repl-status` did not return `OK`, contact GitHub Enterprise Support. For more information, see "Receiving help from GitHub Support."

6. When you have completed upgrading the last replica, and the resync is complete, disable maintenance mode so users can use your GitHub Enterprise Server instance.

Restoring from a failed upgrade

If an upgrade fails or is interrupted, you should revert your instance back to its previous state. The process for completing this depends on the type of upgrade.

Rolling back a patch release

To roll back a patch release, use the `ghe-upgrade` command with the `--allow-patch-rollback` switch. Before rolling back, replication must be temporarily stopped by running `ghe-repl-stop` on all replica instances. When rolling back an upgrade, you must use an upgrade package file with the *.pkg* extension. Hotpatch package files with the *.hpkg* extension are not supported.

```
ghe-upgrade --allow-patch-rollback EARLIER-RELEASE-UPGRADE-PACKAGE.pkg
```

A reboot is required after running the command. Rolling back does not affect the data partition, as migrations are not run on patch releases.

Once the rollback is complete, restart replication by running `ghe-repl-start` on all replicas.

For more information, see "Command-line utilities."

Rolling back a feature release

To roll back from a feature release, restore from a VM snapshot to ensure that root and data partitions are in a consistent state. For more information, see "Taking a snapshot."

Further reading

- "About upgrades to new releases"