Upgrading a cluster

In this article

To upgrade a GitHub Enterprise Server cluster to the latest release, use the administrative shell (SSH).

GitHub determines eligibility for clustering, and must enable the configuration for your instance's license. Clustering requires careful planning and additional administrative overhead. For more information, see "About clustering."

About upgrades to a GitHub Enterprise Server cluster

GitHub Enterprise Server is constantly improving, with new functionality and bug fixes introduced through feature and patch releases. You are responsible for upgrades to your instance. For more information, see "About upgrades to new releases."

To upgrade an instance, you must plan and communicate the upgrade, choose the appropriate package, back up your data, and then perform the upgrade.

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." The hotpatch installation script installs the hotpatch on every node in the cluster and restarts the services in their proper sequence to avoid downtime.

1. Back up your data with GitHub Enterprise Server Backup Utilities.
2. From the administrative shell of any node, use the `ghe-cluster-hotpatch` command to install the latest hotpatch. You can provide a URL for a hotpatch, or manually download the hotpatch and specify a local filename.

   ```
   $ ghe-cluster-hotpatch https://HOTPATCH-URL/FILENAME.hpkg
   ```

Upgrading with an upgrade package

Use an upgrade package to upgrade a GitHub Enterprise Server cluster to the latest feature release. For example, you can upgrade from `2.11` to `2.13`.

Preparing to upgrade

1. Review Cluster network configuration for the version you are upgrading to, and update your configuration as needed.
2. Back up your data with GitHub Enterprise Server Backup Utilities.
3. Schedule a maintenance window for end users of your GitHub Enterprise Server cluster, as it will be unavailable for normal use during the upgrade. Maintenance mode blocks user access and prevents data changes while the cluster upgrade is in progress.
4. On the GitHub Enterprise Server Download Page, copy the URL for the upgrade *.pkg* file to the clipboard.
5. From the administrative shell of any node, use the `ghe-cluster-each` command combined with `curl` to download the release package to each node in a single step. Use the URL you copied in the previous step as an argument.

   ```
   $ ghe-cluster-each -- "cd /home/admin && curl -L -O  https://PACKAGE-URL.pkg"
   > ghe-app-node-1:   % Total    % Received % Xferd  Average Speed   Time    Time
   > ghe-app-node-1:                                 Dload  Upload   Total   Sper
   > 100  496M  100  496M    0     0  24.2M      0  0:00:20  0:00:20 --:--:-- 27.4
   > ghe-data-node-2:   % Total    % Received % Xferd  Average Speed   Time    Tim
   > ghe-data-node-2:                                 Dload  Upload   Total   Spe
   > 100  496M  100  496M    0     0  21.3M      0  0:00:23  0:00:23 --:--:-- 25.8
   > ghe-data-node-1:   % Total    % Received % Xferd  Average Speed   Time    Tim
   > ghe-data-node-1:                                 Dload  Upload   Total   Spe
   > 100  496M  100  496M    0     0  19.7M      0  0:00:25  0:00:25 --:--:-- 25.6
   > ghe-app-node-2:   % Total    % Received % Xferd  Average Speed   Time    Time
   > ghe-app-node-2:                                 Dload  Upload   Total   Sper
   > 100  496M  100  496M    0     0  19.8M      0  0:00:25  0:00:25 --:--:-- 17.6
   > ghe-data-node-3:   % Total    % Received % Xferd  Average Speed   Time    Tim
   > ghe-data-node-3:                                 Dload  Upload   Total   Spe
   > 100  496M  100  496M    0     0  19.7M      0  0:00:25  0:00:25 --:--:-- 25.5
   ```

6. Identify the primary MySQL node, which is defined as `mysql-master = <hostname>` in `cluster.conf`. This node will be upgraded last.

Upgrading the cluster nodes

1. Enable maintenance mode according to your scheduled window by connecting to

the administrative shell of any cluster node and running `ghe-cluster-maintenance -s`.

2. **With the exception of the primary MySQL node**, connect to the administrative shell of each of the GitHub Enterprise Server nodes. Run the `ghe-upgrade` command, providing the package file name you downloaded in Step 4 of Preparing to upgrade:

```
$ ghe-upgrade PACKAGE-FILENAME.pkg
   > *** verifying upgrade package signature...
   >  497MB 0:00:04 [ 117MB/s] [=========================================>] 10
   > gpg: Signature made Fri 19 Feb 2016 02:33:50 PM UTC using RSA key ID 0D65D
   > gpg: checking the trustdb
   > gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
   > gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
   > gpg: Good signature from "GitHub Enterprise (Upgrade Package Key) > "
```

3. The upgrade process will reboot the node once it completes. Verify that you can `ping` each node after it reboots.

4. Connect to the administrative shell of the primary MySQL node. Run the `ghe-upgrade` command, providing the package file name you downloaded in Step 4 of Preparing to upgrade:

```
$ ghe-upgrade PACKAGE-FILENAME.pkg
> *** verifying upgrade package signature...
>  497MB 0:00:04 [ 117MB/s] [=========================================>] 100%
> gpg: Signature made Fri 19 Feb 2016 02:33:50 PM UTC using RSA key ID 0D65D57A
> gpg: checking the trustdb
> gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
> gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
> gpg: Good signature from "GitHub Enterprise (Upgrade Package Key) > "
```

5. The upgrade process will reboot the primary MySQL node once it completes. Verify that you can `ping` each node after it reboots.

6. Connect to the administrative shell of the primary MySQL node and run the `ghe-cluster-config-apply` command.

7. When `ghe-cluster-config-apply` is complete, check that the services are in a healthy state by running `ghe-cluster-status`.

8. Exit maintenance mode from the administrative shell of any node by running `ghe-cluster-maintenance -u`.