Name: Daniela Marie D. Rabang	**Date Performed:** 08/22/2023
Course/Section: CPE232/CPE31S4	**Date Submitted:** 08/29/2023
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.

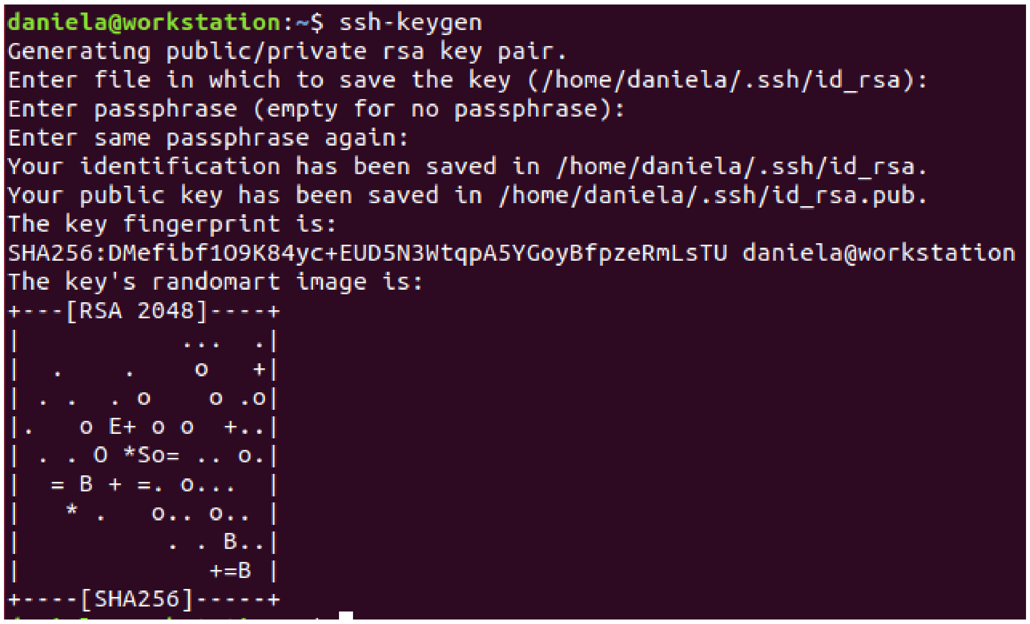
SSH introduced public key authentication as a more secure alternative to the older .rhosts authentication. It improved security by avoiding the need to have password stored in files and eliminated the possibility of a compromised server stealing the user's password.

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

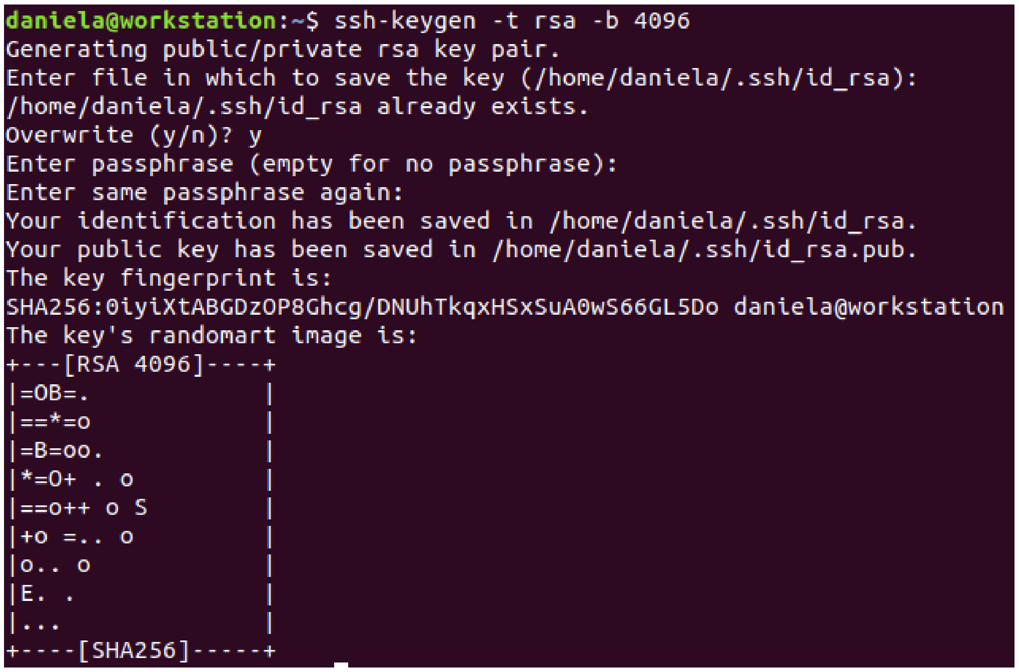
1. The simplest way to generate a key pair is to run *ssh-keygen* without arguments. In this case, it will prompt for the file in which to store keys. First,

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*.



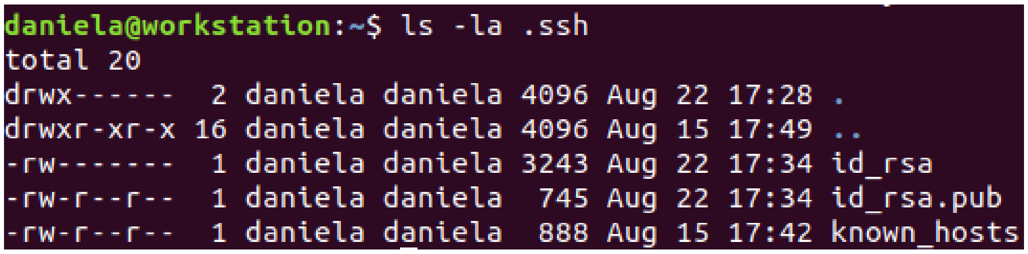
```
daniela@workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/daniela/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/daniela/.ssh/id_rsa.
Your public key has been saved in /home/daniela/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:DMefibf1O9K84yc+EUD5N3WtqpA5YGoyBfpzeRmLsTU daniela@workstation
The key's randomart image is:
+---[RSA 2048]----+
|           ...  .|
|  .    .    o   +|
| . .  . o    o .o|
|.   o E+ o o  +..|
| . . O *So= .. o.|
|  = B + =. o...  |
|   * .   o.. o.. |
|          . . B..|
|             +=B |
+----[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.
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.



```
daniela@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/daniela/.ssh/id_rsa):
/home/daniela/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/daniela/.ssh/id_rsa.
Your public key has been saved in /home/daniela/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:0iyiXtABGDzOP8Ghcg/DNUhTkqxHSxSuA0wS66GL5Do daniela@workstation
The key's randomart image is:
+---[RSA 4096]----+
|=OB=.            |
|==*=o            |
|=B=oo.           |
|*=O+ . o         |
|==o++ o S        |
|+o =.. o         |
|o.. o            |
|E. .             |
|...              |
+----[SHA256]-----+
```

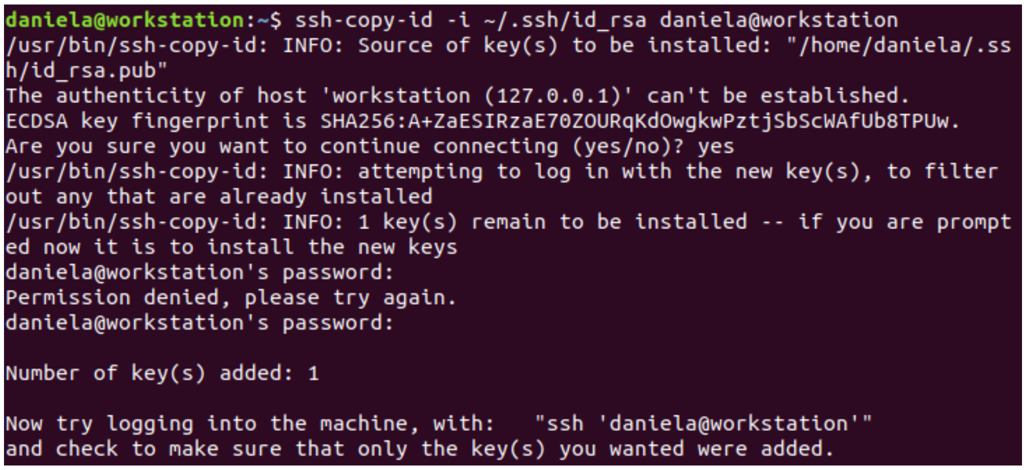
4. Verify that you have created the key by issuing the command *ls -la .ssh.* The command should show the .ssh directory containing a pair of keys. For example, id_rsa.pub and id_rsa.



```
daniela@workstation:~$ ls -la .ssh
total 20
drwx------  2 daniela daniela 4096 Aug 22 17:28 .
drwxr-xr-x 16 daniela daniela 4096 Aug 15 17:49 ..
-rw-------  1 daniela daniela 3243 Aug 22 17:34 id_rsa
-rw-r--r--  1 daniela daniela  745 Aug 22 17:34 id_rsa.pub
-rw-r--r--  1 daniela daniela  888 Aug 15 17:42 known_hosts
```

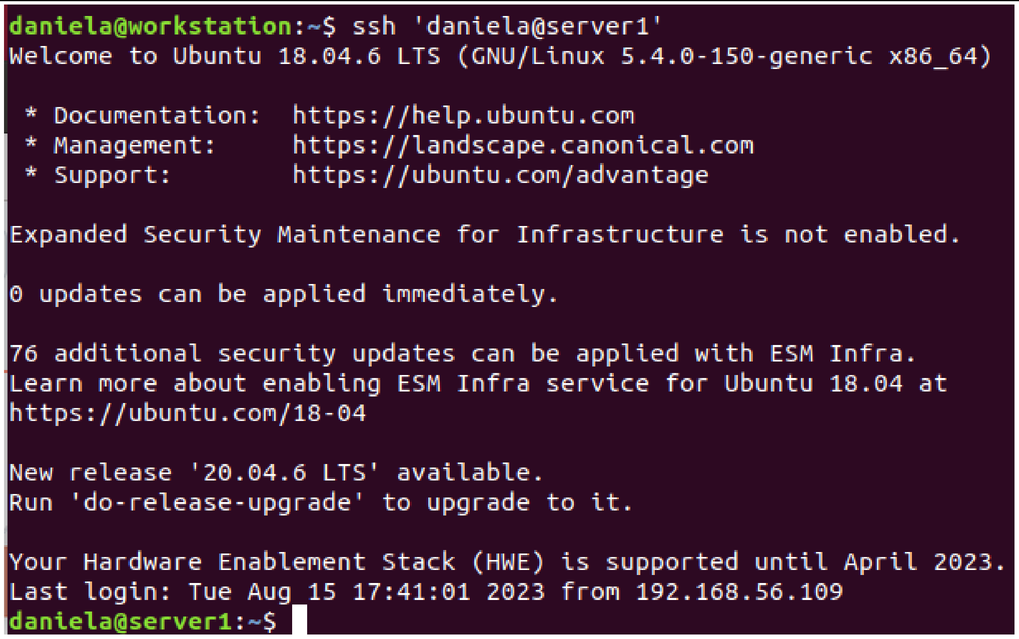
Task 2: Copying the Public Key to the remote servers

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*

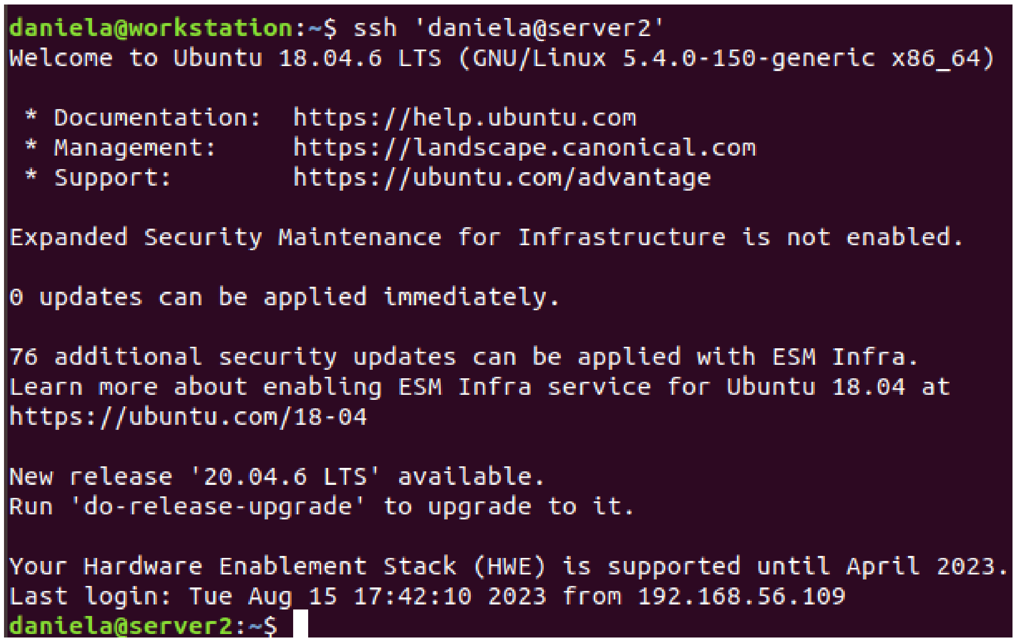


```
daniela@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa daniela@workstation
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/daniela/.ss
h/id_rsa.pub"
The authenticity of host 'workstation (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:A+ZaESIRzaE70ZOURqKdOwgkwPztjSbScWAfUb8TPUw.
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 prompt
ed now it is to install the new keys
daniela@workstation's password:
Permission denied, please try again.
daniela@workstation's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'daniela@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.



```
daniela@workstation:~$ ssh 'daniela@server1'
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

76 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Tue Aug 15 17:41:01 2023 from 192.168.56.109
daniela@server1:~$
```



```
daniela@workstation:~$ ssh 'daniela@server2'
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-150-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Infrastructure is not enabled.

0 updates can be applied immediately.

76 additional security updates can be applied with ESM Infra.
Learn more about enabling ESM Infra service for Ubuntu 18.04 at
https://ubuntu.com/18-04

New release '20.04.6 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2023.
Last login: Tue Aug 15 17:42:10 2023 from 192.168.56.109
daniela@server2:~$
```

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?

   I noticed that server 1 and server 2 can be accessed through the workstation. It does not ask for a password since I already inputted the password once.

-

Reflections:
Answer the following:
1. How will you describe the ssh-program? What does it do?
   - **So the ssh-program is used to give security. This gives the user a secured way to access an unsecured network.**

2. How do you know that you already installed the public key to the remote servers?
    - **You will know that you already installed the public key to the remote servers when you can already access the program through the use of your public key.**

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**).

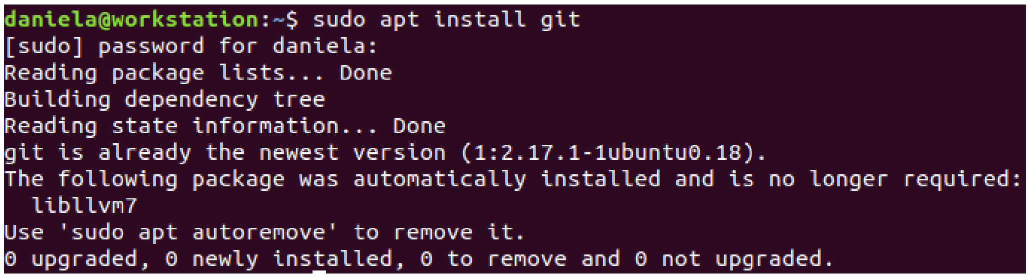
Set up Git

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

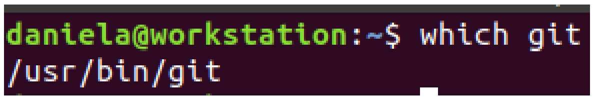
Task 3: Set up the Git Repository

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*



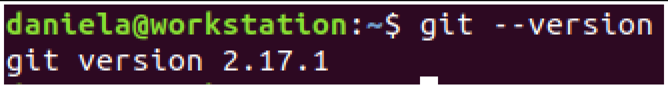
```
daniela@workstation:~$ sudo apt install git
[sudo] password for daniela:
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.17.1-1ubuntu0.18).
The following package was automatically installed and is no longer required:
  libllvm7
Use 'sudo apt autoremove' to remove it.
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*.



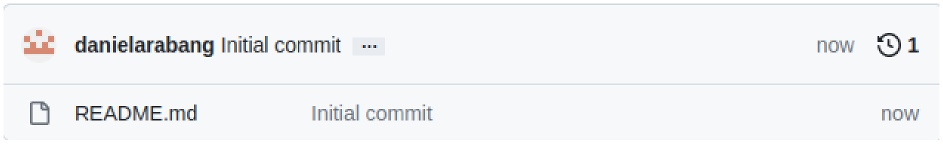
```
daniela@workstation:~$ which git
/usr/bin/git
```

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



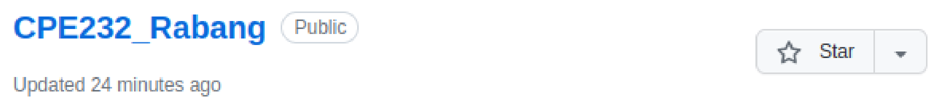
```
daniela@workstation:~$ git --version
git version 2.17.1
```

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.



danielarabang Initial commit ··· now 1

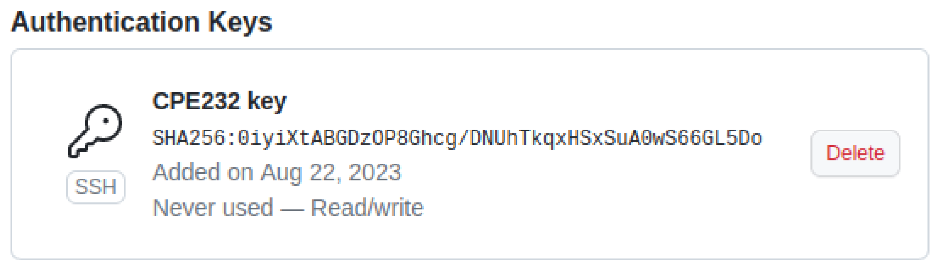
README.md Initial commit now



CPE232_Rabang Public

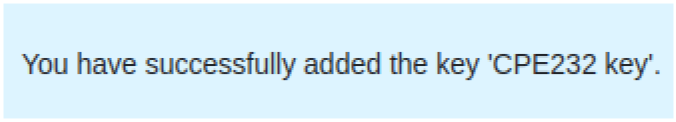
Updated 24 minutes ago

   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.



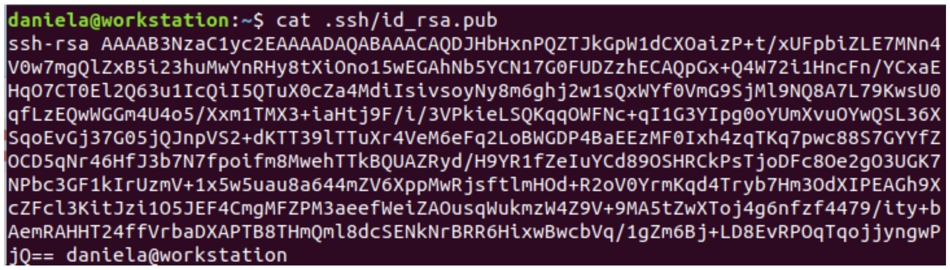
Authentication Keys

CPE232 key
SHA256:0iyiXtABGDzOP8Ghcg/DNUhTkqxHSxSuA0wS66GL5Do
Added on Aug 22, 2023
Never used — Read/write



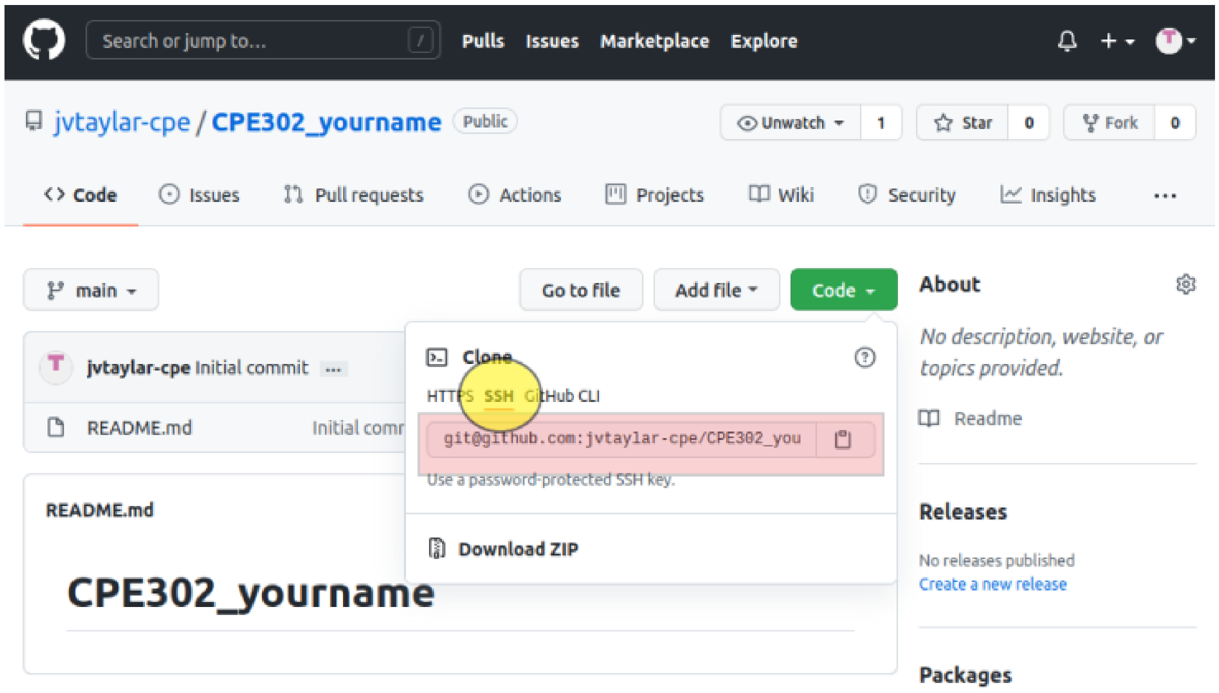
You have successfully added the key 'CPE232 key'.

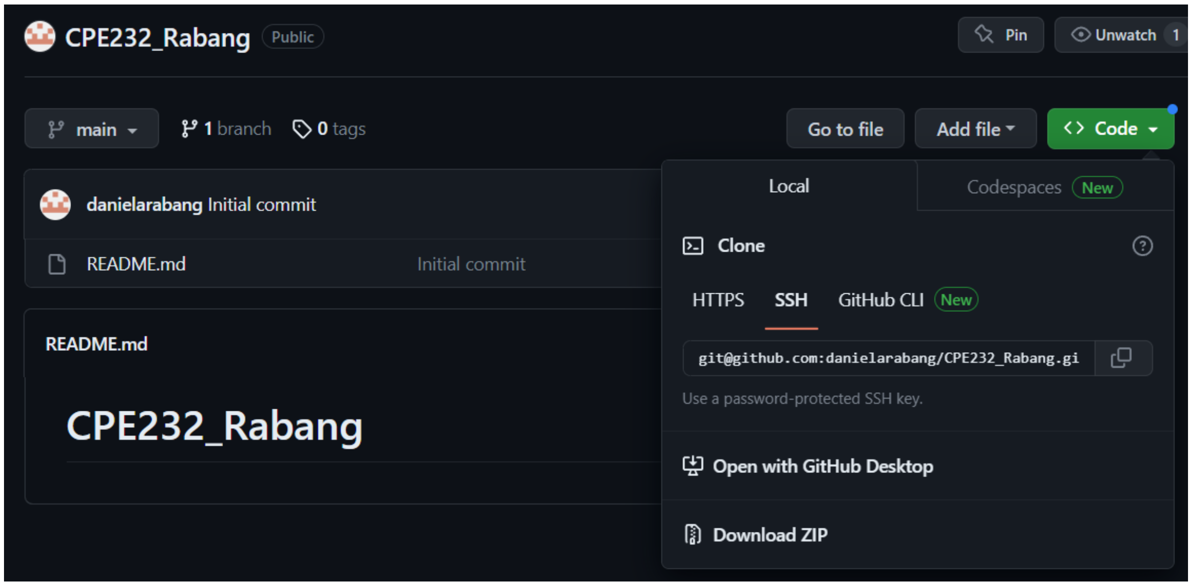
   c. On the local machine's terminal, issue the command cat .ssh/id_rsa.pub and copy the public key. Paste it on the GitHub key and press Add SSH key.



```
daniela@workstation:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDJHbHxnPQZTJkGpW1dCXOaizP+t/xUFpbiZLE7MNn4
V0w7mgQlZxB5i23huMwYnRHy8tXiOno15wEGAhNb5YCN17G0FUDZzhECAQpGx+Q4W72i1HncFn/YCxaE
HqO7CT0El2Q63u1IcQiI5QTuX0cZa4MdiIsivsoyNy8m6ghj2w1sQxWYf0VmG9SjMl9NQ8A7L79KwsU0
qfLzEQwWGGm4U4o5/Xxm1TMX3+iaHtj9F/i/3VPkieLSQKqqOWFNc+qI1G3YIpg0oYUmXvuOYwQSL36X
SqoEvGj37G05jQJnpVS2+dKTT39lTTuXr4VeM6eFq2LoBWGDP4BaEEzMF0Ixh4zqTKq7pwc88S7GYYfZ
OCD5qNr46HfJ3b7N7fpoifm8MwehTTkBQUAZRyd/H9YR1fZeIuYCd89OSHRCkPsTjoDFc8Oe2gO3UGK7
NPbc3GF1kIrUzmV+1x5w5uau8a644mZV6XppMwRjsftlmHOd+R2oV0YrmKqd4Tryb7Hm3OdXIPEAGh9X
cZFcl3KitJzi1O5JEF4CmgMFZPM3aeefWeiZAOusqWukmzW4Z9V+9MA5tZwXToj4g6nfzf4479/ity+b
AemRAHHT24ffVrbaDXAPTB8THmQml8dcSENkNrBRR6HixwBwcbVq/1gZm6Bj+LD8EvRPOqTqojjyngwP
jQ== daniela@workstation
```

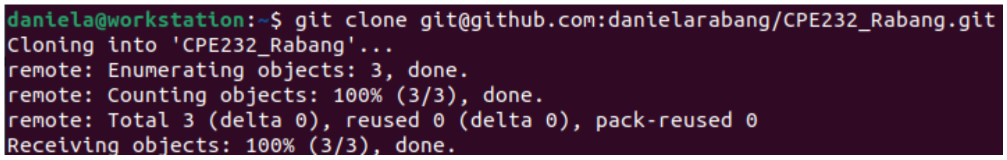
   d. Clone the repository that you created. In doing this, you need to get the link from GitHub. Browse to your repository as shown below. Click on the Code drop down menu. Select SSH and copy the link.





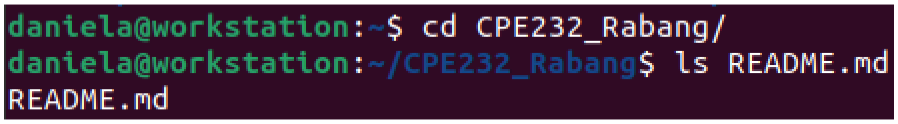
git@github.com:danielarabang/CPE232_Rabang.git

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.



```
daniela@workstation:~$ git clone git@github.com:danielarabang/CPE232_Rabang.git
Cloning into 'CPE232_Rabang'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

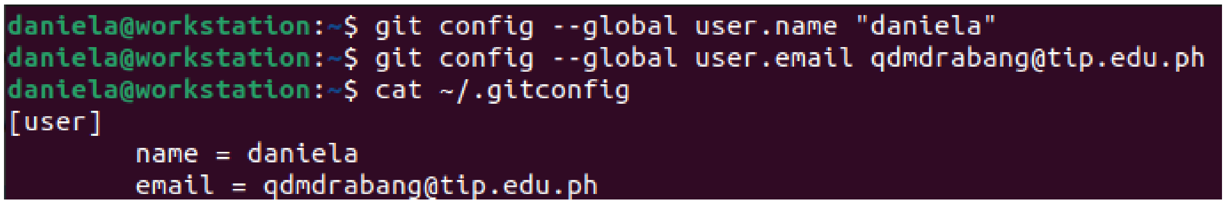
f. sTo verify that you have cloned the GitHub repository, issue the command *ls*. Observe that you have the CPE232_yourname in the list of your directories. Use CD command to go to that directory and LS command to see the file README.md.



```
daniela@workstation:~$ cd CPE232_Rabang/
daniela@workstation:~/CPE232_Rabang$ ls README.md
README.md
```

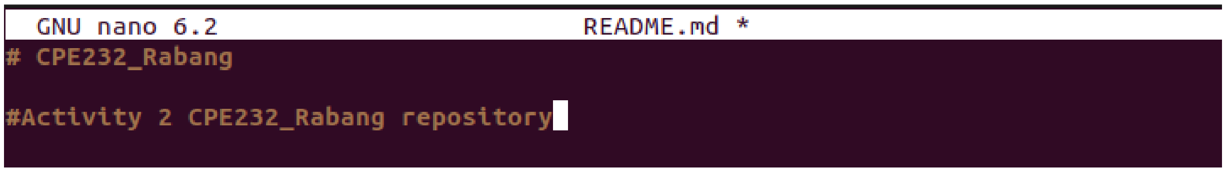
g. Use the following commands to personalize your git.

- *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*



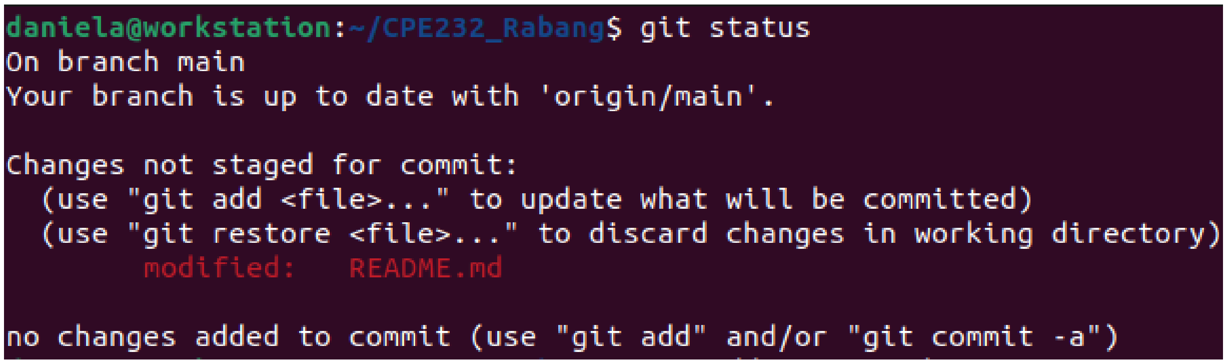
```
daniela@workstation:~$ git config --global user.name "daniela"
daniela@workstation:~$ git config --global user.email qdmdrabang@tip.edu.ph
daniela@workstation:~$ cat ~/.gitconfig
[user]
        name = daniela
        email = qdmdrabang@tip.edu.ph
```

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.



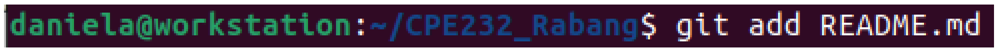
```
  GNU nano 6.2                          README.md *
# CPE232_Rabang

#Activity 2 CPE232_Rabang repository
```

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?



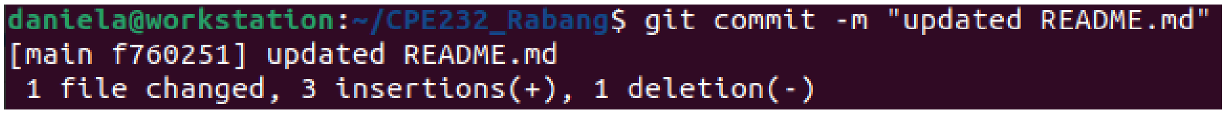
```
daniela@workstation:~/CPE232_Rabang$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

no changes added to commit (use "git add" and/or "git commit -a")
```

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



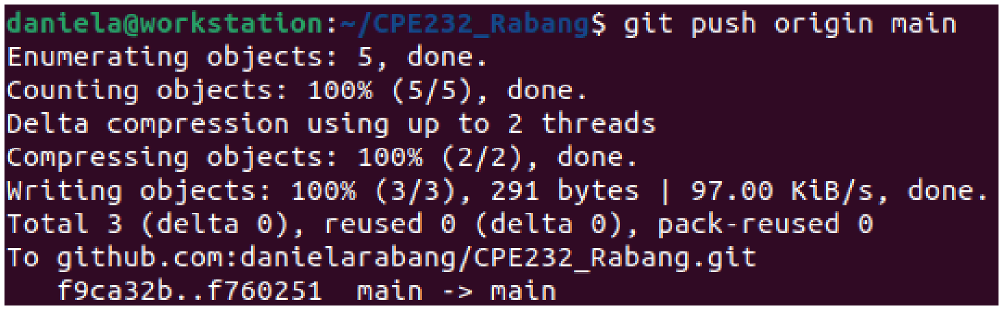
```
daniela@workstation:~/CPE232_Rabang$ git add README.md
```

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.



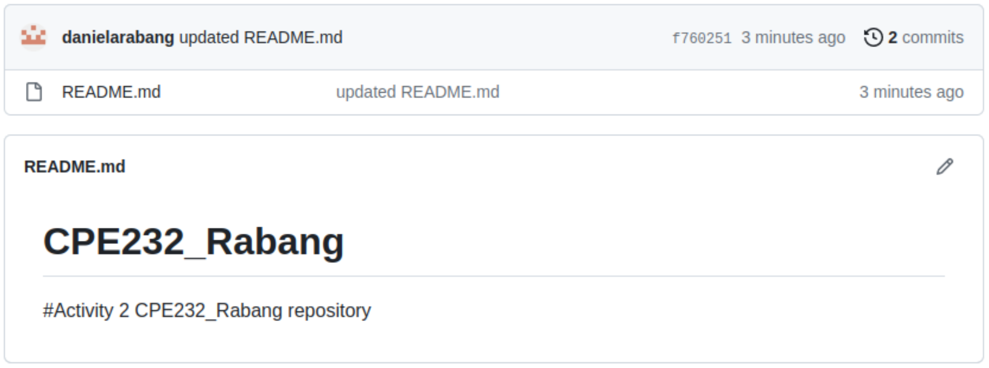
```
daniela@workstation:~/CPE232_Rabang$ git commit -m "updated README.md"
[main f760251] updated README.md
 1 file changed, 3 insertions(+), 1 deletion(-)
```

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.*



```
daniela@workstation:~/CPE232_Rabang$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 291 bytes | 97.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:danielarabang/CPE232_Rabang.git
   f9ca32b..f760251  main -> main
```

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.



danielarabang updated README.md f760251 3 minutes ago 2 commits

README.md updated README.md 3 minutes ago

README.md

CPE232_Rabang

#Activity 2 CPE232_Rabang repository

Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
   - We had to create, update, and commit with the use of ansible commands.

4. How important is the inventory file?

- The inventory file is important since it is where all the files are stored.

Conclusions/Learnings:

This hands-on activity we had done the SSH Key-Based Authentication and git setup. The said manual gave us an idea on how we will do the laboratory work. This is by the use of step by step procedure. The first thing that we had done is to create an ssh key pair that we will use for user authentication this command is done with the use of the ssh-keygen and ssh-keygen -t rsa -b 4096 command. After creating an ssh key pair we ensure that we had it by entering the command ls -la .ssh. Then we had copied the public key to the remote server and tried to access the two servers on the workstation. On the second part, we had set up the git repository by creating a account then adding a new repository. then we had installed git through the use of the terminal. The version and the directory of the git is also found by using the command which git and the git --version. Then we generate a ssh key on the terminal and use the generated key to put on your git account where you can create a key that can access the environment. Then we had cloned the repository by the use of other commands that is also given in the procedure. We also done the git status, git add, git commit with a comment, and git push to save the update.