Creating an SSH Key Pair

We've seen that **ssh** is a program used to log into cs.bridgewater.edu.

SSH is a **communication protocol** that can be used to *securely* copy data from one server to another. In order to authenticate ourselves to a server we can either provide the server with a user name and password, or we can use a pair of **SSH keys** (one private and one public).

Eventually, we're going to use Git to copy our application code from cs.bridgewater.edu to the GitHub servers and authenticate ourselves using SSH keys.

Create an SSH Key Pair

Log onto cs.bridgewater.edu.

We are going to create a pair of SSH keys using the **ssh-keygen** program. When running the program take care to follow the following steps:

★ The ssh-keygen program will ask you to enter the name of a file in which to save the key. We want to use the default location. Before you press enter to use the default location, write down the default location that appears on the screen (you will need this later).

★ The ssh-keygen program will then ask you to enter a passphrase. Press enter, for *no passphrase*, and press enter a second time to accept the empty passphrase.

Type the following command in your terminal to create a pair of SSH keys, replacing *joe@example.com* with your BC email addresses (leaving the quotes).

```
$ ssh-keygen -t rsa -b 4096 -C "joe@example.com"
```

Inspect the Key Files

You can verify that your SSH key pair files were created by using the **cd** command in the terminal to change your working directory to the location that the key files were saved.

For example, when a student with BC username *joe* runs ssh-keygen, it would state that the file named **id_rsa** was created in the directory **/home/joe/.ssh**. Joe would change his working directory to */home/joe/.ssh*.

Enter the following command, replacing **joe** with your BC username.

```
$ cd /home/joe/.ssh
```

Then to view the contents of the directory you can use program named **ls**.

```
$ ls
```

You should see at least two files; one named **id_rsa** and one named **id_rsa.pub**. The file named *id_rsa* is your **private key** file. Keep it secret. Anyone with access to it can masquerade as you. The file named *id_rsa.pub* is your **public key** which we will later provide to GitHub.

You can view the contents of the files using the **cat** program.

```
$ cat id_rsa
$ cat id_rsa.pub
```

Register Your Private Key with the SSH Agent on cs.bridgewater.edu

Next, start the ssh agent on cs.bridgewater.edu, if it is not already running, by running the following command.

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

You should see an agent pid (process identifier) printed to the screen.

Alas, provide the ssh agent with your private key by running the following command.

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

You now have a pair of ssh keys, started the ssh agent, and registered your private key with the ssh agent.

Provide GitHub With Your Public Key

Log onto GitHub.com.

Click on the icon in the upper right corner, scroll down, and choose **Settings**.

On the left-hand side pane of the Settings screen, choose **SSH and GPG keys**.

Press the **New SSH key** button.

Enter a name for the key (e.g. cs.bridgewater.edu), and copy the contents of your **id_rsa.pub** file (your public key) into the key field. Press **Add SSH key**.

Test the SSH Communication Channel between cs.bridgewater.edu and GitHub

Run the following command from the terminal on cs.bridgewater.edu to verify that you can communicate with GitHub via ssh.

```
$ ssh -T git@github.com
```

When prompted if you want to continue, enter **yes** and press enter.
If successful, you should see a message that reads "*You've successfully authenticated, but GitHub does not provide shell access.*"