GitHub

SSH Keys and Personal Tokens

Table of Contents

[SSH Keys 3](#_Toc32866610)

[Check for existing SSH keys 3](#_Toc32866611)

[Generate a new SSH key and add it to the ssh-agent 3](#_Toc32866612)

[Generating a new SSH key 4](#_Toc32866613)

[Adding your SSH key to the ssh-agent 4](#_Toc32866614)

[Adding a new SSH key to your GitHub account 5](#_Toc32866615)

[Changing a remote’s URL 7](#_Toc32866616)

[Switching remote URLs from SSH to HTTPS 7](#_Toc32866617)

[Switching remote URLs from HTTPS to SSH 8](#_Toc32866618)

[Troubleshooting 8](#_Toc32866619)

[Personal Access Tokens 9](#_Toc32866620)

[Creating a Token 9](#_Toc32866621)

[Using a Token on the Command Line 12](#_Toc32866622)

# SSH Keys

*\*Note: Windows instructions assume use of GitHub Desktop, which comes with Git Bash tool.*

## Check for existing SSH keys

Before you generate an SSH key, you can check to see if you have any existing SSH keys.

1. Open Terminal (Mac/Linux). Open Git Bash (Windows).
2. Enter ls -al ~/.ssh to see if existing SSH keys are present:

$ ls -al ~/.ssh

# Lists the files in your .ssh directory, if they exist

1. Check the directory listing to see if you already have a public SSH key. By default, the filenames of the public keys are one of the following:

id\_rsa.pub

id\_ecdsa.pub

id\_ed25519.pub

id\_dsa.pub

If you don't have an existing public and private key pair, or don't wish to use any that are available to connect to GitHub Enterprise, then [generate a new SSH key](#_Generate_a_new).

If you see an existing public and private key pair listed (for example id\_rsa.pub and id\_rsa) that you would like to use to connect to GitHub Enterprise, you can [add your SSH key to the ssh-agent](#_Generate_a_new).

**Tip:** If you receive an error that ~/.ssh doesn't exist, don't worry! We'll create it when we [generate a new SSH key](#_Generate_a_new).

## Generate a new SSH key and add it to the ssh-agent

After you've checked for existing SSH keys, you can generate a new SSH key to use for authentication, then add it to the ssh-agent.

If you don't already have an SSH key, you must [generate a new SSH key](#_Generating_a_new). If you're unsure whether you already have an SSH key, [check for existing keys](#_Check_for_existing).

If you don't want to reenter your passphrase every time you use your SSH key, you can [add your key to the SSH agent](#_Adding_your_SSH), which manages your SSH keys and remembers your passphrase.

### Generating a new SSH key

1. Open Terminal (Mac/Linux). Open Git Bash (Windows).
2. Paste the text below, substituting in your GitHub Enterprise email address.

$ ssh-keygen -t rsa -b 4096 -C "your\_email@example.com"

This creates a new ssh key, using the provided email as a label.

* + Generating public/private rsa key pair.

1. When you're prompted to "Enter a file in which to save the key," press Enter. This accepts the default file location.
   * Enter a file in which to save the key (/Users/you/.ssh/id\_rsa): [Press enter]
2. At the prompt, type a secure passphrase.
   * Enter passphrase (empty for no passphrase): [Type a passphrase]
   * Enter same passphrase again: [Type passphrase again]

### Adding your SSH key to the ssh-agent

Before adding a new SSH key to the ssh-agent to manage your keys, you should have [checked for existing SSH keys](#_Check_for_existing) and [generated a new SSH key](#_Generating_a_new).

1. Start the ssh-agent in the background.

$ eval "$(ssh-agent -s)"

* + Agent pid 59566

1. If you are using macOS Sierra 10.12.2 or later: you will need to modify your ~/.ssh/config file to automatically load keys into the ssh-agent and store passphrases in your keychain.

Host \*

AddKeysToAgent yes

UseKeychain yes

IdentityFile ~/.ssh/id\_rsa

1. Add your SSH private key to the ssh-agent and store your passphrase in the keychain. If you created your key with a different name, or if you are adding an existing key that has a different name, replace id\_rsa in the command with the name of your private key file.

$ ssh-add -K ~/.ssh/id\_rsa

1. [Add the SSH key to your GitHub account](#_Adding_a_new).

## Adding a new SSH key to your GitHub account

Before adding a new SSH key to your GitHub Enterprise account, you should have:

[Checked for existing SSH keys](#_Check_for_existing)

[Generated a new SSH key and added it to the ssh-agent](#_Generate_a_new)

After adding a new SSH key to your GitHub Enterprise account, you can reconfigure any local repositories to use SSH. For more information, see "[Switching remote URLs from HTTPS to SSH](#_Switch_remote_URLs)."

1. Copy the SSH key to your clipboard.

If your SSH key file has a different name than the example code, modify the filename to match your current setup. When copying your key, don't add any newlines or whitespace.

* 1. Mac:

$ pbcopy < ~/.ssh/id\_rsa.pub

# Copies the contents of the id\_rsa.pub file to your clipboard

* 1. Windows:

$ clip < ~/.ssh/id\_rsa.pub

# Copies the contents of the id\_rsa.pub file to your clipboard

* 1. Linux:

$ sudo apt-get install xclip

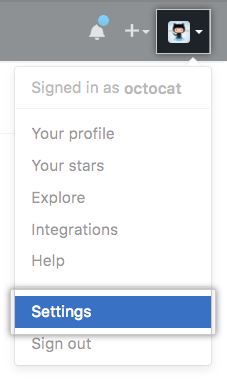
# Downloads and installs xclip. If you don't have `apt-get`, you might need to use another installer (like `yum`)

$ xclip -sel clip < ~/.ssh/id\_rsa.pub

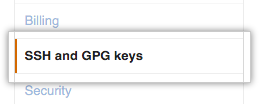
# Copies the contents of the id\_rsa.pub file to your clipboard

**Tip:** If pbcopy/clip/xclip isn't working, you can locate the hidden .ssh folder, open the file in your favorite text editor, and copy it to your clipboard.

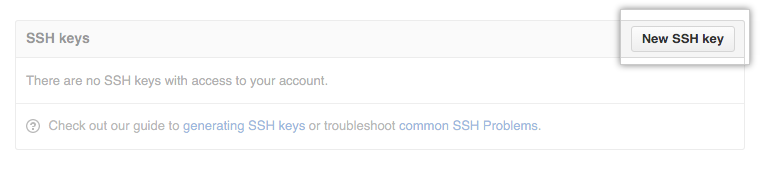
1. In the upper-right corner of any page, click your profile photo, then click **Settings**.



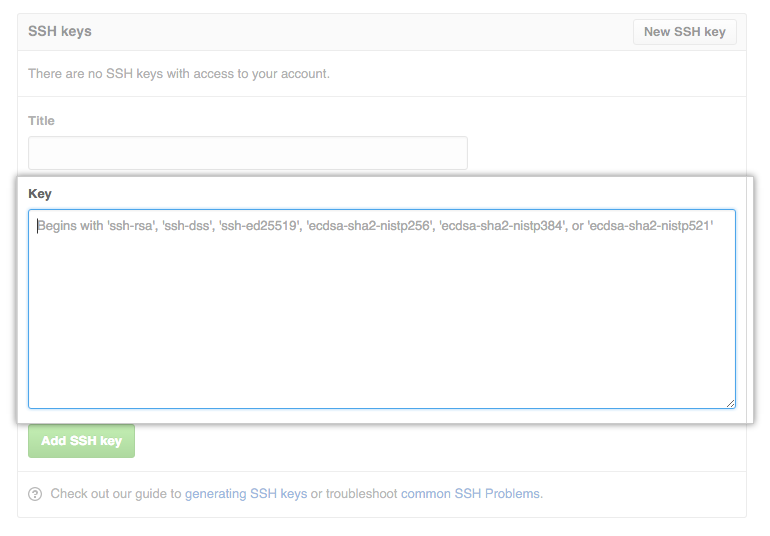
1. In the user settings sidebar, click **SSH and GPG keys**.



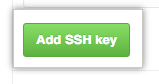
1. Click **New SSH key** or **Add SSH key**.



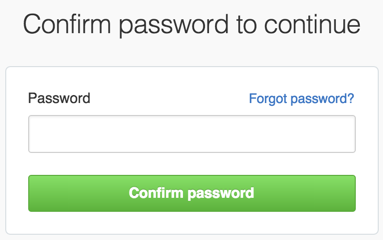
1. In the "Title" field, add a descriptive label for the new key. For example, if you're using a personal Mac, you might call this key "Personal MacBook Air".
2. Paste your key into the "Key" field.



1. Click **Add SSH key**.



1. If prompted, confirm your GitHub Enterprise password.



## Changing a remote’s URL

The git remote set-url command changes an existing remote repository URL.

The git remote set-url command takes two arguments:

* An existing remote name. For example, origin or upstream are two common choices.
* A new URL for the remote. For example:
  + If you're updating to use HTTPS, your URL might look like:

https://[hostname]/USERNAME/REPOSITORY.git

* + If you're updating to use SSH, your URL might look like:

git@hostname:USERNAME/REPOSITORY.git

### Switching remote URLs from SSH to HTTPS

1. Open Terminal (Mac/Linux). Open Git Bash (Windows).
2. Change the current working directory to your local project.
3. List your existing remotes in order to get the name of the remote you want to change.

$ git remote -v

* + origin git@hostname:USERNAME/REPOSITORY.git (fetch)
  + origin git@hostname:USERNAME/REPOSITORY.git (push)

1. Change your remote's URL from SSH to HTTPS with the git remote set-url command.

$ git remote set-url origin https://hostname/USERNAME/REPOSITORY.git

1. Verify that the remote URL has changed.

$ git remote -v

# Verify new remote URL

* + origin https://hostname/USERNAME/REPOSITORY.git (fetch)
  + origin https://hostname/USERNAME/REPOSITORY.git (push)

The next time you git fetch, git pull, or git push to the remote repository, you'll be asked for your GitHub username and password.

* If you have [two-factor authentication](https://help.github.com/en/enterprise/2.18/user/articles/securing-your-account-with-two-factor-authentication-2fa) enabled, you must [create a personal access token](https://help.github.com/en/enterprise/2.18/user/articles/creating-a-personal-access-token-for-the-command-line) to use instead of your GitHub password.
* You can [use a credential helper](https://help.github.com/en/enterprise/2.18/user/articles/caching-your-github-password-in-git) so Git will remember your GitHub username and password every time it talks to GitHub.

### [Switching remote URLs from HTTPS to SSH](https://help.github.com/en/enterprise/2.18/user/github/using-git/changing-a-remotes-url#switching-remote-urls-from-https-to-ssh)

1. Open Terminal (Mac/Linux). Open Git Bash (Windows).
2. Change the current working directory to your local project.
3. List your existing remotes in order to get the name of the remote you want to change.

$ git remote -v

* + origin https://hostname/USERNAME/REPOSITORY.git (fetch)
  + origin https://hostname/USERNAME/REPOSITORY.git (push)

1. Change your remote's URL from HTTPS to SSH with the git remote set-url command.
   * $ git remote set-url origin git@hostname:USERNAME/REPOSITORY.git
2. Verify that the remote URL has changed.

$ git remote -v

# Verify new remote URL

* + origin git@hostname:USERNAME/REPOSITORY.git (fetch)
  + origin git@hostname:USERNAME/REPOSITORY.git (push)

### [Troubleshooting](https://help.github.com/en/enterprise/2.18/user/github/using-git/changing-a-remotes-url#troubleshooting)

You may encounter these errors when trying to change a remote.

[**No such remote '[name]'**](https://help.github.com/en/enterprise/2.18/user/github/using-git/changing-a-remotes-url#no-such-remote-name)

This error means that the remote you tried to change doesn't exist:

$ git remote set-url sofake <https://hostname/octocat/Spoon-Knife>

* + fatal: No such remote 'sofake'

Check that you've correctly typed the remote name.

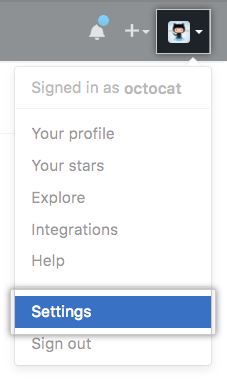
# Personal Access Tokens

You can create a personal access token and use it in place of a password when performing Git operations over HTTPS with Git on the command line or the API.

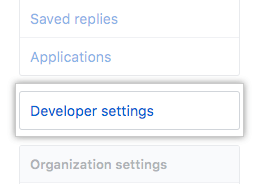
A personal access token is required to authenticate to GitHub Enterprise when [2FA](https://help.github.com/en/enterprise/2.18/user/articles/about-two-factor-authentication) is enabled on your account.

## Creating a Token

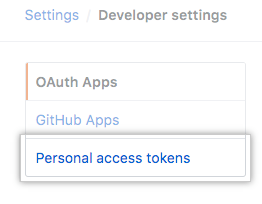
In the upper-right corner of any page, click your profile photo, then click **Settings**.



In the left sidebar, click **Developer settings**.



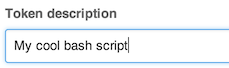
In the left sidebar, click **Personal access tokens**.



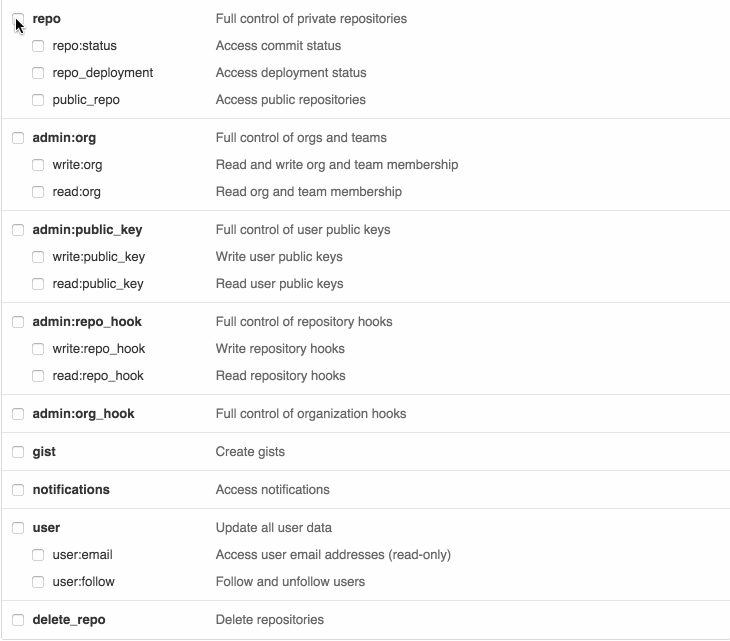
Click **Generate new token**.



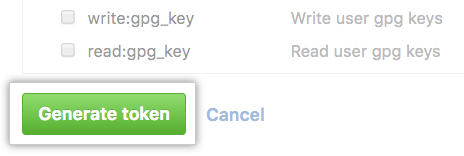
Give your token a descriptive name.



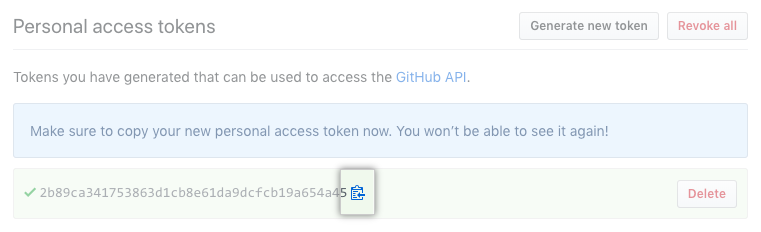
Select the scopes, or permissions, you'd like to grant this token. To use your token to access repositories from the command line, select **repo**.



Click **Generate token**.



Click  to copy the token to your clipboard. For security reasons, after you navigate off the page, you will not be able to see the token again.



**Warning:** Treat your tokens like passwords and keep them secret. When working with the API, use tokens as environment variables instead of hardcoding them into your programs.

## Using a Token on the Command Line

Once you have a token, you can enter it instead of your password when performing Git operations over HTTPS.

For example, on the command line you would enter the following:

$ git clone https://hostname/username/repo.git

Username: your\_username

Password: your\_token

Personal access tokens can only be used for HTTPS Git operations. If your repository uses an SSH remote URL, you will need to [switch the remote from SSH to HTTPS](#_Switching_remote_URLs).