CY 5010: Foundations to Information Assurance

Lab 1: SSH Authentication

Part 1: Password Authentication

We successfully ran the ssh command to connect to the master VM using username and password provided as shown on figure 1.

1. We ran the id command to view User Identifier of our team which is 1046 and group Id (1048) which was assigned to team8 on the master server.
2. We also ran ifconfig command to view the configuration of network interfaces on the master server.

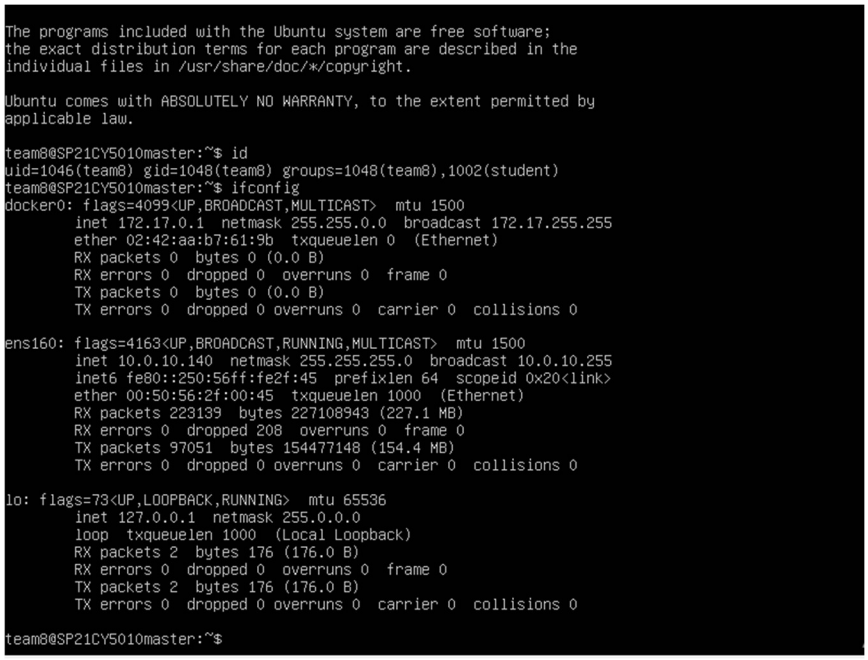


Figure 1

Part 2: Public Key Authentication

In part 1, we had to type in the password for authentication when we ran the ssh command. Thus, we now set up public key authentication in part 2. Exited the master VM and configured the following on our local VM.

1. Generated RSA private and public keys using the following command.

   ssh-keygen -t rsa -b 2048
2. Then copied the public key we generated to the master VM. This will be used later for authentication when we connect to the master server using ssh.

   ssh-copied -i ~/ssh/id-rsa.pub team8@cy5010.ccs.neu.edu -p 17001

3. Now that we have the private key on our VM and public key on the master server, we tried to ssh to the master server using this key pair. We were able to successfully ssh without typing the password for authentication as seen on figure 2.

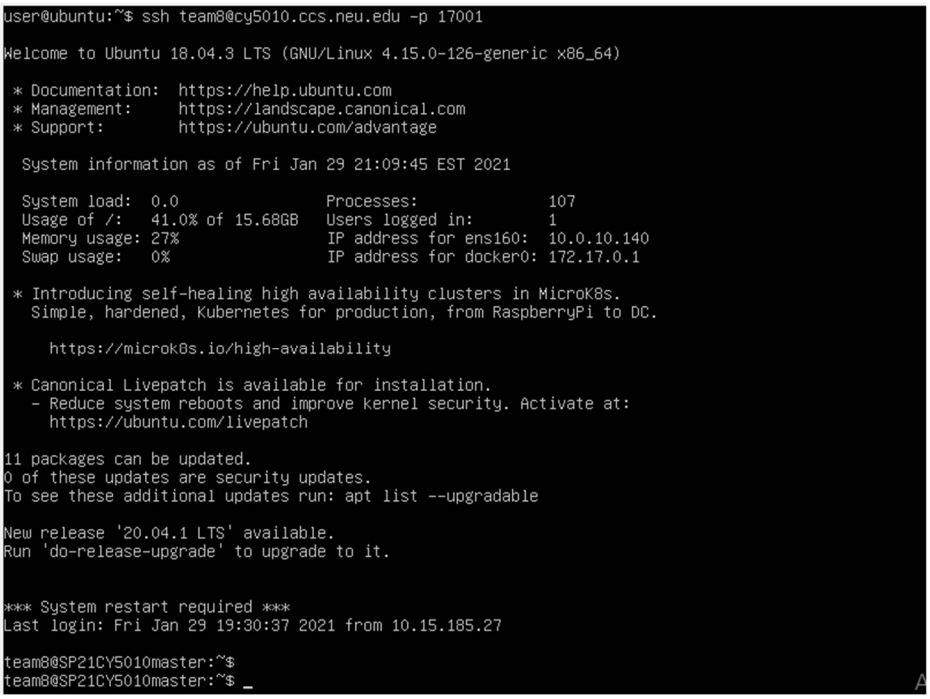


Figure 2

Part 3: Securely Copying Files To/From the Host/VM

In this part, we followed the instructions given from the lab tasks.

1. Copied the Test.jpg file from the master server to our VM using the following command. scp -P 17001 team8@cy5001.ccs.neu.edu:/home/SharedFolder/Test.jpg .
2. Created details.team file using the following command that is also seen on figure 3. echo -e “CY5010 Spring 2021\n TEAM 8 \n +Sowmyashree Bevur Manya Vankatesh\n +Vanessa Kibaja” > details.team

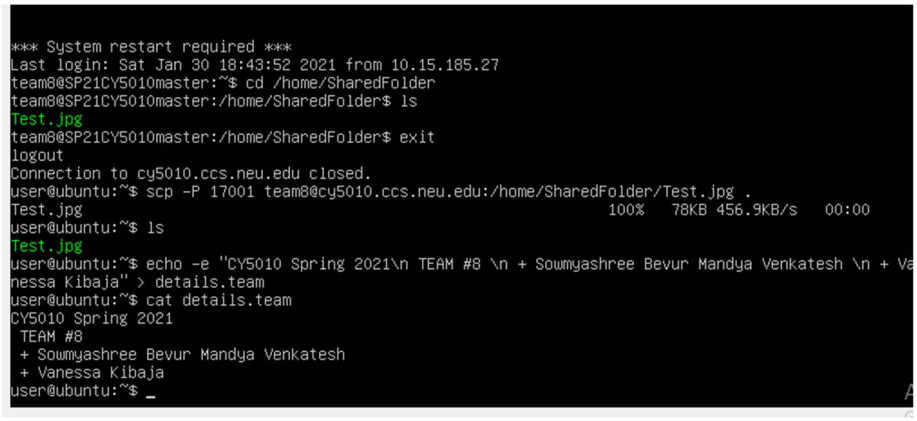


Figure 3

3. Appended the contents of details.team file to Test.jpg file using the following command.
   cat details.team >> Test.jpg
4. Generated and moved the sha-256 hash value of Test.jpg file to hash.team file using the following command as seen on figure 4.
   sha256sum Test.jpg > hast.team

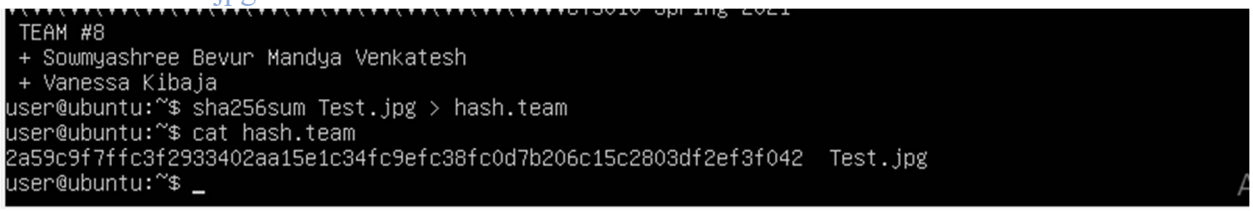


Figure 4

5. Finally created a lab1 directory on the master server (mkdir lab1) and moved all the files from step 1-4 in this directory as seen below.
   scp -P 17001 /home/user/hash.team team8@cy5010.ccs.neu.edu:/home/team8/lab1
   scp -P 17001 /home/user/Test.jpg team8@cy5010.ccs.neu.edu:/home/team8/lab1
   scp -P 17001 /home/user/details.team team8@cy5010.ccs.neu.edu:/home/team8/lab1

Figure 5

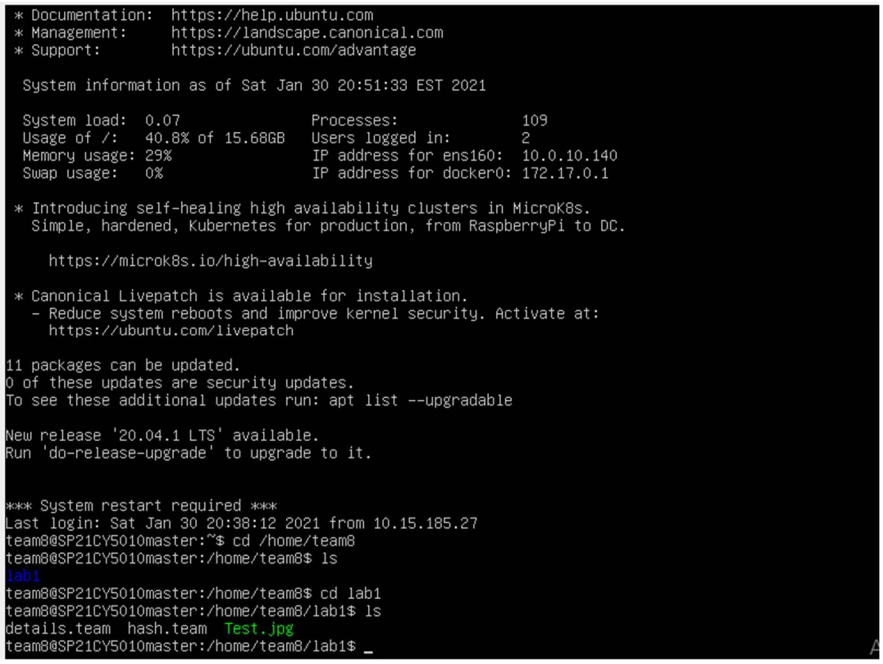


Figure 6

Questions:

1. **How can you debug on your SSH connection?**
   We can debug SSH by running it in verbose mode. Verbose option provides more in-depth detail on what the ssh command is doing like the status of connection, error message if the connection failed, etc. Hence, we can debug using the information obtained from verbose.

   Verbose has 3 levels (-v, -vv, -vvv) in which level 3 (-vvv) provides more advance information than level 1 (-v). For our case, we can debug ssh using the following command.

   ssh -v team8@cy5010.ccs.neu.edu

2. **What is the other use-cases (functionalities) for SSH? Explain in brief.**
   Here are most common uses of SSH:
   a. SSH command itself is being used to securely connect two different machines with authentication of either a password or using private and public keys. This helps to eliminate intermission from Network sniffers. (Which was realized in this Lab Assignment)

   b. Securely copy files from source to destination using the scp command. Here the files are encrypted before transferred and decrypted as soon as it arrives at the destinations. This one-way process makes it easier for users to transfer their files without manually

encrypting and decrypting their files. (We also used this functionality in our Assignment)

c. Port Forwarding/ Tunneling: SSH enables Tunneling. This functionality enables insecure applications like Ftp run in a much secure environment. SSH encrypts the entire data traffic hence creating an end to end encryption between the users.

d. Access control: We can utilize SSH to not only connect to servers but also provide specific level-based access control. We can restrict the programs used by Users with specific access to the given servers. For example, we can disable email access to certain users.

e. Secure remote command: A security administrator can use SSH to view all the active processes or opened ports on many devices on the same network using SSH. Since SSH can open a lot of connections to many server ports over a given network, as an administrator we can utilize this feature to execute a single command across all the given servers thus reducing the time of operations. For example, if we want to start all the apache servers on the given servers, we can simply do this using the above feature.

3. **State the linux command to securely copy a file from your VM to the master server.**

scp -P 17001 /home/user/hash.team team8@cy5001.ccs.neu.edu:/home/team8/lab1

We used the above command the copy hash.team file from the local VM onto the master server. /home/user/hash.team was the pathname for the file on the VM to be copied onto the master server. We used the scp command (scp) that connects to the master server team8@cy5001.ccs.neu.edu through port 17001 (-P 17001). The semi-colon (:) separates the server address and the file path of the destination directory.

4. **Submit the screenshot of the ssh-keygen command and its output. Explain in your own words the command which you used to create a stronger key.**

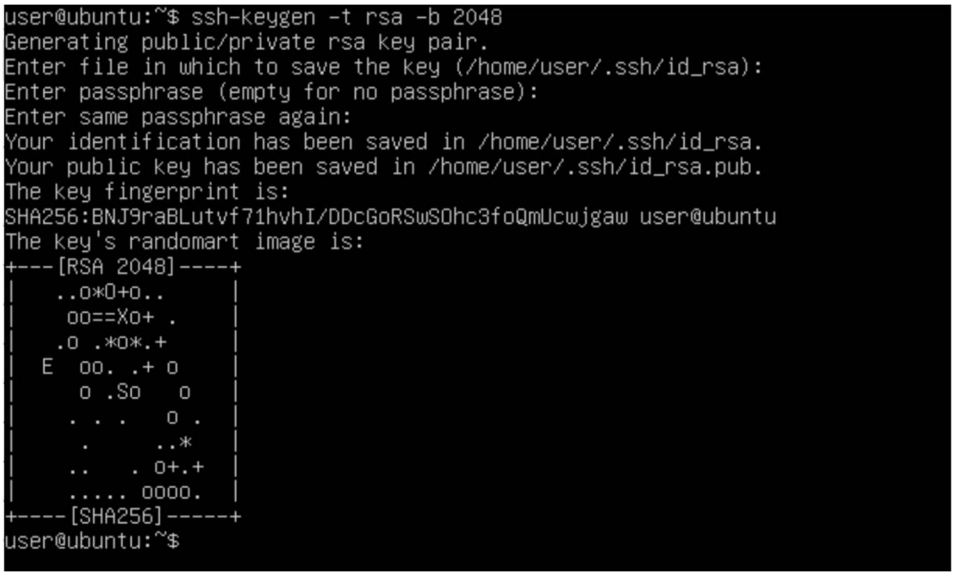
We ran the following command to generate the public and private keys from our VM.

ssh-keygen -t rsa -b 2048

The ssh-keygen command above, uses the -t option to select encryption algorithm (rsa in our case) and -b option for specifying the key size (here 2048 bits). It is recommended to use minimum of 2048 bit key size for rsa since it would take thousands of years to crack it.

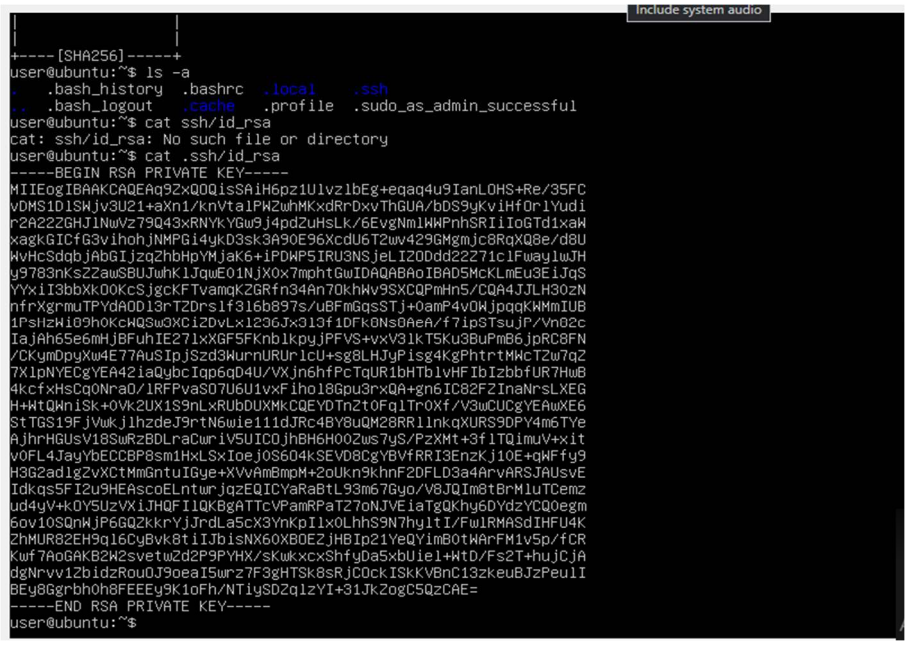
We observed the following output for the command that we ran:

- After generating the key pairs, it asked for a file path to save the keys in and passphrase to use for authentication (home//.ssh.id_rsa directory). We decided to keep the passphrase empty.
- It also generated a key print. This is unique value to the key which is obtained by using a cryptographic hash function. SSH used SHA-256 to do the same.
- In addition to the Key print it also provided with key's randomart image.



ssh-keygen

Here are the screen shots of the generated keys:



Private Key

Public Key