Command-line utilities

In this article

GitHub Enterprise Server includes a variety of utilities to help resolve particular problems or perform specific tasks.

You can execute these commands from anywhere on the VM after signing in as an SSH admin user. For more information, see "Accessing the administrative shell (SSH)."

General

ghe-announce

This utility sets a banner at the top of every GitHub Enterprise page. You can use it to broadcast a message to your users.

```
# Sets a message that's visible to everyone
$ ghe-announce -s MESSAGE
> Announcement message set.
# Removes a previously set message
$ ghe-announce -u
> Removed the announcement message
```

You can also set an announcement banner using the enterprise settings on GitHub Enterprise Server. For more information, see "Customizing user messages for your enterprise."

ghe-aqueduct

This utility displays information on background jobs, both active and in the queue. It provides the same job count numbers as the admin stats bar at the top of every page.

This utility can help identify whether the Aqueduct server is having problems processing background jobs. Any of the following scenarios might be indicative of a problem with Aqueduct:

- The number of background jobs is increasing, while the active jobs remain the same.
- The event feeds are not updating.
- Webhooks are not being triggered.
- The web interface is not updating after a Git push.

If you suspect Aqueduct is failing, contact GitHub Enterprise Support for help.

With this command, you can also pause or resume jobs in the queue.

```
$ ghe-aqueduct status
# lists queues and the number of currently queued jobs for all queues
$ ghe-aqueduct queue_depth --queue QUEUE
# lists the number of currently queued jobs for the specified queue
$ ghe-aqueduct pause --queue QUEUE
# pauses the specified queue
$ ghe-aqueduct resume --queue QUEUE
# resumes the specified queue
```

ghe-check-disk-usage

This utility checks the disk for large files or files that have been deleted but still have open file handles. This should be run when you're trying to free up space on the root partition.

```
ghe-check-disk-usage
```

ghe-cleanup-caches

This utility cleans up a variety of caches that might potentially take up extra disk space on the root volume. If you find your root volume disk space usage increasing notably over time it would be a good idea to run this utility to see if it helps reduce overall usage.

```
ghe-cleanup-caches
```

ghe-cleanup-settings

This utility wipes all existing Management Console settings.

Tip: Typically, you will only execute this if you've contacted support and they've asked you to do so.

```
ghe-cleanup-settings
```

ghe-config

With this utility, you can both retrieve and modify the configuration settings of your GitHub Enterprise Server instance.

```
$ ghe-config core.github-hostname
# Gets the configuration value of `core.github-hostname`
```

```
$ ghe-config core.github-hostname URL
# Sets the configuration value of `core.github-hostname` to the specified URL
$ ghe-config -l
# Lists all the configuration values
```

Allows you to find the universally unique identifier (UUID) of your node in `cluster.conf`.

```
$ ghe-config HOSTNAME.uuid
```

Allows you to exempt a list of users from REST API rate limits. A hard limit of 120,000 requests will still apply to these users. Usernames you provide for this command are case-sensitive. For more information, see "Resources in the REST API."

```
$ ghe-config app.github.rate-limiting-exempt-users "hubot github-actions[bot]"
# Exempts the users hubot and github-actions[bot] from rate limits. Usernames are ca
```

ghe-config-apply

This utility applies Management Console settings, reloads system services, prepares a storage device, reloads application services, and runs any pending database migrations. It is equivalent to clicking **Save settings** in the Management Console's web UI or to sending a POST request to the `/setup/api/configure` endpoint.

You will probably never need to run this manually, but it's available if you want to automate the process of saving your settings via SSH.

```
ghe-config-apply
```

ghe-console

This utility opens the GitHub Rails console on your GitHub Enterprise appliance. **Do not use** this command without direction from GitHub Enterprise Support. Incorrect use could cause damage or data loss.

```
ghe-console
```

ghe-dbconsole

This utility opens a MySQL database session on your GitHub Enterprise appliance. **Do not use** this command without direction from GitHub Enterprise Support. Incorrect use could cause damage or data loss.

```
ghe-dbconsole
```

ghe-es-index-status

This utility returns a summary of Elasticsearch indexes in CSV format.

Print an index summary with a header row to `STDOUT`:

```
$ ghe-es-index-status -do
> warning: parser/current is loading parser/ruby23, which recognizes
> warning: 2.3.3-compliant syntax, but you are running 2.3.4.
> warning: please see https://github.com/whitequark/parser#compatibility-with-ruby-r
```

```
> Name,Primary,Searchable,Writable,UpToDate,RepairProgress,Version
> code-search-1,true,true,true,true,100.0,72e27df7c631b45e026b42bfef059328fa040e17
> commits-5,true,true,true,true,100.0,7ed28813100c47813ef654c0ee2bb9abf21ab744
> gists-4,true,true,true,true,100.0,cf8e7d04fcf2564c902e2873c424a279cc41079d
> issues-4,false,false,false,true,100.0,d0bb08f71eebf6e7b070572aa399b185dbdc8a76
> issues-5,true,true,true,true,100.0,d0bb08f71eebf6e7b070572aa399b185dbdc8a76
> projects-2,true,true,true,true,100.0,c5cac1c4b3c66d42e609d088d174dbc3dd44469a
> pull-requests-6,true,true,true,true,100.0,6a466ad6b896a3499509990979bf9a18d7d41de
> repos-6,true,true,true,true,100.0,6c8b5fbba0fc1e409558db411d05e092c1387082
> users-5,true,true,true,true,100.0,38984875552bb826c9ec42999f409cb2e95556eb
> wikis-4,true,true,true,true,100.0,2613dec44bd14e14577803ac1f9e4b7e07a7c234
```

Print an index summary and pipe results to `column` for readability:

```
$ ghe-es-index-status -do | column -ts,
> warning: parser/current is loading parser/ruby23, which recognizes
> warning: 2.3.3-compliant syntax, but you are running 2.3.4.
> warning: please see https://github.com/whitequark/parser#compatibility-with-ruby-
> Name             Primary  Searchable  Writable  UpToDate  RepairProgress  Version
> code-search-1    true     true        true      true      100.0           72e27df
> commits-5        true     true        true      true      100.0           7ed2881
> gists-4          true     true        true      true      100.0           cf8e7d0
> issues-4         false    false       false     true      100.0           d0bb08f
> issues-5         true     true        true      true      100.0           d0bb08f
> projects-2       true     true        true      true      100.0           c5cac1c
> pull-requests-6  true     true        true      true      100.0           6a466ad
> repos-6          true     true        true      true      100.0           6c8b5fb
> users-5          true     true        true      true      100.0           3898487
> wikis-4          true     true        true      true      100.0           2613dec
```

ghe-legacy-github-services-report

This utility lists repositories on your appliance that use GitHub Services, an integration method that will be discontinued on October 1, 2018. Users on your appliance may have set up GitHub Services to create notifications for pushes to certain repositories. For more information, see "Announcing the deprecation of GitHub Services" on the GitHub Blog or "Replacing GitHub Services." For more information about this command or for additional options, use the `-h` flag.

```
ghe-legacy-github-services-report
```

ghe-logs-tail

This utility lets you tail log all relevant log files from your installation. You can pass options in to limit the logs to specific sets. Use the -h flag for additional options.

```
ghe-logs-tail
```

ghe-maintenance

This utility allows you to control the state of the installation's maintenance mode. It's designed to be used primarily by the Management Console behind-the-scenes, but it can be used directly. For more information, see "Enabling and scheduling maintenance mode."

```
ghe-maintenance -h
```

ghe-motd

This utility re-displays the message of the day (MOTD) that administrators see when accessing the instance via the administrative shell. The output contains an overview of the instance's state.

```
ghe-motd
```

ghe-nwo

This utility returns a repository's name and owner based on the repository ID.

```
ghe-nwo REPOSITORY_ID
```

ghe-org-admin-promote

Use this command to give organization owner privileges to users with site admin privileges on the appliance, or to give organization owner privileges to any single user in a single organization. You must specify a user and/or an organization. The `ghe-org-admin-promote` command will always ask for confirmation before running unless you use the `-y` flag to bypass the confirmation.

You can use these options with the utility:

- The `-u` flag specifies a username. Use this flag to give organization owner privileges to a specific user. Omit the `-u` flag to promote all site admins to the specified organization.
- The `-o` flag specifies an organization. Use this flag to give owner privileges in a specific organization. Omit the `-o` flag to give owner permissions in all organizations to the specified site admin.
- The `-a` flag gives owner privileges in all organizations to all site admins.
- The `-y` flag bypasses the manual confirmation.

This utility cannot promote a non-site admin to be an owner of all organizations. You can promote an ordinary user account to a site admin with ghe-user-promote.

Give organization owner privileges in a specific organization to a specific site admin

```
ghe-org-admin-promote -u USERNAME -o ORGANIZATION
```

Give organization owner privileges in all organizations to a specific site admin

```
ghe-org-admin-promote -u USERNAME
```

Give organization owner privileges in a specific organization to all site admins

```
ghe-org-admin-promote -o ORGANIZATION
```

Give organization owner privileges in all organizations to all site admins

```
ghe-org-admin-promote -a
```

ghe-reactivate-admin-login

Use this command to immediately unlock the Management Console after 10 failed login attempts in the span of 10 minutes.

```
$ ghe-reactivate-admin-login
```

ghe-saml-mapping-csv

This utility allows administrators to output or update the SAML `NameID` mappings for users on an instance. The utility can output a CSV file that lists all existing mappings. You can also update mappings for users on your instance by editing the resulting file, then using the utility to assign new mappings from the file.

To output a CSV file containing a list of all user SAML `NameID` mappings on the instance, run the following command.

```
ghe-saml-mapping-csv -d
```

After output completes, the utility displays the path to the file. The default path for output depends on the patch release of GitHub Enterprise Server 3.4 your instance is running.

- In version 3.4.17 and earlier, the utility writes the file to `/tmp`.
- In version 3.4.18 and later, the utility writes the file to `/data/user/tmp`.

If you plan to update mappings, to ensure that the utility can access the file, we recommend that you keep the file in the default location.

To prepare to update mappings, edit the file and make the desired changes. To see the result of updating the mappings using the new values in your edited CSV file, perform a dry run. Run the following command, replacing /PATH/TO/FILE with the actual path to the file you edited.

```
ghe-saml-mapping-csv -u -n -f /PATH/TO/FILE
```

To update SAML mappings on the instance with new values from the file, run the following command, replacing /PATH/TO/FILE with the actual path to the file you edited.

```
ghe-saml-mapping-csv -u -f /PATH/TO/FILE
```

ghe-service-list

This utility lists all of the services that have been started or stopped (are running or waiting) on your appliance.

```
$ ghe-service-list

start/running
  - github-resqued, process 12711
  - github-unicorn, process 12726
  - github-gitauth, process 12743
  - git-daemon, process 12755
  - babeld, process 12771
  - github-svn-proxy, process 12802
  - gist-unicorn, process 12832
  - gist-resqued, process 12881
  - render-unicorn, process 12939
  - hookshot-unicorn, process 13076
  - nodeload2, process 13192
```

```
  - slumlord-unicorn, process 13304
  - ghe-storage, process 2012
  - enterprise-manage-unicorn, process 2024
  - enterprise-manage-resque, process 2053
stop/waiting
  - ghe-replica-mode
```

ghe-set-password

With `ghe-set-password`, you can set a new password to authenticate into the Management Console.

```
ghe-set-password
```

ghe-setup-network

This utility allows you to configure the primary network interface.

To enter visual mode, which will guide you through configuration of network settings:

```
$ ghe-setup-network -v
```

Use the -h flag for additional options.

ghe-ssh-check-host-keys

This utility checks the existing SSH host keys against the list of known leaked SSH host keys.

```
$ ghe-ssh-check-host-keys
```

If a leaked host key is found the utility exits with status `1` and a message:

```
> One or more of your SSH host keys were found in the blacklist.
> Please reset your host keys using ghe-ssh-roll-host-keys.
```

If a leaked host key was not found, the utility exits with status `0` and a message:

```
> The SSH host keys were not found in the SSH host key blacklist.
> No additional steps are needed/recommended at this time.
```

ghe-ssh-roll-host-keys

This utility rolls the SSH host keys and replaces them with newly generated keys.

```
$ sudo ghe-ssh-roll-host-keys
Proceed with rolling SSH host keys? This will delete the
existing keys in /etc/ssh/ssh_host_* and generate new ones. [y/N]

# Press 'Y' to confirm deleting, or use the -y switch to bypass this prompt

> SSH host keys have successfully been rolled.
```

ghe-ssh-weak-fingerprints

This utility returns a report of known weak SSH keys stored on the GitHub Enterprise appliance. You can optionally revoke user keys as a bulk action. The utility will report weak system keys, which you must manually revoke in the Management Console.

```
# Print a report of weak user and system SSH keys
$ ghe-ssh-weak-fingerprints

# Revoke all weak user keys
$ ghe-ssh-weak-fingerprints --revoke
```

ghe-ssl-acme

This utility allows you to install a Let's Encrypt certificate on your GitHub Enterprise appliance. For more information, see "Configuring TLS."

You can use the `-x` flag to remove the ACME configuration.

```
ghe-ssl-acme -e
```

ghe-ssl-ca-certificate-install

This utility allows you to install a custom root CA certificate on your GitHub Enterprise server. The certificate must be in PEM format. Furthermore, if your certificate provider includes multiple CA certificates in a single file, you must separate them into individual files that you then pass to `ghe-ssl-ca-certificate-install` one at a time.

Run this utility to add a certificate chain for S/MIME commit signature verification. For more information, see "About commit signature verification."

Run this utility when your GitHub Enterprise Server instance is unable to connect to another server because the latter is using a self-signed SSL certificate or an SSL certificate for which it doesn't provide the necessary CA bundle. One way to confirm this is to run `openssl s_client -connect host:port -verify 0 -CApath /etc/ssl/certs` from your GitHub Enterprise Server instance. If the remote server's SSL certificate can be verified, your `SSL-Session` should have a return code of 0, as shown below.

```
SSL-Session:
    Protocol  : TLSv1
    Cipher    : AES128-SHA
    Session-ID: C794EBCC3CBC10F747C9AFC029C03C1048FC99CFC34D13D7444E0F267C58DF4C
    Session-ID-ctx:
    Master-Key: 02A7C47CFD6EEC87D3C710E9DD87390E04EF82DDD7514AE03127D5DC1945FC0CAEFE
    Key-Arg   : None
    Start Time: 1394581597
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
```

If, on the other hand, the remote server's SSL certificate can *not* be verified, your `SSL-Session` should have a nonzero return code:

```
SSL-Session:
    Protocol  : TLSv1
    Cipher    : AES128-SHA
    Session-ID: 82CB288051A6DB66094C50A69CF1292AEE7E54C6B01B659B98AB336F8C33863E
    Session-ID-ctx:
    Master-Key: 01B025B2F764043A27919A8D1355AAECD8844FF0831B1D664042334790574A6F402
    Key-Arg   : None
```

```
    Start Time: 1394581782
    Timeout   : 300 (sec)
    Verify return code: 27 (certificate not trusted)
```

You can use these additional options with the utility:

- The `-r` flag allows you to uninstall a CA certificate.
- The `-h` flag displays more usage information.

```
ghe-ssl-ca-certificate-install -c CERTIFICATE_PATH
```

ghe-ssl-certificate-setup

This utility allows you to update an SSL certificate for your GitHub Enterprise Server instance.

For more information about this command or for additional options, use the `-h` flag.

```
ghe-ssl-certificate-setup
```

ghe-ssl-generate-csr

This utility allows you to generate a private key and certificate signing request (CSR), which you can share with a commercial or private certificate authority to get a valid certificate to use with your instance. For more information, see "Configuring TLS."

For more information about this command or for additional options, use the `-h` flag.

```
ghe-ssl-generate-csr
```

ghe-storage-extend

Some platforms require this script to expand the user volume. For more information, see "Increasing storage capacity".

```
$ ghe-storage-extend
```

ghe-version

This utility prints the version, platform, and build of your GitHub Enterprise Server instance.

```
$ ghe-version
```

ghe-webhook-logs

This utility returns webhook delivery logs for administrators to review and identify any issues.

```
ghe-webhook-logs
```

To show all failed hook deliveries in the past day:

```
ghe-webhook-logs -f -a YYYY-MM-DD
```

The date format should be `YYYY-MM-DD`, `YYYY-MM-DD HH:MM:SS`, or `YYYY-MM-DD HH:MM:SS (+/-) HH:M`.

To show the full hook payload, result, and any exceptions for the delivery:

```
ghe-webhook-logs -g DELIVERY_GUID
```

Clustering

ghe-cluster-status

Check the health of your nodes and services in a cluster deployment of GitHub Enterprise Server.

```
$ ghe-cluster-status
```

ghe-cluster-support-bundle

This utility creates a support bundle tarball containing important logs from each of the nodes in either a Geo-replication or Clustering configuration.

By default, the command creates the tarball in */tmp*, but you can also have it `cat` the tarball to `STDOUT` for easy streaming over SSH. This is helpful in the case where the web UI is unresponsive or downloading a support bundle from */setup/support* doesn't work. You must use this command if you want to generate an *extended* bundle, containing older logs. You can also use this command to upload the cluster support bundle directly to GitHub Enterprise support.

> **Note**: To use the `--p` / `--period` argument that appears in the following commands, your instance must be running the latest patch release. For more information, see Release notes.

To create a standard bundle:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-cluster-support-bundle -o' > cluster-support-bu
```

To create a standard bundle including data from the last 3 hours:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-cluster-support-bundle -p '3 hours'  -o" > suppo
```

To create a standard bundle including data from the last 2 days:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-cluster-support-bundle -p '2 days'  -o" > suppo
```

To create a standard bundle including data from the last 4 days and 8 hours:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-cluster-support-bundle -p '4 days 8 hours'  -o"
```

To create an extended bundle including data from the last 8 days:

```
$ ssh -p 122 admin@HOSTNAME -- ghe-cluster-support-bundle -x -o' > cluster-support-b
```

To send a bundle to GitHub Support:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-cluster-support-bundle -u'
```

To send a bundle to GitHub Support and associate the bundle with a ticket:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-cluster-support-bundle -t TICKET_ID'
```

ghe-dpages

This utility allows you to manage the distributed GitHub Pages server.

```
ghe-dpages
```

To show a summary of repository location and health:

```
ghe-dpages status
```

To evacuate a GitHub Pages storage service before evacuating a cluster node:

```
ghe-dpages evacuate pages-server-UUID
```

ghe-spokes

This utility allows you to manage the three copies of each repository on the distributed Git servers.

```
ghe-spokes
```

To show a summary of repository location and health:

```
ghe-spokes status
```

To show the servers in which the repository is stored:

```
ghe-spokes route
```

To evacuate storage services on a cluster node:

```
ghe-spokes server evacuate git-server-UUID
```

ghe-storage

This utility allows you to evacuate all storage services before evacuating a cluster node.

```
ghe-storage evacuate storage-server-UUID
```

Git

ghe-btop

A `top`-like interface for current Git operations.

```
ghe-btop [  | --help | --usage ]
```

ghe-governor

This utility helps to analyze Git traffic. It queries *Governor* data files, located under `/data/user/gitmon`. GitHub holds one hour of data per file, retained for two weeks. For more information, see Analyzing Git traffic using Governor in GitHub Community.

```
ghe-governor <subcommand> <column> [options]
```

```
ghe-governor -h
Usage: ghe-governor [-h] <subcommand> args

OPTIONS:
  -h | --help        Show this message.

Valid subcommands are:
  aggregate              Find the top (n) groups of queries for a grouping function
  health                 Summarize all recent activity on one or more servers
  top                    Find the top (n) queries for a given metric
  dump                   Dump individual operations
  test-quotas            Check quota information

Try ghe-governor <subcommand> --help for more information on the arguments each sub
```

ghe-repo

This utility allows you to change to a repository's directory and open an interactive shell as the `git` user. You can perform manual inspection or maintenance of a repository via commands like `git-*` or `git-nw-*`.

```
ghe-repo USERNAME/REPONAME
```

ghe-repo-gc

This utility manually repackages a repository network to optimize pack storage. If you have a large repository, running this command may help reduce its overall size. GitHub Enterprise automatically runs this command throughout your interaction with a repository network.

You can add the optional `--prune` argument to remove unreachable Git objects that aren't referenced from a branch, tag, or any other ref. This is particularly useful for immediately removing previously expunged sensitive information.

Warning: Before using the `--prune` argument to remove unreachable Git objects, put your GitHub Enterprise Server instance into maintenance mode, or ensure all repositories within the same repository network are locked. For more information, see "Enabling and scheduling maintenance mode" and "Locking a repository."

```
ghe-repo-gc USERNAME/REPONAME
```

GitHub Actions

ghe-actions-check

This utility checks that all services for GitHub Actions are healthy. For more information, see "Getting started with GitHub Actions for GitHub Enterprise Server" and "Troubleshooting GitHub Actions for your enterprise."

```
ghe-actions-check
```

ghe-actions-precheck

This utility tests the blob storage configuration for GitHub Actions on your GitHub Enterprise Server instance. You can use the utility to verify your storage configuration before you enable GitHub Actions for your instance.

For more information about the configuration of GitHub Actions, see "Getting started with GitHub Actions for GitHub Enterprise Server."

```
ghe-actions-precheck -p [PROVIDER] -cs ["CONNECTION-STRING"]
```

If your storage system is configured correctly, you'll see the following output.

```
All Storage tests passed
```

High availability

ghe-repl-promote

This command disables replication on an existing replica node and converts the replica node to a primary node using the same settings as the original primary node. All replication services are enabled. For more information, see "Initiating a failover to your replica appliance."

Promoting a replica does not automatically set up replication for existing appliances. After promoting a replica, if desired, you can set up replication from the new primary to existing appliances and the previous primary.

```
ghe-repl-promote
```

ghe-repl-setup

Run this utility on an existing node to begin enabling a high availability configuration. The utility puts the node in standby mode before you begin replication with `ghe-repl-start`. For more information, see "Creating a high availability replica."

After running the utility, the following configuration occurs on the node.

- An encrypted WireGuard VPN tunnel is established for communication between the nodes.
- Database services are configured for replication and started.
- Application services are disabled. Attempts to access the replica node over HTTP or HTTPS, Git, or other supported protocols will display "Server in replication mode" message, a maintenance page, or an error message.

When running this utility, replace PRIMARY-NODE-IP with the IP address of your instance's primary node.

```
ghe-repl-setup PRIMARY-NODE-IP
```

ghe-repl-start

This utility begins replication of all datastores on a node. Run this utility after running `ghe-repl-setup`. For more information, see "Creating a high availability replica."

```
ghe-repl-start
```

ghe-repl-status

This utility displays the status of replication on a node, returning an `OK`, `WARNING` or `CRITICAL` status for each datastore's replication stream. For more information, see "Monitoring a high-availability configuration."

- If any of the replication channels are in a `WARNING` state, the command will exit with code `1`.
- If any of the channels are in a `CRITICAL` state, the command will exit with code `2`.
- The output conforms to the expectations of Nagios' check_by_ssh plugin. For more information, see the check_by_ssh plugin on the official Nagios plugins page.

```
ghe-repl-status
```

The `-v` and `-vv` options provide additional details about each datastore's replication state.

```
ghe-repl-status -v
```

ghe-repl-stop

This command temporarily disables replication for all datastores on an existing replica node. All replication services are stopped. To resume replication, use `ghe-repl-start`.

```
ghe-repl-stop
```

ghe-repl-teardown

This utility completely disables replication on an existing replica node, removing the replica configuration. You can run the following command from a replica node, but if the replica node is unreachable, you can also run the command from the primary node.

```
ghe-repl-teardown
```

Import and export

ghe-migrator

`ghe-migrator` is a hi-fidelity tool to help you migrate from one GitHub instance to another. You can consolidate your instances or move your organization, users, teams, and repositories from GitHub.com to GitHub Enterprise.

For more information, please see our guides on migrating data to and from your enterprise.

git-import-detect

Given a URL, detect which type of source control management system is at the other end. During a manual import this is likely already known, but this can be very useful in automated scripts.

```
git-import-detect
```

git-import-hg-raw

This utility imports a Mercurial repository to this Git repository. For more information, see "Importing from other version control systems with the administrative shell."

```
git-import-hg-raw
```

git-import-svn-raw

This utility imports Subversion history and file data into a Git branch. This is a straight copy of the tree, ignoring any trunk or branch distinction. For more information, see "Importing from other version control systems with the administrative shell."

```
git-import-svn-raw
```

git-import-tfs-raw

This utility imports from Team Foundation Version Control (TFVC). For more information, see "Importing from other version control systems with the administrative shell)."

```
git-import-tfs-raw
```

git-import-rewrite

This utility rewrites the imported repository. This gives you a chance to rename authors and, for Subversion and TFVC, produces Git branches based on folders. For more information, see "Importing from other version control systems with the administrative shell."

```
git-import-rewrite
```

Security

ghe-find-insecure-git-operations

This utility searches your instance's logs and identifies Git operations over SSH that use insecure algorithms or hash functions, including DSA, RSA-SHA-1, HMAC-SHA-1, and CBC ciphers. You can use the output to support each client's transition to a more secure SSH connection. For more information, see the GitHub Blog.

```
ghe-find-insecure-git-operations
```

Support

ghe-diagnostics

This utility performs a variety of checks and gathers information about your installation that you can send to support to help diagnose problems you're having.

Currently, this utility's output is similar to downloading the diagnostics info in the Management Console, but may have additional improvements added to it over time that aren't available in the web UI. For more information, see "Providing data to GitHub Support."

```
ghe-diagnostics
```

ghe-support-bundle

> **Note:** If your GitHub Enterprise Server instance is in a geo-replication configuration, or if your instance is a cluster, you should use the `ghe-cluster-support-bundle` command to retrieve the support bundle. For more information, see "Command-line utilities."

This utility creates a support bundle tarball containing important logs from your instance. By default, the command creates the tarball in */tmp*, but you can also have it `cat` the tarball to `STDOUT` for easy streaming over SSH. This is helpful in the case where the web UI is unresponsive or downloading a support bundle from */setup/support* doesn't work. You must use this command if you want to generate an *extended* bundle, containing older logs. You can also use this command to upload the support bundle directly to GitHub Enterprise support.

> **Note**: To use the `--p` / `--period` argument that appears in the following commands, your instance must be running the latest patch release. For more information, see Release notes.

To create a standard bundle:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-support-bundle -o' > support-bundle.tgz
```

To create a standard bundle including data from the last 3 hours:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-support-bundle -p '3 hours'  -o" > support-bundl
```

To create a standard bundle including data from the last 2 days:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-support-bundle -p '2 days'  -o" > support-bundle
```

To create a standard bundle including data from the last 4 days and 8 hours:

```
$ ssh -p 122 admin@HOSTNAME -- "ghe-support-bundle -p '4 days 8 hours'  -o" > suppo
```

To create an extended bundle including data from the last 8 days:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-support-bundle -x -o' > support-bundle.tgz
```

To send a bundle to GitHub Support:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-support-bundle -u'
```

To send a bundle to GitHub Support and associate the bundle with a ticket:

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-support-bundle -t TICKET_ID'
```

ghe-support-upload

This utility sends information from your appliance to GitHub Enterprise support. You can either specify a local file, or provide a stream of up to 100MB of data via `STDIN`. The uploaded data can optionally be associated with a support ticket.

To send a file to GitHub Support and associate the file with a ticket:

```
ghe-support-upload -f FILE_PATH -t TICKET_ID
```

To upload data via `STDIN` and associating the data with a ticket:

```
ghe-repl-status -vv | ghe-support-upload -t TICKET_ID -d "Verbose Replication Status
```

In this example, `ghe-repl-status -vv` sends verbose status information from a replica appliance. You should replace `ghe-repl-status -vv` with the specific data you'd like to stream to `STDIN`, and `Verbose Replication Status` with a brief description of the data. Typically, you will only execute this if you've contacted support and they've asked you to do so.

Upgrading GitHub Enterprise Server

ghe-update-check

This utility will check to see if a new patch release of GitHub Enterprise is available. If it is, and if space is available on your instance, it will download the package. By default, it's saved to */var/lib/ghe-updates*. An administrator can then perform the upgrade.

A file containing the status of the download is available at */var/lib/ghe-updates/ghe-update-check.status*.

To check for the latest GitHub Enterprise release, use the `-i` switch.

```
$ ssh -p 122 admin@HOSTNAME -- 'ghe-update-check'
```

ghe-upgrade

This utility installs or verifies an upgrade package. You can also use this utility to roll back a patch release if an upgrade fails or is interrupted. For more information, see "Upgrading GitHub Enterprise Server."

To verify an upgrade package:

```
ghe-upgrade --verify UPGRADE-PACKAGE-FILENAME
```

To install an upgrade package:

```
ghe-upgrade UPGRADE-PACKAGE-FILENAME
```

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.

ghe-upgrade-scheduler

This utility manages scheduled installation of upgrade packages. You can show, create new, or remove scheduled installations. You must create schedules using cron expressions. For more information, see the Cron Wikipedia entry.

The `ghe-upgrade-scheduler` utility is best suited for scheduling hotpatch upgrades, which do not require maintenance mode or a reboot in most cases. This utility is not practical for full package upgrades, which require an administrator to manually set maintenance mode, reboot the instance, and unset maintenance mode. For more information about the different types of upgrades, see "Upgrading GitHub Enterprise Server"

To schedule a new installation for a package:

```
$ ghe-upgrade-scheduler -c "0 2 15 12 *" UPGRADE-PACKAGE-FILENAME
```

To show scheduled installations for a package:

```
$ ghe-upgrade-scheduler -s UPGRADE PACKAGE FILENAME
> 0 2 15 12 * /usr/local/bin/ghe-upgrade -y -s UPGRADE-PACKAGE-FILENAME > /data/use
```

To remove scheduled installations for a package:

```
$ ghe-upgrade-scheduler -r UPGRADE PACKAGE FILENAME
```

User management

ghe-license-usage

This utility exports a list of the installation's users in JSON format. If your instance is connected to GitHub Enterprise Cloud, GitHub Enterprise Server uses this information for reporting licensing information to GitHub Enterprise Cloud. For more information, see "Managing GitHub Connect."

By default, the list of users in the resulting JSON file is encrypted. Use the `-h` flag for more options.

```
ghe-license-usage
```

ghe-org-membership-update

This utility will enforce the default organization membership visibility setting on all members in your instance. For more information, see "Configuring visibility for organization membership." Setting options are `public` or `private`.

```
ghe-org-membership-update --visibility=SETTING
```

`ghe-user-csv`

This utility exports a list of all the users in the installation into CSV format. The CSV file includes the email address, which type of user they are (e.g., admin, user), how many repositories they have, how many SSH keys, how many organization memberships, last logged IP address, etc. Use the `-h` flag for more options.

```
ghe-user-csv -o > users.csv
```

ghe-user-demote

This utility demotes the specified user from admin status to that of a regular user. We recommend using the web UI to perform this action, but provide this utility in case the `ghe-user-promote` utility is run in error and you need to demote a user again from the CLI.

```
ghe-user-demote USERNAME
```

ghe-user-promote

This utility promotes the specified user account to a site administrator.

```
ghe-user-promote USERNAME
```

ghe-user-suspend

This utility suspends the specified user, preventing them from logging in, pushing, or pulling from your repositories.

```
ghe-user-suspend USERNAME
```

ghe-user-unsuspend

This utility unsuspends the specified user, granting them access to login, push, and pull

from your repositories.

```
ghe-user-unsuspend USERNAME
```