

4.3 Git on the Server - Generating Your SSH Public Key

Generating Your SSH Public Key

Many Git servers authenticate using SSH public keys. In order to provide a public key, each user in your system must generate one if they don't already have one. This process is similar across all operating systems. First, you should check to make sure you don't already have a key. By default, a user's SSH keys are stored in that user's `~/.ssh` directory. You can easily check to see if you have a key already by going to that directory and listing the contents:

```
$ cd ~/.ssh
$ ls
authorized_keys2  id_dsa          known_hosts
config           id_dsa.pub
```

You're looking for a pair of files named something like `id_dsa` or `id_rsa` and a matching file with a `.pub` extension. The `.pub` file is your public key, and the other file is the corresponding private key. If you don't have these files (or you don't even have a `.ssh` directory), you can create them by running a program called `ssh-keygen`, which is provided with the SSH package on Linux/macOS systems and comes with Git for Windows:

```
$ ssh-keygen -o
Generating public/private rsa key pair.
Enter file in which to save the key (/home/schacon/.ssh/id_rsa):
Created directory '/home/schacon/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/schacon/.ssh/id_rsa.
Your public key has been saved in /home/schacon/.ssh/id_rsa.pub.
The key fingerprint is:
d0:82:24:8e:d7:f1:bb:9b:33:53:96:93:49:da:9b:e3 schacon@mylaptop.local
```

First it confirms where you want to save the key (`.ssh/id_rsa`), and then it asks twice for a passphrase, which you can leave empty if you don't want to type a password when you use the key. However, if you do use a password, make sure to add the `-o` option; it saves the private key in a format that is more resistant to brute-force password cracking than is the default format. You can also use the `ssh-agent` tool to prevent having to enter the password each time.

Now, each user that does this has to send their public key to you or whoever is administrating the Git server (assuming you're using an SSH server setup that requires public keys). All they have to do is copy the contents of the `.pub` file and email it. The public keys look something like this:

```
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAK10UpkDhRfHY17SbrmTIpNLTGK9Tjom/BWDSU
GP1+nafz1HDTYW7hdI4yZ5ew18JH4JW9jbhUFRviQzM7x1ELEVF4h91FX5QVkbPppSwg0cda3
Pbv7kOdJ/MTyB1WXFRCr+HAo3FXRitBqxiX1nKhXpHAZsMciLq8V6RjsNAQwdsdMFvS1VK/7XA
t3FaoJoAsncM1Q9x5+3V0Ww68/eIFmb1zuUF1jQJKprX88XypNDvjYNby6vw/Pb0rwert/En
mZ+AW40ZPnPtPI89ZPmVMLuayrD2cE86Z/i18b+gw3r3+1nKatmIkjn2so1d01QraTlMqVSsbx
NrRFi9wrf+M7Q== schacon@mylaptop.local
```

For a more in-depth tutorial on creating an SSH key on multiple operating systems, see the GitHub guide on SSH keys at <https://docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>.

[prev](#) | [next](#)