github:help

- Contact Support
- Back to GitHub

SSH / Generating SSH Keys

How can we help? Search

Generating SSH Keys

- mac
- windows
- linux
- all

If you have decided not to use the recommended HTTPS method, we can use SSH keys to establish a secure connection between your computer and GitHub. The steps below will walk you through generating an SSH key and then adding the public key to your GitHub account.

Step 1: Check for SSH keys

*Have an existing keypair you'd like to use? You can skip to **Step 4**.*

First, we need to check for existing ssh keys on your computer. Open up TerminalTerminalGit Bashthe command line and run:

```
cd ~/.ssh# Checks to see if there is a directory named ".ssh" in your user directory
```

If it says "No such file or directory" skip to **step 3**. Otherwise continue to **step 2**.

Step 2: Backup and remove existing SSH keys

Since there is already an SSH directory you'll want to back the old one up and remove it:

```
ls# Lists all the subdirectories in the current directory
# config  id_rsa  id_rsa.pub  known_hosts

mkdir key_backup# Makes a subdirectory called "key_backup" in the current directory

cp id_rsa* key_backup# Copies the id_rsa keypair into key_backup

rm id_rsa*# Deletes the id_rsa keypair
```

Step 3: Generate a new SSH key

To generate a new SSH key, enter the code below. We want the default settings so when asked to enter a file in which to save the key, just press enter.

```
ssh-keygen -t rsa -C "your_email@youremail.com"# Creates a new ssh key using the provided email
# Generating public/private rsa key pair.
# Enter file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]
```

```
ssh-keygen -t rsa -C "your_email@youremail.com"# Creates a new ssh key using the provided email
# Generating public/private rsa key pair.
# Enter file in which to save the key (/c/Users/you/.ssh/id_rsa): [Press enter]
```

```
ssh-keygen -t rsa -C "your_email@youremail.com"# Creates a new ssh key using the provided email
# Generating public/private rsa key pair.
# Enter file in which to save the key (/home/you/.ssh/id_rsa):
```

```
ssh-keygen -t rsa -C "your_email@youremail.com"# Creates a new ssh key using the provided email
# Generating public/private rsa key pair.
# Enter file in which to save the key (/your_home_path/.ssh/id_rsa):
```

Now you need to enter a passphrase.

Why do passphrases matter?

Passwords aren't very secure, you already know this. If you use one that's easy to remember, it's easier to guess or brute-force (try many options until one works). If you use one that's random it's hard to remember, and thus you're more inclined to write the password down. Both of these are Very Bad Things™. This is why you're using ssh keys.

But using a key without a passphrase is basically the same as writing down that random password in a file on your computer. Anyone who gains access to your drive has gained access to every system you use that key with. This is also a Very Bad Thing™. The solution is obvious: add a passphrase.

But I don't want to enter a long passphrase every time I use the key!

Neither do we! Thankfully, there's a nifty little tool called `ssh-agent` that can save your passphrase securely so you don't have to re-enter it. If you're on OSX Leopard or later, your keys can be saved in the system's keychain to make your life even easier. Unfortunately, it takes a little work to get it up and running on Windows. Most linux installations will automatically start `ssh-agent` for you when you log in. Depending on your OS, `ssh-agent` may be automatically run for you when you log in.

For more information about SSH key passphrases, check out our help guide.

```
# Enter passphrase (empty for no passphrase): [Type a passphrase]
# Enter same passphrase again: [Type passphrase again]
```

Which should give you something like this:

```
# Your identification has been saved in /Users/you/.ssh/id_rsa.
# Your public key has been saved in /Users/you/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email@youremail.com
# Your identification has been saved in /c/Users/you/.ssh/id_rsa.
# Your public key has been saved in /c/Users/you/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email@youremail.com
```

```
# Your identification has been saved in /home/you/.ssh/id_rsa.
# Your public key has been saved in /home/you/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email@youremail.com
```

```
# Your identification has been saved in /your_home_path/.ssh/id_rsa.
# Your public key has been saved in /your_home_path/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email@youremail.com
```

Step 4: Add your SSH key to GitHub

Run the following code to copy the key to your clipboard.

```
pbcopy < ~/.ssh/id_rsa.pub# Copies the contents of the id_rsa.pub file to your clipboard
```

> **Be warned:** it is important to copy the key exactly without adding newlines or whitespace. Thankfully the `pbcopy` command makes it easy to perform this setup perfectly.

Run the following code to copy the key to your clipboard.

```
clip < ~/.ssh/id_rsa.pub# Copies the contents of the id_rsa.pub file to your clipboard
```

> **Be warned:** it is important to copy the key exactly without adding newlines or whitespace. Thankfully the `clip` command makes it easy to perform this setup perfectly.

Run the following code to copy the key to your clipboard.

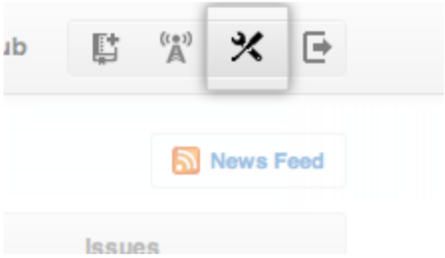
```
sudo apt-get install xclip# Downloads and installs xclip
```

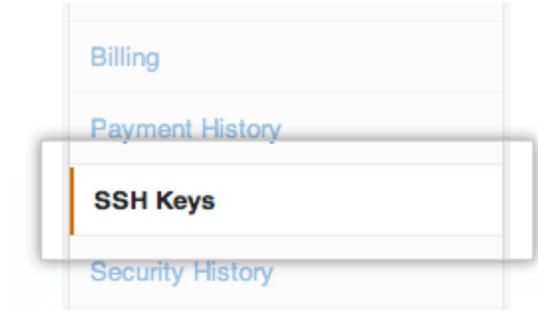
```
xclip -sel clip < ~/.ssh/id_rsa.pub# Copies the contents of the id_rsa.pub file to your clipboard
```

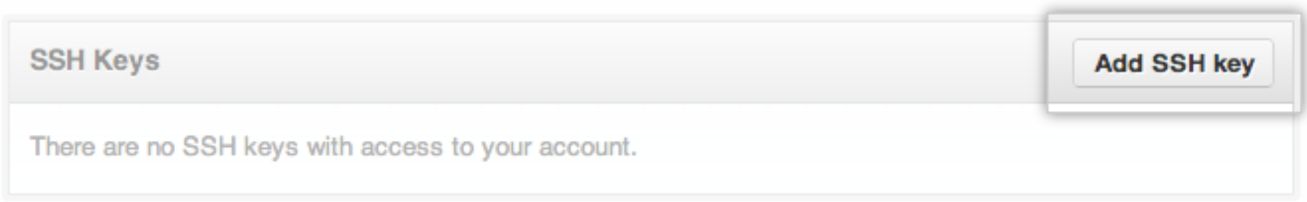
> **Be warned:** it is important to copy the key exactly without adding newlines or whitespace. Thankfully the `xclip` command makes it easy to perform this setup perfectly.

Open the id_rsa.pub file with a text editor. This is your SSH key. Select all and copy to your clipboard.

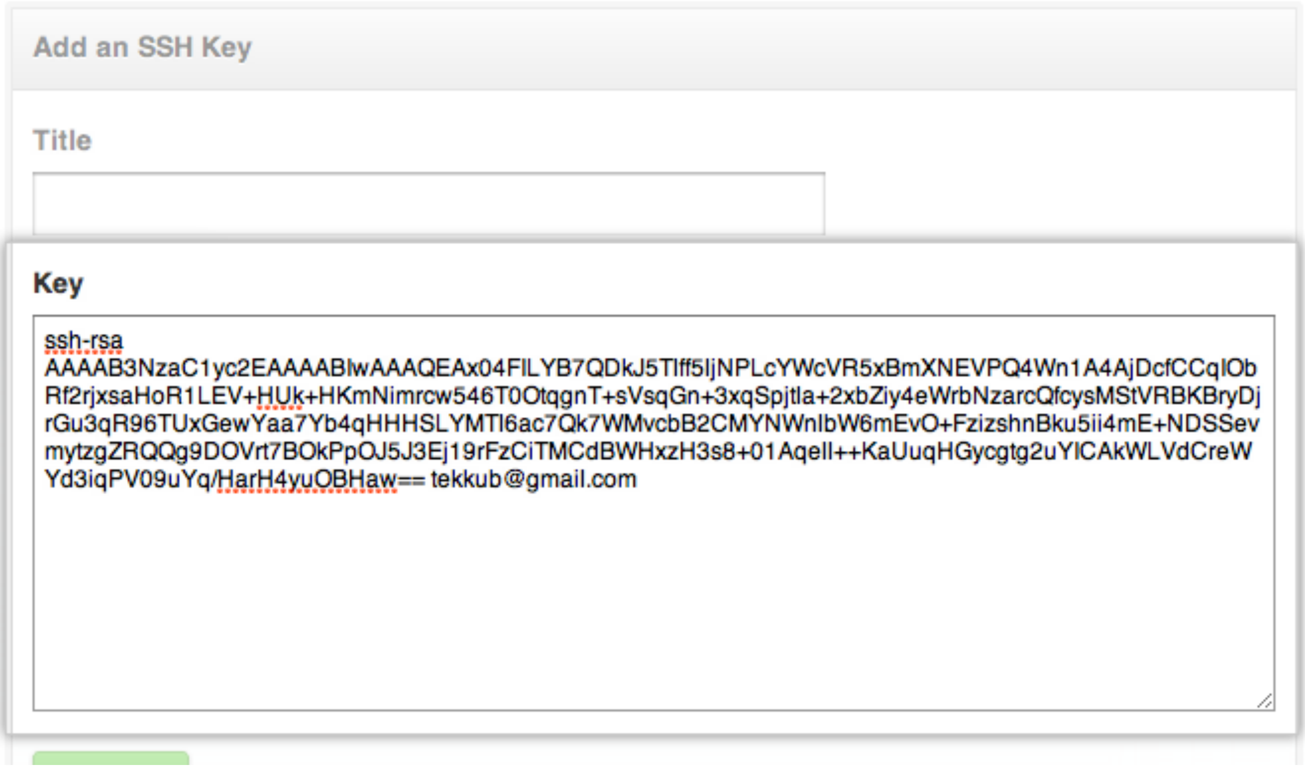
> **Be warned:** it is important to copy the key exactly without adding newlines or whitespace.

1. 

Go to your Account Settings
2. 

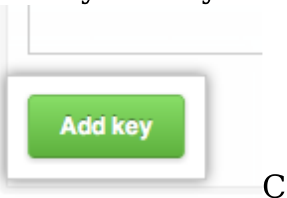
Click "SSH Keys" in the left sidebar
3. 

There are no SSH keys with access to your account.

   Click "Add SSH key"
4. 

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAx04FILYB7QDkJ5TIff5IjNPLcYWcVR5xBmXNEVPQ4Wn1A4AjDcfCCqIOb
Rf2rjxsaHoR1LEV+HUk+HKmNimrcw546T0OtqgnT+sVsqGn+3xqSpjtla+2xbZiy4eWrbNzarcQfcysMStVRBKBryDj
rGu3qR96TUxGewYaa7Yb4qHHHSLYMTl6ac7Qk7WMvcbB2CMYNWnlbW6mEvO+FzizshnBku5ii4mE+NDSSev
mytzgZRQQg9DOVrt7BOkPpOJ5J3Ej19rFzCiTMCdBWHxzH3s8+01Aqell++KaUuqHGycgtg2uYICAkWLVdCreW
Yd3iqPV09uYq/HarH4yuOBHaw== tekkub@gmail.com
```

   Paste your key into the "Key" field
5. 

Click "Add key"
6. Confirm the action by entering your GitHub password

Step 5: Test everything out

To make sure everything is working you'll now SSH to GitHub. When you do this, you will be asked to authenticate this action using your password, which for this purpose is the passphrase you created earlier. Don't change the `git@github.com` part. That's supposed to be there.

```
ssh -T git@github.com# Attempts to ssh to github
```

You may see this warning:

```
# The authenticity of host 'github.com (207.97.227.239)' can't be established.
# RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
# Are you sure you want to continue connecting (yes/no)?
```

Don't worry, this is supposed to happen. Verify that the fingerprint matches the one here and type "yes".

```
# Hi username! You've successfully authenticated, but GitHub does not
# provide shell access.
```

If that username is correct, you've successfully set up your SSH key. Don't worry about the shell access thing, you don't want that anyway.

If you see "access denied" please consider using HTTPS instead of SSH. If you need SSH start at these instructions for diagnosing the issue.

- contact a human

- Terms of Service
- Privacy
- Security