



# Optional Activity: Avoiding Passwords with SSH Keys

## Why SSH Keys?

So far, you have used usernames and passwords to login to a remote server. Passwords have a lot of disadvantages. Many people try to memorize passwords, which leads to them choosing a password that can be easily guessed or broken by an attacker. People also reuse passwords across multiple sites, so that if an attacker breaks a password they can compromise multiple accounts. You also need to type a password (or copy and paste it from a password manager) each time you use it.

If you insist on using passwords, be sure to choose a long, random password to avoid attackers guessing your password and breaking into your account. Use a password manager to remember it and then copy and paste it each time.

However, I recommend you instead use SSH keys. An SSH key uses strong cryptographic standards, meaning an attacker would need billions of years to break your key. In addition, you can configure SSH keys so that they are unlocked when you login to your account, so that you can avoid typing a password each time you use it.

## A Brief Introduction to Public Key Cryptography

SSH keys use public key cryptography. You will create both a public key and a private key. Your public key, as the name suggests, is public -- you can safely give this to any person or system. Your private key should remain private. Ideally, you should protect it with a password, as described below, so that if it is stolen, there is an extra level of protection that prevents someone from using it.

The public and private key have a special, mathematical relationship. Anything you encrypt with your **private key** can be decrypted with your **public key**. No other key will decrypt it. Likewise, anything someone encrypts with your **public key** can only be decrypted with your **private key**.

Public key cryptography can be used in a variety of ways, but we will be using it for **authentication** -- meaning you can prove your identity to another system. Here is how that will work:

- You create a public and private key pair.
- You store your public key with a service, such as with a Digital Ocean server or with GitHub.
- When you want to login, you present your public key to the service.
- The service asks you to prove you own that public key by having you encrypt something with your private key.

- The server will decrypt that thing with your public key, thereby proving you own the associated private key.

The nice thing about this is that it is completely automated once you set it up. Your ssh program just needs access to your private key, and it will do everything for you.

## Setup

1. To setup your SSH key, you will use a program called **ssh-keygen**. This will generate your public and private key pair and store them in a directory called **~/.ssh**. Normally, the public key will be called **id\_rsa.pub** and the private key will be called **id\_rsa**. Be sure to use a strong password when asked for one. I recommend a long phrase or a [correct-battery-horse-staple password](http://correcthorsebatteryaple.net/) (<http://correcthorsebatteryaple.net/>).
2. To configure a remote server (e.g. Digital Ocean) with your public key, you can use a program called **ssh-copy-id**. You can do this with **ssh-copy-id user@host**. All this does is copy your public key to the remote host and then add it to a file called **~/.ssh/authorized\_keys**. You can also [follow these instructions to add the key to your GitHub account](https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/) (<https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/>).

You should now be able to use **ssh user@host**. You will be asked to enter the password that you used to create the key. This unlocks the private key so you can use it.

## Avoiding Passwords

As described above, you still have to enter the password for the SSH key so that you can unlock it each time that you use it. This gives you the advantage of greater security as compared to using the password to login, but still the inconvenience of having to remember this password.

To avoid this hassle, you can use an SSH agent. Follow one of these instructions:

### OS X

Run this command:

```
ssh-add -K ~/.ssh/id_rsa
```

to add the key to your keychain. Then create an `~/.ssh/config` file. You can do this with

```
touch ~/.ssh/config
```

In that `~/.ssh/config` file, add the following lines:

```
Host *  
  UseKeychain yes
```

```
AddKeysToAgent yes
IdentityFile ~/.ssh/id_rsa
```

The `IdentityFile` is the path to your private key. The `UseKeychain yes` is the key part, which tells SSH to look in your OSX keychain for the key passphrase.

## Windows

Use Git Bash and use the following [instructions to for Git Bash](https://gist.github.com/bsara/5c4d90db3016814a3d2fe38d314f9c23).  
(<https://gist.github.com/bsara/5c4d90db3016814a3d2fe38d314f9c23>)

## Linux

Run this command:

```
ssh-add ~/.ssh/id_rsa
```









to add the key to your Gnome keyring. For details see [this link](https://wiki.archlinux.org/index.php/GNOME/Keyring#SSH_keys)  
([https://wiki.archlinux.org/index.php/GNOME/Keyring#SSH\\_keys](https://wiki.archlinux.org/index.php/GNOME/Keyring#SSH_keys)).

## Testing

You should be able to use ssh to connect to your server without having to enter your password each time.

## Resources

In case you need a detailed walk-through of this, or you mess up, try deleting your `~/.ssh` folder on both your local and remote machines and going through a tutorial. Here is a [Mac-specific tutorial from GitHub](https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/) (<https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/>) that covers the same material as above.

✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/372214">https://byu.instructure.com/courses/5724/modules/items/372214</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/362817">https://byu.instructure.com/courses/5724/modules/items/362817</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/362818">https://byu.instructure.com/courses/5724/modules/items/362818</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/370432">https://byu.instructure.com/courses/5724/modules/items/370432</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/362923">https://byu.instructure.com/courses/5724/modules/items/362923</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/362924">https://byu.instructure.com/courses/5724/modules/items/362924</a> )
✓ 	✓  ( <a href="https://byu.instructure.com/courses/5724/modules/items/362925">https://byu.instructure.com/courses/5724/modules/items/362925</a> )

✓  (<https://byu.instructure.com/courses/5724/modules/items/362926>)

 (<https://byu.instructure.com/courses/5724/modules/items/427808>)

📍  (<https://byu.instructure.com/courses/5724/modules/items/362930>)

✓  (<https://byu.instructure.com/courses/5724/modules/items/362927>)

✓  (<https://byu.instructure.com/courses/5724/modules/items/362929>)

✓  (<https://byu.instructure.com/courses/5724/modules/items/362928>)

✓  (<https://byu.instructure.com/courses/5724/modules/items/362833>)

✓  (<https://byu.instructure.com/courses/5724/modules/items/415269>)