Migrating repositories from Bitbucket Server to GitHub Enterprise Cloud

In this article

About repository migrations with GitHub Enterprise Importer

Prerequisites

Step 1: Install the BBS2GH extension of the GitHub CLI

Step 2: Update the BBS2GH extension of the GitHub CLI

Step 3: Set environment variables

Step 4: Set up blob storage

Step 5: Migrate a repository

Step 6: Validate your migration and check the error log

Step 7: Migrate multiple repositories

You can migrate repositories from Bitbucket Server to GitHub Enterprise Cloud using the GitHub CLI.

About repository migrations with GitHub Enterprise Importer

You can migrate individual repositories or all repositories from a BitBucket Server instance using GitHub CLI.

At this time, migrating from Bitbucket Server with the GitHub API is not supported.

Prerequisites

- To ensure you understand the known support limitations of the Importer, review "About GitHub Enterprise Importer."
- We strongly recommend that you perform a trial run of your migration and complete your production migration soon after. To learn more about trial run best practices, see "Preparing to run a migration with GitHub Enterprise Importer."
- While not required, we recommend halting work during your production migration. The Importer doesn't support delta migrations, so any changes that happen during the migration will not migrate. If you choose not to halt work during your production migration, you'll need to manually migrate these changes.
- For the destination organization on GitHub.com, you must be an organization owner or have the migrator role. For more information, see "Granting the migrator role for GitHub Enterprise Importer."
- You need the username and password for a Bitbucket Server account with admin or super admin permissions.

Step 1: Install the BBS2GH extension of the GitHub CLI

If this is your first migration, you'll need to install the BBS2GH extension of the GitHub CLI. For more information about GitHub CLI, see "About GitHub CLI."

1. Install the GitHub CLI. For installation instructions for GitHub CLI, see the GitHub CLI repository.

   > **Note:** You need version 2.4.0 or newer of GitHub CLI. You can check the version you have installed with the `gh --version` command.

2. Install the BBS2GH extension.

   ```
   Shell
   gh extension install github/gh-bbs2gh
   ```

Any time you need help with the BBS2GH extension, you can use the `--help` flag with a command. For example, `gh bbs2gh --help` will list all the available commands, and `gh bbs2gh migrate-repo --help` will list all the options available for the `migrate-repo` command.

Step 2: Update the BBS2GH extension of the GitHub CLI

The BBS2GH extension of the GitHub CLI is updated weekly. To make sure you're using the latest version, update the extension.

```
Shell
gh extension upgrade github/gh-bbs2gh
```

Step 3: Set environment variables

Before you can use the BBS2GH extension to migrate to GitHub Enterprise Cloud, you must create a personal access token that can access the destination organization, then set the personal access token as an environment variable.

You'll also need to set environment variables for your Bitbucket Server username and password and, if your Bitbucket Server instance runs on Windows, your SMB password.

1. Create and record a personal access token (classic) that will authenticate for the destination organization on GitHub Enterprise Cloud, making sure that the token meets all requirements. For more information, see "Managing access for GitHub Enterprise Importer."

2. Set environment variables, replacing TOKEN with the personal access token you recorded above, USERNAME with the username of a Bitbucket Server account that has admin or super admin permissions, and PASSWORD with the password for the Bitbucket Server account.

   - If you're using Terminal, use the `export` command.

     ```
     Shell
     ```

```
export GH_PAT="TOKEN"
export BBS_USERNAME="USERNAME"
export BBS_PASSWORD="PASSWORD"
# If your Bitbucket Server instance runs on Windows
export SMB_PASSWORD="PASSWORD"
```

- If you're using PowerShell, use the `$env` command.

Shell

```
$env:GH_PAT="TOKEN"
$env:BBS_USERNAME="USERNAME"
$env:BBS_PASSWORD="PASSWORD"
# If your Bitbucket Server instance runs on Windows
$env:SMB_PASSWORD="PASSWORD"
```

Step 4: Set up blob storage

Because many Bitbucket Server instances sit behind firewalls, the GitHub CLI uses blob storage as an intermediate location to store your data that is reachable from the internet.

You will first generate an archive of the data you want to migrate and push the data to blob storage from behind your firewall.

The GitHub CLI supports the following blob storage providers:

- Amazon Web Services (AWS) S3
- Azure Blob Storage

Before you can run a migration, you need to set up a storage container with your chosen cloud provider to store your data.

Setting up an AWS S3 storage bucket

In AWS, set up a S3 bucket. For more information, see Creating a bucket in the AWS documentation.

You will also need an AWS access key and secret key with the following permissions:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucketMultipartUploads",
                "s3:AbortMultipartUpload",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:s3:::github-migration-bucket",
                "arn:aws:s3:::github-migration-bucket/*"
            ]
        }
    ]
```

```
}
```

> **Note:** GitHub Enterprise Importer does not delete your archive from AWS after your migration is finished. To reduce storage costs, we recommend configuring auto-deletion of your archive after a period of time. For more information, see Setting lifecycle configuration on a bucket in the AWS documentation.

When you're ready to run your migration, you will need to provide your AWS credentials to the GitHub CLI: region, access key, secret key, and session token (if required). You can pass them as arguments, or set environment variables called `AWS_REGION`, `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, and `AWS_SESSION_TOKEN`.

You will also need to pass in the name of the S3 bucket using the `--aws-bucket-name` argument.

Setting up an Azure Blob Storage storage account

In Azure, create a storage account and make a note of your connection string. For more information, see Manage storage account access keys in Microsoft Docs.

> **Note:** GitHub Enterprise Importer does not delete your archive from Azure Blob Storage after your migration is finished. To reduce storage costs, we recommend configuring auto-deletion of your archive after a period of time. For more information, see Optimize costs by automatically managing the data lifecycle in Microsoft Docs.

When you're ready to run your migration, you can either pass your connection string into the GitHub CLI as an argument, or pass it in using an environment variable called `AZURE_STORAGE_CONNECTION_STRING`.

Step 5: Migrate a repository

You can migrate repositories with the `gh bbs2gh migrate-repo` command.

When you migrate a repository, by default, the BBS2GH extension of the GitHub CLI performs the following steps:

1. Connects to your Bitbucket Server instance and generates a migration archive per repository
2. Downloads the migration archive from the Bitbucket Server instance to the machine where you're running the BBS2GH extension of the GitHub CLI, using SFTP (Linux) or SMB (Windows)
3. Uploads the migration archives to the blob storage provider of your choice
4. Starts your migration in GitHub Enterprise Cloud, using the URLs of the archives stored with your blob storage provider
5. Deletes the migration archive from your local machine. (You'll need to delete the archive from your blob storage provider manually once the migration has finished.)

Alternatively, you can use the GitHub CLI to generate the archive, download that archive manually, and then use the GitHub CLI to continue the migration.

- "Allowing the GitHub CLI to download the migration archive"
- "Downloading the migration archive manually"

Allowing the GitHub CLI to download the migration archive

To migrate a single repository, use the `gh bbs2gh migrate-repo` command.

You must follow this step from a computer that can access:

- Your Bitbucket Server instance via HTTPS
- Your Bitbucket Server instance via SFTP, if your Bitbucket Server instance runs on Linux. In general, if you can access the server via SSH, then you can also use SFTP.
- Your Bitbucket Server instance via SMB, if your Bitbucket Server instance runs on Windows
- Your chosen blob storage provider

Shell

```
gh bbs2gh migrate-repo --bbs-server-url BBS-SERVER-URL \
  --bbs-project PROJECT --bbs-repo CURRENT-NAME \
  --github-org DESTINATION --github-repo NEW-NAME \
  # Use the following options if your Bitbucket Server instance runs on Linux
  --ssh-user SSH-USER --ssh-private-key PATH-TO-KEY
  # Use the following options if your Bitbucket Server instance runs on Windows
  --smb-user SMB-USER
  # Use the following option if you're using AWS S3 as your blob storage provider
  --aws-bucket-name AWS-BUCKET-NAME
  # Use the following option if you are running a Bitbucket Data Center cluster
or your Bitbucket Server is behind a load balancer
  --archive-download-host ARCHIVE-DOWNLOAD-HOST
```

Replace the placeholders in the command above with the following values.

Placeholder	Value
BBS-SERVER-URL	The URL for your Bitbucket Server instance
PROJECT	The key for the Bitbucket Server project of the repository you want to migrate
CURRENT-NAME	The name of the repository you want to migrate
DESTINATION	Name of the destination organization
NEW-NAME	The name you want the migrated repository to have
SSH-USER	If your Bitbucket Server instance runs on Linux, the username to use when connecting to your Bitbucket Server via SFTP
PATH-TO-KEY	If your Bitbucket Server instance runs on Linux, the path to your SSH private key, such as `~/.ssh/id_rsa`. For SSH key requirements, see "Managing access for GitHub Enterprise Importer".
SMB-USER	If your Bitbucket Server instance runs on Windows, the username to use when connecting to your Bitbucket Server via SMB
AWS-BUCKET-NAME	The bucket name for your AWS S3 bucket
ARCHIVE-DOWNLOAD-HOST	The host to use to connect to the Bitbucket Server/Data Center instance via SSH or SMB. You only need to specify this if you are running a

You only need to specify this if you are running a [Bitbucket Data Center cluster] or your Bitbucket Server is behind a load balancer.

Note: If you get an error mentioning `Renci.SshNet`, then the CLI is having issues making an SFTP connection to your server to download your migration archive. For information about how to troubleshoot these issues, see "Troubleshooting your migration with GitHub Enterprise Importer."

Downloading the migration archive manually

By default, the BBS2GH extension of the GitHub CLI performs the entire migration, including downloading the migration archive from the Bitbucket Server instance using SFTP or SMB.

However, some customers prefer to download the migration archive manually, because their server does not offer SFTP access, for example. In that case, you can use the GitHub CLI to generate the archive, download that archive manually, and then use the GitHub CLI to continue the migration.

You must follow this step from a computer that can access:

- Your Bitbucket Server instance via HTTPS
- Your chosen blob storage provider

First, use the `gh bbs2gh migrate-repo` command with only the following arguments:

Shell

```
gh bbs2gh migrate-repo --bbs-server-url BBS-SERVER-URL \
  --bbs-project PROJECT \
  --bbs-repo CURRENT-NAME
```

Replace the placeholders in the command above with the following values.

Placeholder	Value
BBS-SERVER-URL	The URL for your Bitbucket Server instance
PROJECT	The key for the Bitbucket Server project of the repository you want to migrate
CURRENT-NAME	The name of the repository you want to migrate

Your migration archive will be generated, and its path will be printed in the command output:

```
[12:14] [INFO] Export completed. Your migration archive should be ready on your
instance at $BITBUCKET_SHARED_HOME/data/migration/export/Bitbucket_export_9.tar
```

In general, `$BITBUCKET_SHARED_HOME` will be set to `/var/atlassian/application-data/bitbucket/shared` on Linux and `C:\Atlassian\ApplicationData\Bitbucket\Shared` on Windows, but this may differ depending on your server configuration. To help you identify your shared home directory, see "Troubleshooting your migration with GitHub Enterprise Importer."

Download the migration archive from your Bitbucket Server instance, and store the archive on the machine where you're running the GitHub CLI.

To import your migration archive into GitHub, use the `gh bbs2gh migrate-repo` command

again, with a different set of arguments:

```shell
gh bbs2gh migrate-repo --archive-path ARCHIVE-PATH \
  --github-org DESTINATION --github-repo NEW-NAME \
  --bbs-server-url BBS-SERVER-URL \
  --bbs-project PROJECT \
  --bbs-repo CURRENT-NAME \
  # Use the following option if you're using AWS S3 as your blob storage provider
  --aws-bucket-name AWS-BUCKET-NAME
```

Replace the placeholders in the command above with the following values.

Placeholder	Value
ARCHIVE-PATH	The path to the Bitbucket Server migration archive you downloaded from your instance
DESTINATION	Name of the destination organization
NEW-NAME	The name you want the migrated repository to have
BBS-SERVER-URL	The URL for your Bitbucket Server instance
PROJECT	The key for the Bitbucket Server project of the repository you want to migrate
CURRENT-NAME	The name of the repository you want to migrate
AWS-BUCKET-NAME	The bucket name for your AWS S3 bucket

Step 6: Validate your migration and check the error log

When your migration is complete, we recommend reviewing your migration log. For more information, see "Accessing your migration logs for GitHub Enterprise Importer."

We recommend that you review your migrated repositories for a soundness check.

Step 7: Migrate multiple repositories

If you want to migrate multiple repositories to GitHub Enterprise Cloud at once, use the GitHub CLI to generate a migration script. The resulting script will contain a list of migration commands, one per repository.

> **Note:** Generating a script outputs a PowerShell script. If you're using Terminal, you will need to output the script with the `.ps1` file extension and install PowerShell for either Mac or Linux to run it.

Generating a migration script

You must follow this step from a computer that can access your Bitbucket Server instance via HTTPS.

To generate a migration script, run the `gh bbs2gh generate-script` command.

Shell

```
gh bbs2gh generate-script --bbs-server-url BBS-SERVER-URL \
  --github-org DESTINATION \
  --output FILENAME \
  # Use the following options if your Bitbucket Server instance runs on Linux
  --ssh-user SSH-USER --ssh-private-key PATH-TO-KEY
  # Use the following options if your Bitbucket Server instance runs on Windows
  --smb-user SMB-USER
  # Use the following option if you are running a Bitbucket Data Center cluster
or your Bitbucket Server is behind a load balancer
  --archive-download-host ARCHIVE-DOWNLOAD-HOST
```

If you want the script to download the migration log for each migrated repository, add the `--download-migration-logs` flag. For more information about migration logs, see "Accessing your migration logs for GitHub Enterprise Importer."

Replace the placeholders in the command above with the following values.

Placeholder	Value
BBS-SERVER-URL	The URL for your Bitbucket Server instance
DESTINATION	Name of the destination organization
FILENAME	A filename for the resulting migration script If you're using Terminal, use a `.ps1` file extension as the generated script requires PowerShell to run. You can install PowerShell for Mac or Linux.
SSH-USER	If your Bitbucket Server instance runs on Linux, the username to use when connecting to your Bitbucket Server via SFTP
PATH-TO-KEY	If your Bitbucket Server instance runs on Linux, the path to your SSH private key, such as `~/.ssh/id_rsa`. For SSH key requirements, see "Managing access for GitHub Enterprise Importer".
SMB-USER	If your Bitbucket Server instance runs on Windows, the username to use when connecting to your Bitbucket Server via SMB
ARCHIVE-DOWNLOAD-HOST	The host to use to connect to the Bitbucket Server/Data Center instance via SSH or SMB. You only need to specify this if you are running a Bitbucket Data Center cluster or your Bitbucket Server is behind a load balancer.

Reviewing the migration script

After you generate the script, review the file and, optionally, edit the script.

- If there are any repositories you don't want to migrate, delete or comment out the corresponding lines.
- By default, repository names in GitHub will follow a `projectKey-repositoryName` convention. For example, a Bitbucket Server repository named `airports` that is part of the `open-source` project, which has the key `OS`, would be called `OS-airports` in GitHub. If you want any repositories to have a different name on GitHub, update the

value for the corresponding `--github-repo` flag.

Running your migration script

To migrate your repositories, run the generated script.

You must follow this step from a computer that can access:

- Your Bitbucket Server instance via HTTPS
- Your Bitbucket Server instance via SFTP, if your Bitbucket Server instance runs on Linux. In general, if you can access the server via SSH, then you can also use SFTP.
- Your Bitbucket Server instance via SMB, if your Bitbucket Server instance runs on Windows
- Your chosen blob storage provider

Before running the script, you must set additional environment variables to authenticate to your blob storage provider.

- For AWS S3, set the following environment variables.
  - `AWS_ACCESS_KEY`: The access key for your bucket
  - `AWS_SECRET_KEY`: The secret key for your bucket
  - `AWS_REGION`: The AWS region where your bucket is located
  - `AWS_SESSION_TOKEN`: The session token, if you're using AWS temporary credentials (see Using temporary credentials with AWS resources in the AWS documentation)
- For Azure Blob Storage, set `AZURE_STORAGE_CONNECTION_STRING` to the connection string for your Azure storage account.

Only connection strings using storage account access keys are supported. Connection strings which use shared access signatures (SAS) are not supported. For more information about storage account access keys, see Manage storage account access keys in the Azure documentation.

To migrate multiple repositories, run the script you generated above. Replace FILENAME in the commands below with the filename you provided when generating the script.

- If you're using Terminal, use `./`.

```shell
./FILENAME
```

- If you're using PowerShell, use `.\`.

```shell
.\FILENAME
```