Exporting migration data from GitHub Enterprise Server

In this article

To change platforms or move from a trial instance to a production instance, you can export migration data from a GitHub Enterprise Server instance by preparing the instance, locking the repositories, and generating a migration archive.

Preparing the GitHub Enterprise Server source instance

1. Verify that you are a site administrator on the GitHub Enterprise Server source. The best way to do this is to verify that you can SSH into the instance.
2. Generate an access token with the `repo` and `admin:org` scopes on the GitHub Enterprise Server source instance.
3. To minimize downtime, make a list of repositories you want to export from the source instance. You can add multiple repositories to an export at once using a text file that lists the URL of each repository on a separate line.

Exporting the GitHub Enterprise Server source repositories

Note: Locking a repository prevents all write access to the repository. You cannot associate new teams or collaborators with a locked repository.

If you're performing a trial run, you do not need to lock the repository. When you migrate data from a repository that's in use, GitHub strongly recommends locking the repository. For more information, see "About ghe-migrator."

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. To prepare a repository for export, use the `ghe-migrator add` command with the repository's URL:
   - If you're locking the repository, append the command with `--lock`. If you're performing a trial run, `--lock` is not needed.

     ```
     $ ghe-migrator add https://HOSTNAME/USERNAME/REPO-NAME --lock
     ```

   - You can exclude file attachments by appending `--exclude_attachments` to the command. File attachments can be large and may needlessly bloat your final migration archive.
   - To prepare multiple repositories at once for export, create a text file listing each repository URL on a separate line, and run the `ghe-migrator add` command with the `-i` flag and the path to your text file.

     ```
     $ ghe-migrator add -i PATH/TO/YOUR/REPOSITORY_URL.txt
     ```

3. When prompted, enter your GitHub Enterprise Server username:

   ```
   Enter username authorized for migration:  admin
   ```

4. When prompted for a personal access token, enter the access token you created in "Preparing the GitHub Enterprise Server source instance":

   ```
   Enter personal access token:  **************
   ```

5. When `ghe-migrator add` has finished it will print the unique "Migration GUID" that it generated to identify this export as well as a list of the resources that were added to the export. You will use the Migration GUID that it generated in subsequent `ghe-migrator add` and `ghe-migrator export` steps to tell `ghe-migrator` to continue operating on the same export.

   ```
   > 101 models added to export
   > Migration GUID: EXAMPLE-MIGRATION-GUID
   > Number of records in this migration:
   > users                         |  5
   > organizations                 |  1
   > repositories                  |  1
   > teams                         |  3
   > protected_branches            |  1
   > pull_request_reviews          |  1
   > milestones                    |  1
   > issues                        |  3
   > pull_requests                 |  5
   > pull_request_review_comments  |  4
   > commit_comments               |  2
   > issue_comments                | 10
   > issue_events                  | 63
   > releases                      |  3
   > attachments                   |  4
   > projects                      |  2
   ```

   Each time you add a new repository with an existing Migration GUID it will update the existing export. If you run `ghe-migrator add` again without a Migration GUID it will start a new export and generate a new Migration GUID. **Do not re-use the**

Migration GUID generated during an export when you start preparing your migration for import.

6. To add more repositories to the same export, use the `ghe-migrator add` command with the `-g` flag. You'll pass in the new repository URL and the Migration GUID from Step 5:

```
$ ghe-migrator add https://HOSTNAME/USERNAME/OTHER-REPO-NAME -g MIGRATION-GUID
```

7. When you've finished adding repositories, generate the migration archive using the `ghe-migrator export` command with the `-g` flag and the Migration GUID from Step 5:

```
$ ghe-migrator export -g MIGRATION-GUID
> Archive saved to: /data/github/current/tmp/MIGRATION-GUID.tar.gz
```

   - To specify where migration files should be staged append the command with `--staging-path=/full/staging/path`. Defaults to `/data/user/tmp`.

8. Close the connection to your GitHub Enterprise Server instance:

```
$ exit
> logout
> Connection to HOSTNAME closed.
```

9. Copy the migration archive to your computer using the `scp` command. The archive file will be named with the Migration GUID:

```
$ scp -P 122 admin@HOSTNAME:/data/github/current/tmp/MIGRATION-GUID.tar.gz ~/De
```

10. To prepare the archived migration data for import into a GitHub Enterprise Server instance, see "Preparing to migrate data to GitHub Enterprise Server".