Name: Ashley L. Mallari	**Date Performed: August 24, 2023**
Course/Section: CPE31S6	**Date Submitted: September 7, 2023**
Instructor: Dr. Jonathan V. Taylar	**Semester and SY: 1st Sem | 2023-2024**

Activity 2: SSH Key-Based Authentication and Setting up Git

1. **Objectives:**

1.1 Configure remote and local machine to connect via SSH using a KEY instead of using a password

1.2 Create a public key and private key

1.3 Verify connectivity

1.4 Setup Git Repository using local and remote repositories

1.5 Configure and Run ad hoc commands from local machine to remote servers

Part 1: Discussion

It is assumed that you are already done with the last Activity (**Activity 1: Configure Network using Virtual Machines).** *Provide screenshots for each task*.

It is also assumed that you have VMs running that you can SSH but requires a password. Our goal is to remotely login through SSH using a key without using a password. In this activity, we create a public and a private key. The private key resides in the local machine while the public key will be pushed to remote machines. Thus, instead of using a password, the local machine can connect automatically using SSH through an authorized key.

What Is ssh-keygen?

Ssh-keygen is a tool for creating new authentication key pairs for SSH. Such key pairs are used for automating logins, single sign-on, and for authenticating hosts.

SSH Keys and Public Key Authentication

The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

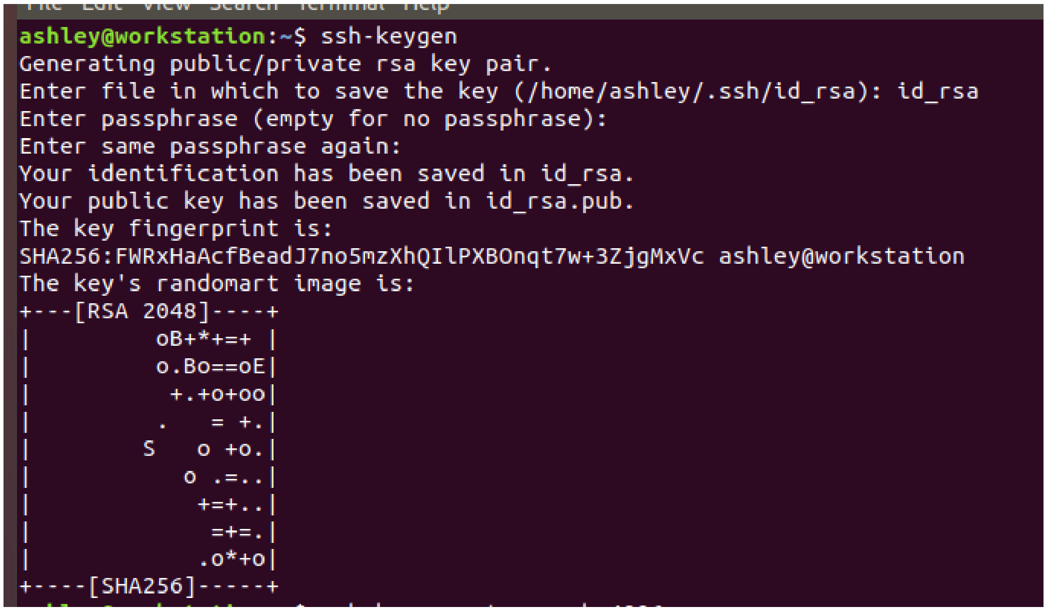
SSH introduced public key authentication as a more secure alternative to the older .rhosts authentication. It improved security by avoiding the need to have password stored in files and eliminated the possibility of a compromised server stealing the user's password.

However, SSH keys are authentication credentials just like passwords. Thus, they must be managed somewhat analogously to usernames and passwords. They should have a proper termination process so that keys are removed when no longer needed.

Task 1: Create an SSH Key Pair for User Authentication

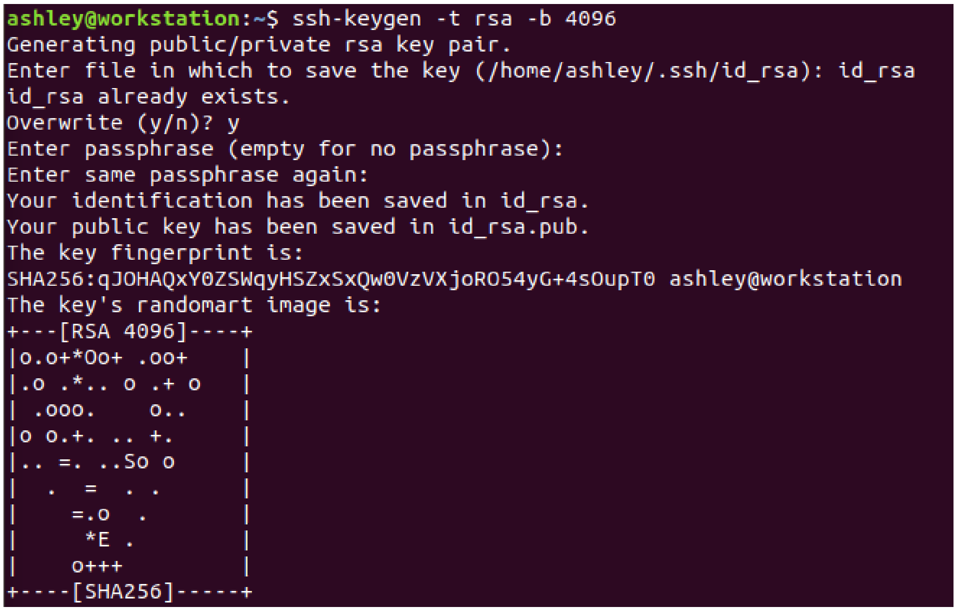
1. The simplest way to generate a key pair is to run *ssh-keygen* without arguments. In this case, it will prompt for the file in which to store keys. First,

the tool asked where to save the file. SSH keys for user authentication are usually stored in the users .ssh directory under the home directory. However, in enterprise environments, the location is often different. The default key file name depends on the algorithm, in this case *id_rsa* when using the default RSA algorithm. It could also be, for example, *id_dsa* or *id_ecdsa*.



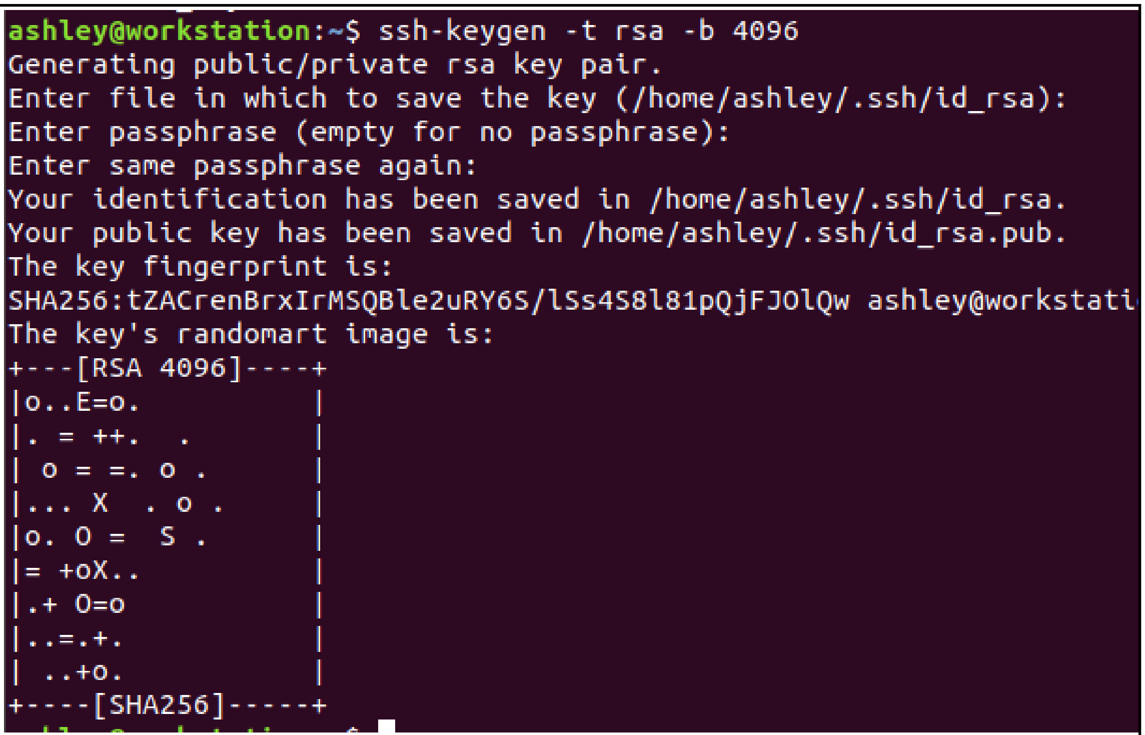
```
ashley@workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ashley/.ssh/id_rsa): id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:FWRxHaAcfBeadJ7no5mzXhQIlPXBOnqt7w+3ZjgMxVc ashley@workstation
The key's randomart image is:
+---[RSA 2048]----+
|         oB+*+=+ |
|         o.Bo==oE|
|          +.+o+oo|
|         .   = +.|
|        S   o +o.|
|           o .=..|
|            +=+..|
|             =+=.|
|            .o*+o|
+----[SHA256]-----+
```

2. Issue the command *ssh-keygen -t rsa -b 4096.* The algorithm is selected using the -t option and key size using the -b option.



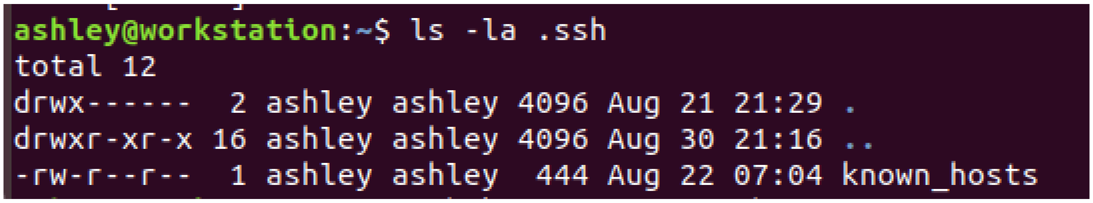
```
ashley@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ashley/.ssh/id_rsa): id_rsa
id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:qJOHAQxY0ZSWqyHSZxSxQw0VzVXjoRO54yG+4sOupT0 ashley@workstation
The key's randomart image is:
+---[RSA 4096]----+
|o.o+*Oo+ .oo+    |
|.o .*.. o .+ o   |
| .ooo.    o..    |
|o o.+. .. +.     |
|.. =. ..So o     |
|  .  =  . .      |
|    =.o  .       |
|     *E .        |
|    o+++         |
+----[SHA256]-----+
```

3. When asked for a passphrase, just press enter. The passphrase is used for encrypting the key, so that it cannot be used even if someone obtains the private key file. The passphrase should be cryptographically strong.



```
ashley@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ashley/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ashley/.ssh/id_rsa.
Your public key has been saved in /home/ashley/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:tZACrenBrxIrMSQBle2uRY6S/lSs4S8l81pQjFJOlQw ashley@workstati
The key's randomart image is:
+---[RSA 4096]----+
|o..E=o.          |
|. = ++.   .      |
| o = =. o .      |
|... X  . o .     |
|o. O =  S .      |
|= +oX..          |
|.+ O=o           |
|..=.+.           |
| ..+o.           |
+----[SHA256]-----+
```

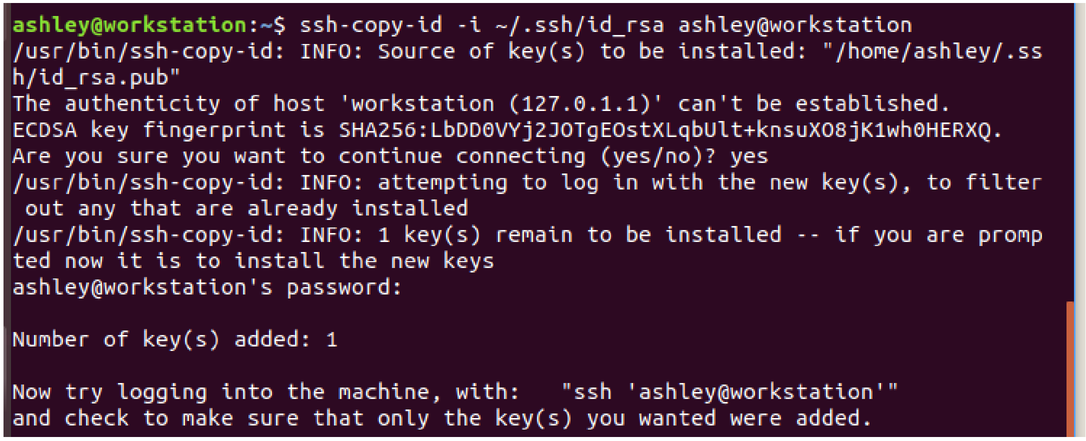
4. Verify that you have created the key by issuing the command *ls -la .ssh.* The command should show the .ssh directory containing a pair of keys. For example, id_rsa.pub and id_rsa.



```
ashley@workstation:~$ ls -la .ssh
total 12
drwx------  2 ashley ashley 4096 Aug 21 21:29 .
drwxr-xr-x 16 ashley ashley 4096 Aug 30 21:16 ..
-rw-r--r--  1 ashley ashley  444 Aug 22 07:04 known_hosts
```

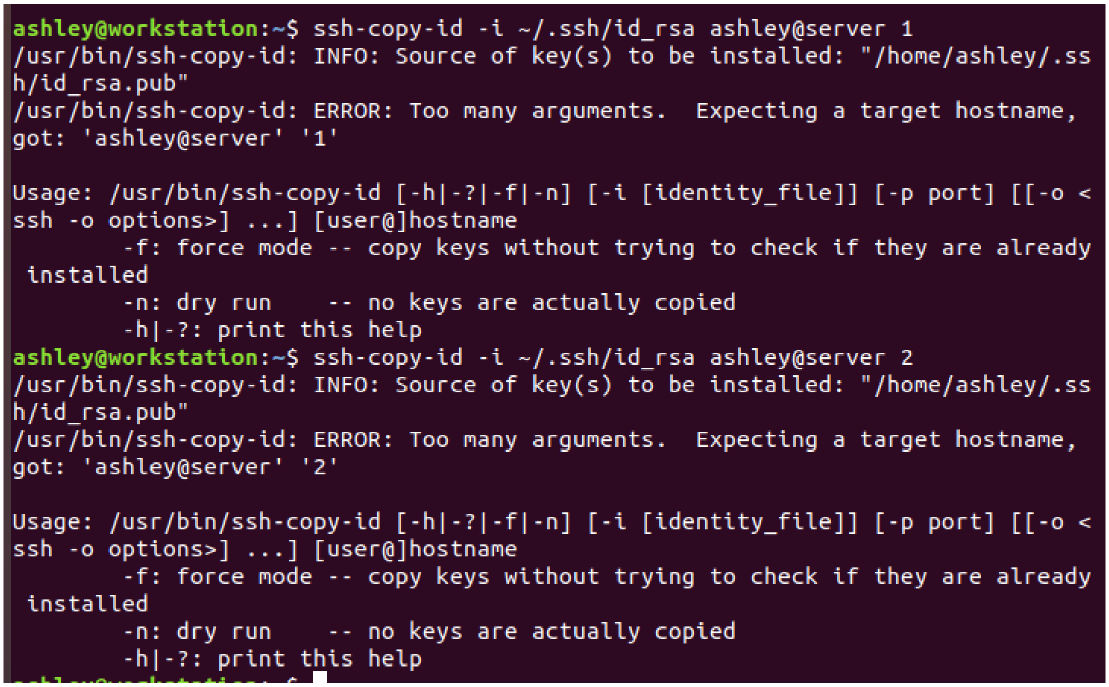
Task 2: Copying the Public Key to the remote servers

1. To use public key authentication, the public key must be copied to a server and installed in an *authorized_keys* file. This can be conveniently done using the *ssh-copy-id* tool.
2. Issue the command similar to this: *ssh-copy-id -i ~/.ssh/id_rsa user@host*



```
ashley@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa ashley@workstation
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ashley/.ss
h/id_rsa.pub"
The authenticity of host 'workstation (127.0.1.1)' can't be established.
ECDSA key fingerprint is SHA256:LbDD0VYj2JOTgEOstXLqbUlt+knsuXO8jK1wh0HERXQ.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
 out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are promp
ted now it is to install the new keys
ashley@workstation's password: 

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ashley@workstation'"
and check to make sure that only the key(s) you wanted were added.
```

3. Once the public key has been configured on the server, the server will allow any connecting user that has the private key to log in. During the login process, the client proves possession of the private key by digitally signing the key exchange.
4. On the local machine, verify that you can SSH with Server 1 and Server 2. What did you notice? Did the connection ask for a password? If not, why? 0



```
ashley@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa ashley@server 1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ashley/.ss
h/id_rsa.pub"
/usr/bin/ssh-copy-id: ERROR: Too many arguments.  Expecting a target hostname, 
got: 'ashley@server' '1'

Usage: /usr/bin/ssh-copy-id [-h|-?|-f|-n] [-i [identity_file]] [-p port] [[-o <
ssh -o options>] ...] [user@]hostname
        -f: force mode -- copy keys without trying to check if they are already
 installed
        -n: dry run    -- no keys are actually copied
        -h|-?: print this help
ashley@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa ashley@server 2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ashley/.ss
h/id_rsa.pub"
/usr/bin/ssh-copy-id: ERROR: Too many arguments.  Expecting a target hostname, 
got: 'ashley@server' '2'

Usage: /usr/bin/ssh-copy-id [-h|-?|-f|-n] [-i [identity_file]] [-p port] [[-o <
ssh -o options>] ...] [user@]hostname
        -f: force mode -- copy keys without trying to check if they are already
 installed
        -n: dry run    -- no keys are actually copied
        -h|-?: print this help
```

Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
2. How do you know that you already installed the public key to the remote servers?

Part 2: Discussion

Provide screenshots for each task.

It is assumed that you are done with the last activity (**Activity 2: SSH Key-Based Authentication**).

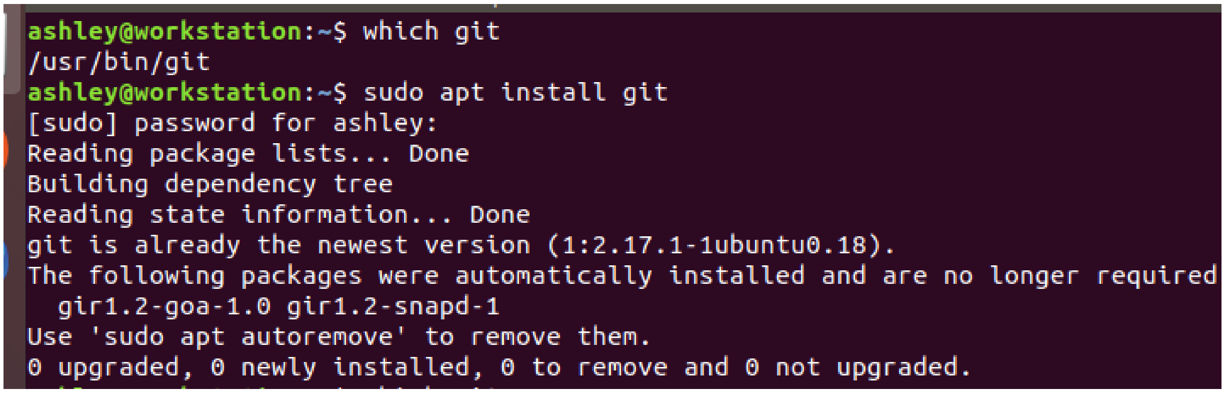
Set up Git

At the heart of GitHub is an open-source version control system (VCS) called Git. Git is responsible for everything GitHub-related that happens locally on your computer. To use Git on the command line, you'll need to download, install, and configure Git on your computer. You can also install GitHub CLI to use GitHub from the command line. If you don't need to work with files locally, GitHub lets you complete many Git-related actions directly in the browser, including:

- Creating a repository
- Forking a repository
- Managing files
- Being social

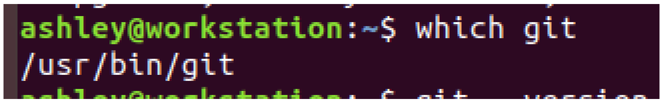
Task 3: Set up the Git Repositorygiyt

1. On the local machine, verify the version of your git using the command *which git.* If a directory of git is displayed, then you don't need to install git. Otherwise, to install git, use the following command: *sudo apt install git*



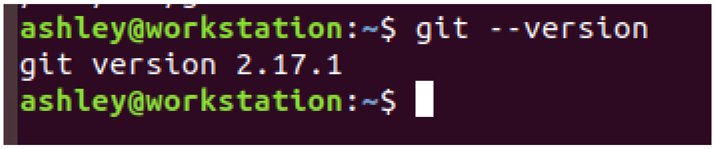
```
ashley@workstation:~$ which git
/usr/bin/git
ashley@workstation:~$ sudo apt install git
[sudo] password for ashley:
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.17.1-1ubuntu0.18).
The following packages were automatically installed and are no longer required
  gir1.2-goa-1.0 gir1.2-snapd-1
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git.*



```
ashley@workstation:~$ which git
/usr/bin/git
```

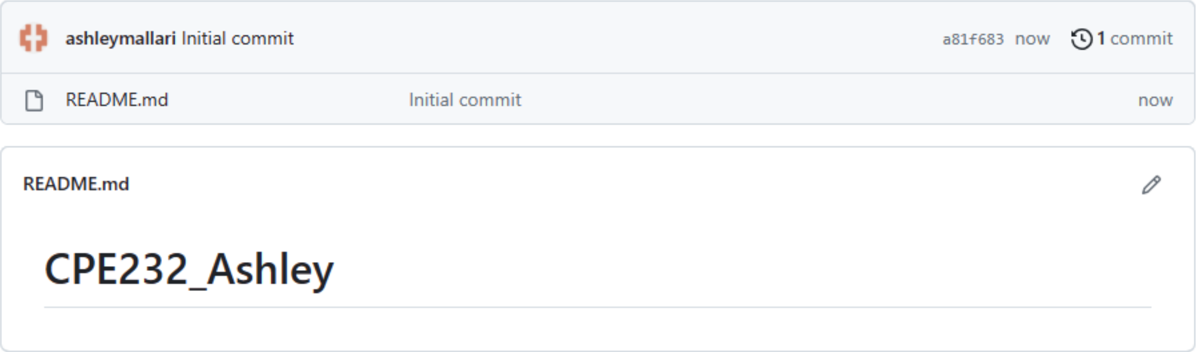
3. The version of git installed in your device is the latest. Try issuing the command *git --version* to know the version installed.



```
ashley@workstation:~$ git --version
git version 2.17.1
ashley@workstation:~$ 
```

4. Using the browser in the local machine, go to www.github.com.

5. Sign up in case you don’t have an account yet. Otherwise, login to your GitHub account.
    a. Create a new repository and name it as CPE232_yourname. Check Add a README file and click Create repository.



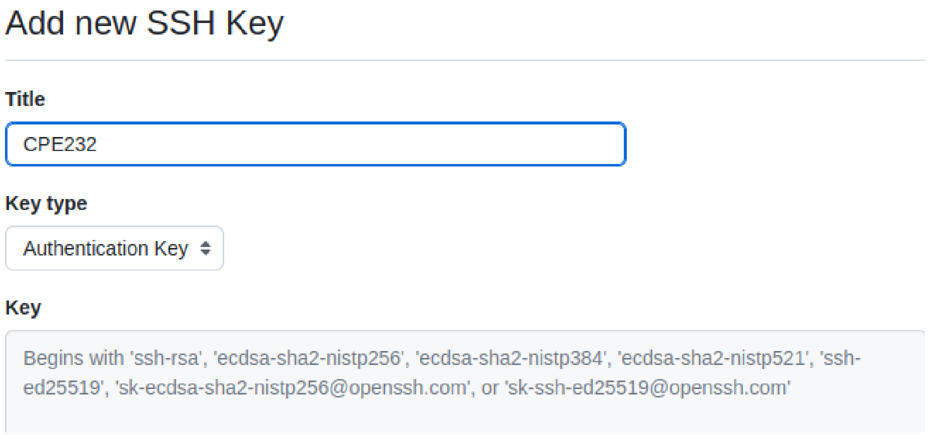
ashleymallari Initial commit a81f683 now 1 commit

README.md Initial commit now

README.md

CPE232_Ashley

    b. Create a new SSH key on GitHub. Go your profile’s setting and click SSH and GPG keys. If there is an existing key, make sure to delete it. To create a new SSH keys, click New SSH Key. Write CPE232 key as the title of the key.



Add new SSH Key

Title

CPE232

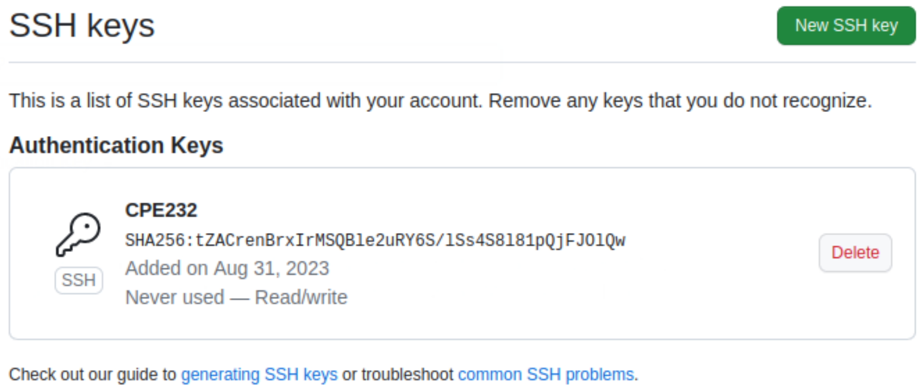
Key type

Authentication Key

Key

Begins with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'

    c. On the local machine’s terminal, issue the command cat .ssh/id_rsa.pub and copy the public key. Paste it on the GitHub key and press Add SSH key.



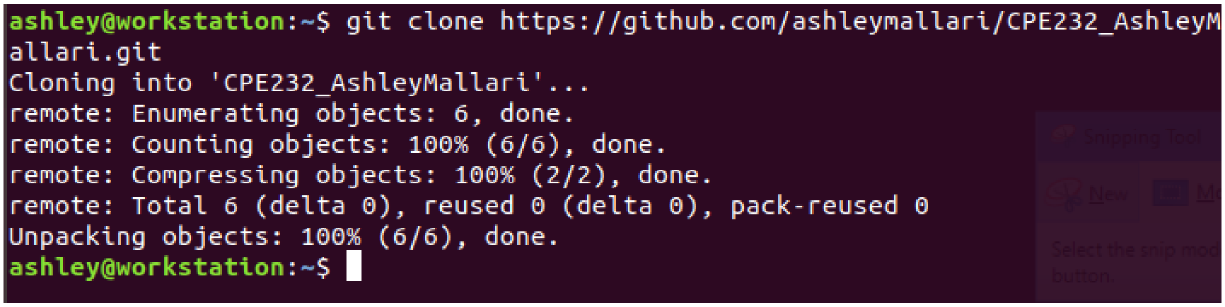
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

CPE232
SHA256:tZACrenBrxIrMSQBle2uRY6S/lSs4S8l81pQjFJOlQw
Added on Aug 31, 2023
Never used — Read/write

Check out our guide to generating SSH keys or troubleshoot common SSH problems.

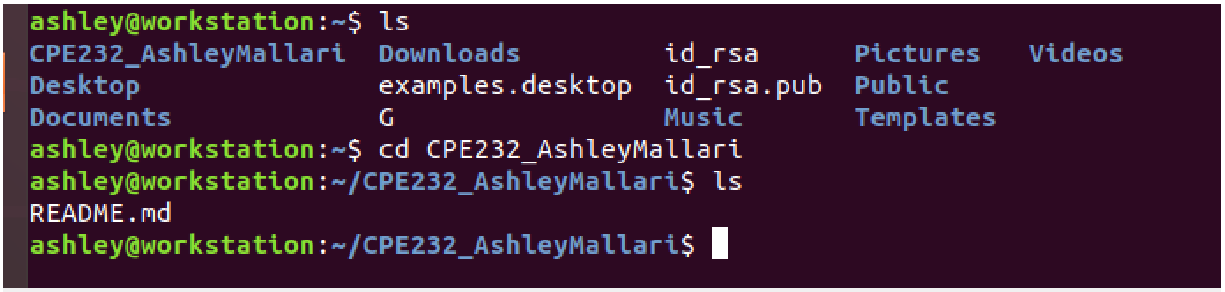
d. Clone the repository that you created. In doing this, you need to get the link from GitHub. Browse to your repository as shown below. Click on the Code drop down menu. Select SSH and copy the link.



```
ashley@workstation:~$ git clone https://github.com/ashleymallari/CPE232_AshleyM
allari.git
Cloning into 'CPE232_AshleyMallari'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
ashley@workstation:~$ 
```

e. Issue the command git clone followed by the copied link. For example, *git clone git@github.com:jvtaylar-cpe/CPE232_yourname.git*. When prompted to continue connecting, type yes and press enter.

f. To verify that you have cloned the GitHub repository, issue the command *ls*. Observe that you have the CPE232_yourname in the list of your directories. Use CD command to go to that directory and LS command to see the file README.md.

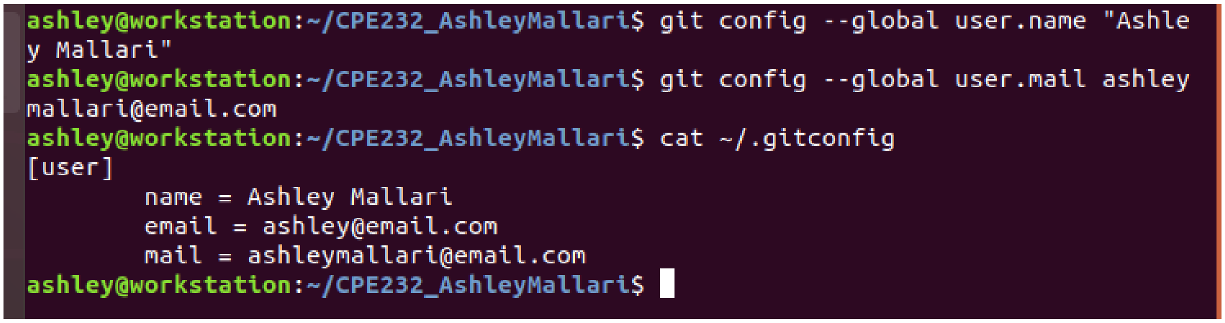


```
ashley@workstation:~$ ls
CPE232_AshleyMallari  Downloads         id_rsa      Pictures   Videos
Desktop               examples.desktop  id_rsa.pub  Public
Documents             G                 Music       Templates
ashley@workstation:~$ cd CPE232_AshleyMallari
ashley@workstation:~/CPE232_AshleyMallari$ ls
README.md
ashley@workstation:~/CPE232_AshleyMallari$ 
```

g. Use the following commands to personalize your git.

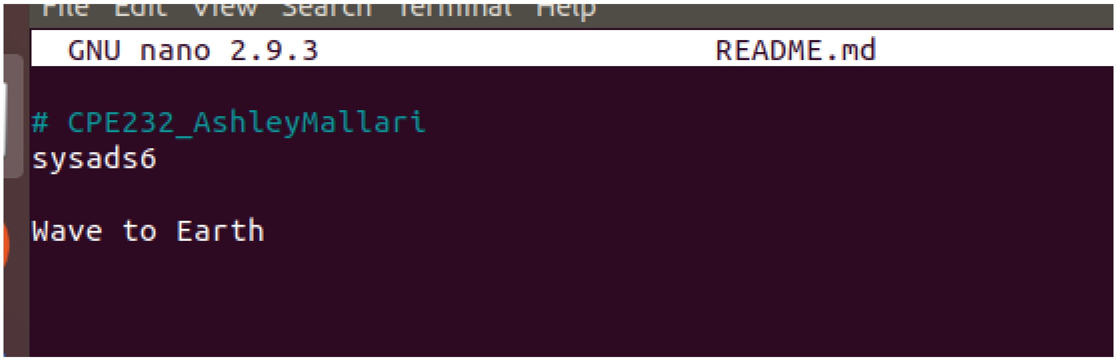
- *git config --global user.name "Your Name"*

- *git config --global user.email yourname@email.com*
- Verify that you have personalized the config file using the command *cat ~/.gitconfig*



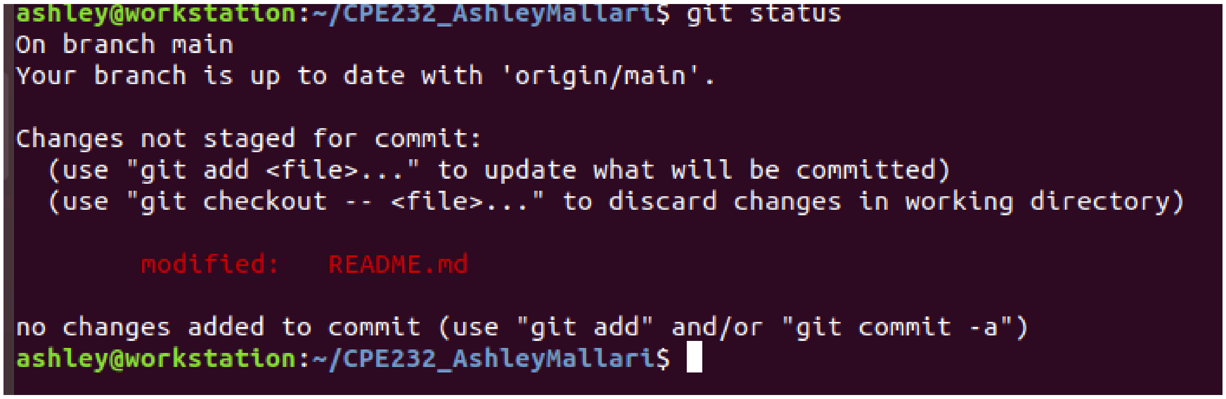
```
ashley@workstation:~/CPE232_AshleyMallari$ git config --global user.name "Ashle
y Mallari"
ashley@workstation:~/CPE232_AshleyMallari$ git config --global user.mail ashley
mallari@email.com
ashley@workstation:~/CPE232_AshleyMallari$ cat ~/.gitconfig
[user]
        name = Ashley Mallari
        email = ashley@email.com
        mail = ashleymallari@email.com
ashley@workstation:~/CPE232_AshleyMallari$ 
```

h. Edit the README.md file using nano command. Provide any information on the markdown file pertaining to the repository you created. Make sure to write out or save the file and exit.



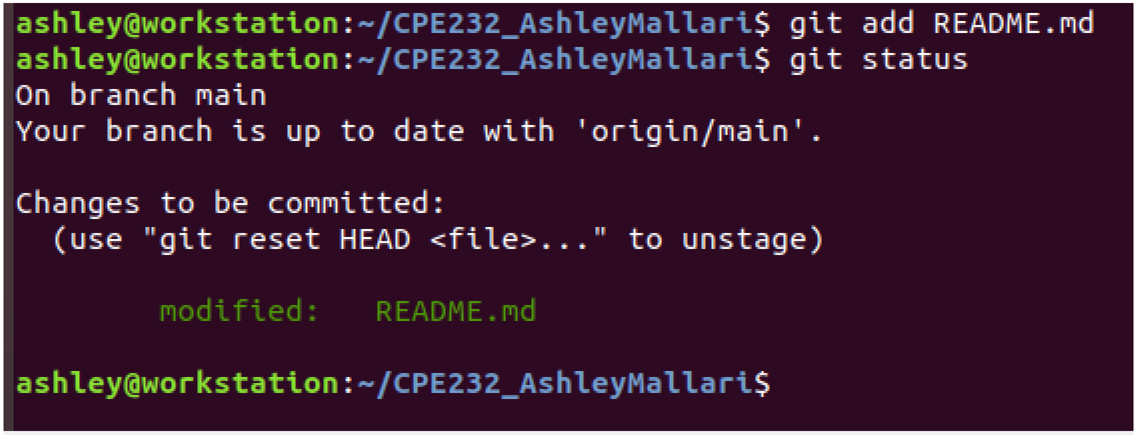
```
  GNU nano 2.9.3                          README.md

# CPE232_AshleyMallari
sysads6

Wave to Earth
```

i. Use the *git status* command to display the state of the working directory and the staging area. This command shows which changes have been staged, which haven't, and which files aren't being tracked by Git. Status output does not show any information regarding the committed project history. What is the result of issuing this command?



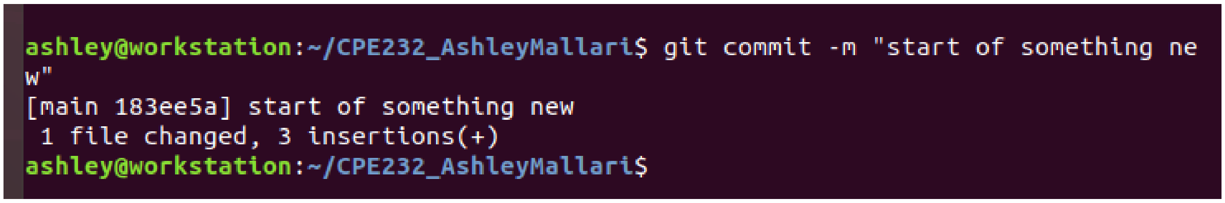
```
ashley@workstation:~/CPE232_AshleyMallari$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
ashley@workstation:~/CPE232_AshleyMallari$ 
```

j. Use the command *git add README.md* to add the file into the staging area.



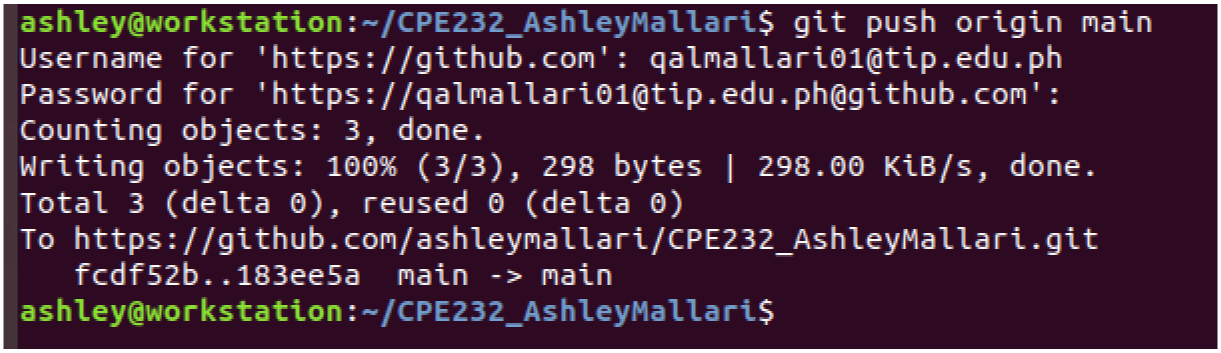
```
ashley@workstation:~/CPE232_AshleyMallari$ git add README.md
ashley@workstation:~/CPE232_AshleyMallari$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

ashley@workstation:~/CPE232_AshleyMallari$
```

k. Use the *git commit -m “your message”* to create a snapshot of the staged changes along the timeline of the Git projects history. The use of this command is required to select the changes that will be staged for the next commit.



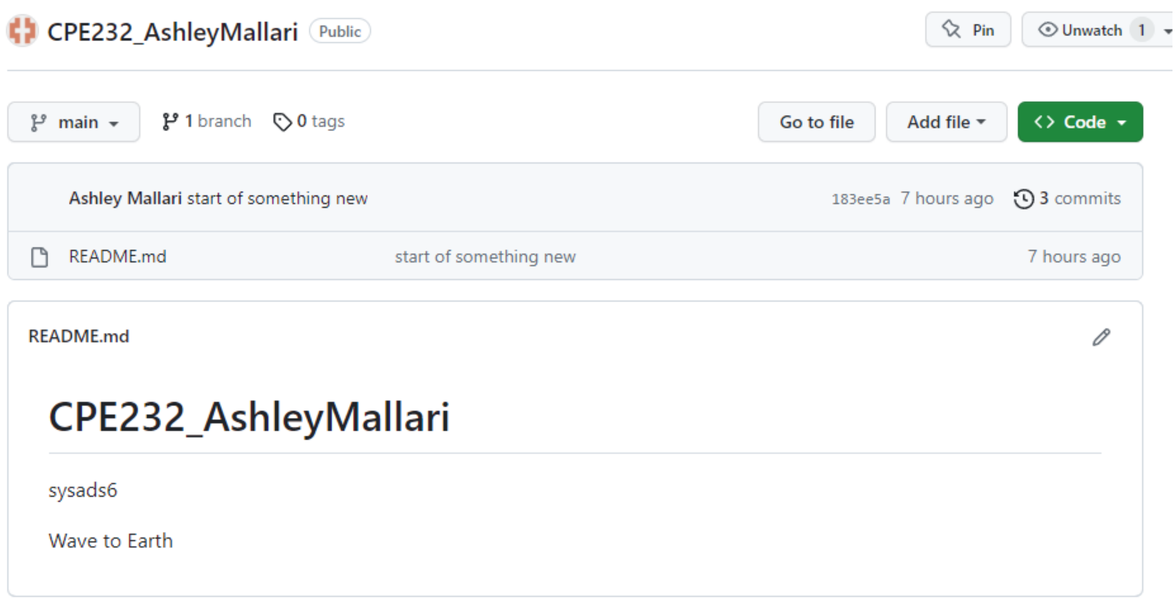
```
ashley@workstation:~/CPE232_AshleyMallari$ git commit -m "start of something ne
w"
[main 183ee5a] start of something new
 1 file changed, 3 insertions(+)
ashley@workstation:~/CPE232_AshleyMallari$
```

l. Use the command *git push <remote><branch>* to upload the local repository content to GitHub repository. Pushing means to transfer commits from the local repository to the remote repository. As an example, you may issue *git push origin main.*



```
ashley@workstation:~/CPE232_AshleyMallari$ git push origin main
Username for 'https://github.com': qalmallari01@tip.edu.ph
Password for 'https://qalmallari01@tip.edu.ph@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 298 bytes | 298.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ashleymallari/CPE232_AshleyMallari.git
   fcdf52b..183ee5a  main -> main
ashley@workstation:~/CPE232_AshleyMallari$
```

m. On the GitHub repository, verify that the changes have been made to README.md by refreshing the page. Describe the README.md file. You can notice the how long was the last commit. It should be some minutes ago and the message you typed on the git commit command should be there. Also, the README.md file should have been edited according to the text you wrote.



Ashley Mallari start of something new 183ee5a 7 hours ago 3 commits

README.md start of something new 7 hours ago

README.md

CPE232_AshleyMallari

sysads6

Wave to Earth

Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
   - Ansible playbooks and logs used to determine the actions taken on your remote servers. Common tasks performed with Ansible include software installation, configuration management, user management, service management, and file transfer. The specific tasks you've executed using Ansible would depend on your infrastructure and the goals of your automation efforts.

4. How important is the inventory file?
   - The inventory file is pivotal in Ansible as it defines the scope, organization, and configuration of the target hosts and plays a vital role in streamlining automation, improving organization, and ensuring proper access control in your infrastructure management tasks.

Conclusions/Learnings:

- I learned a lot in this activity, beginning with creating an ssh public and private key to gain access to and control other computers remotely. The procedure of setting up git repositories and remote repositories is a little tough for me because I repeated the process when I came home and was confused because the instructions in the provided document were unclear but I have taken pictures of the lectures that our professor taught us. That is why I accomplished the activity accurately according to the given questions and instructions. Therefore, I had a great time performing this exercise, and I hope to learn more about this kind of activity.