

Name: Taller, Rose Ann A.	Date Performed: August 22, 2023
Course/Section: CPE31S4	Date Submitted: August 28, 2023
Instructor: Dr. Jonathan V. Taylor	Semester and SY: First Sem, 2023-2024
Activity 2: SSH Key-Based Authentication and Setting up Git	
1. Objectives: <ul style="list-style-type: none"> 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 <p>It is assumed that you are already done with the last Activity (Activity 1: Configure Network using Virtual Machines). <i>Provide screenshots for each task.</i></p> <p>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.</p> <p>What is ssh-keygen?</p> <p>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.</p> <p>SSH Keys and Public Key Authentication</p> <p>The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.</p> <p>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.</p> <p>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.</p>	
Task 1: Create an SSH Key Pair for User Authentication <ul style="list-style-type: none"> 1. The simplest way to generate a key pair is to run <i>ssh-keygen</i> 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`.

```
rose@workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/rose/.ssh/id_rsa):
/home/rose/.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/rose/.ssh/id_rsa.
Your public key has been saved in /home/rose/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:lmPCONNl9rs+yZCKOHok3IK1XA2GTFXLwLU5DnAVsUk rose@workstation
The key's randomart image is:
+---[RSA 2048]-----+
| oo+++Eo          |
| oo++ *           |
| ..o%             |
| .o.*.o .         |
| oo=oo ..S        |
| oo+= oo .        |
| +...oo..         |
| o o .o +.        |
| oo .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.

```
rose@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/rose/.ssh/id_rsa):
/home/rose/.ssh/id_rsa already exists.
Overwrite (y/n)? y
```

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.

```

rose@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/rose/.ssh/id_rsa):
/home/rose/.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/rose/.ssh/id_rsa.
Your public key has been saved in /home/rose/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:lh5k0CdluDioB+iZQFiWgmHpAaRWsUViLysb7iP/Zgw rose@workstation
The key's randomart image is:
+---[RSA 4096]-----+
|B=+*oo .o          |
|*+= = .+           |
|=0.+ 0+.+          |
|=. = .*            |
|o=0. . S           |
|o+E  o .           |
| + o .             |
|o. +               |
|.oo+.              |
+----[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`.

```

rose@workstation:~$ ls -la .ssh
total 20
drwx----- 2 rose rose 4096 Aug 22 17:40 .
drwxr-xr-x 15 rose rose 4096 Aug 22 17:43 ..
-rw----- 1 rose rose 3243 Aug 22 17:51 id_rsa
-rw-r--r-- 1 rose rose 742 Aug 22 17:51 id_rsa.pub
-rw-r--r-- 1 rose rose 888 Aug 22 17:38 known_hosts
rose@workstation:~$

```

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`

```

rose@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa rose@server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/rose/.ssh/
id_rsa.pub"
/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
rose@server1's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'rose@server1'"
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.

```
rose@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa rose@server2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/rose/.ssh/
id_rsa.pub"
/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
rose@server2's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'rose@server2'"
and check to make sure that only the key(s) you wanted were added.

rose@workstation:~$
```

4. On the local machine, verify that you can SSH with Server 1 and Server 2.

```
rose@server1: ~
File Edit View Search Terminal Help
rose@workstation:~$ ssh rose@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.

85 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 22 17:38:06 2023 from 192.168.56.105
rose@server1:~$
```

```
rose@workstation:~$ ssh rose@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.

85 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 22 17:38:19 2023 from 192.168.56.105
rose@server2:~$
```

What did you notice? Did the connection ask for a password? If not, why?

- ***It's directly access the servers and doesn't require for password since we configure the public key which results to a successful key-based authentication.***

Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
 - ***It is a secure network protocol program that enables encrypted communication to avoid unauthorized access between devices. It also secures the process of file transferring and tunneling.***
2. How do you know that you already installed the public key to the remote servers?
 - ***When we successfully log in even without entering a password to access an SSH connection since they use the previously installed public key.***

Part 2: Discussion

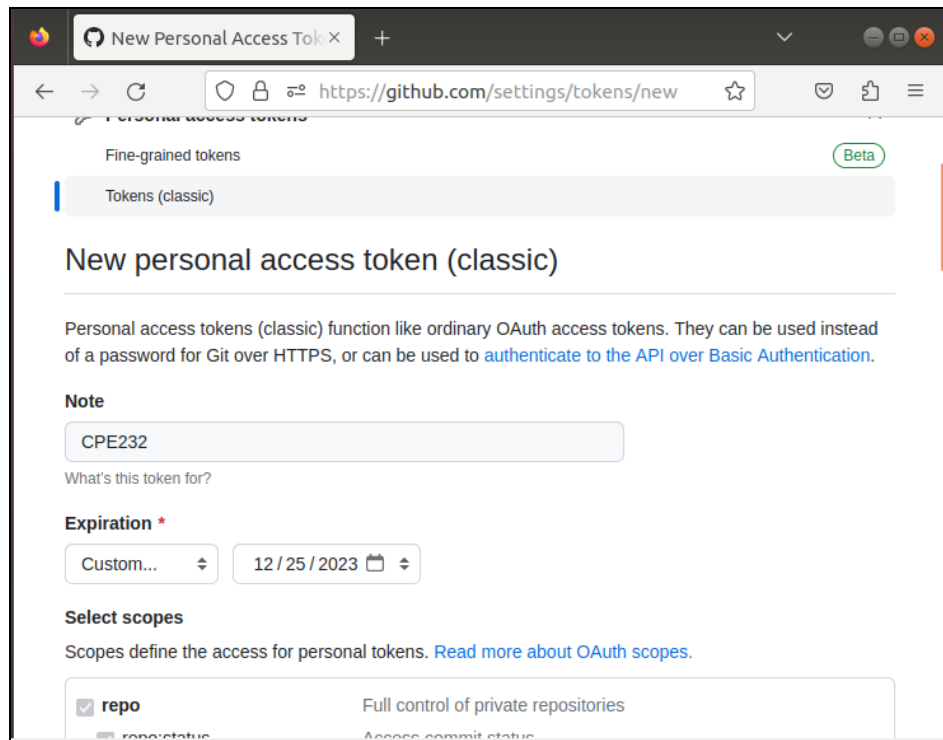
Provide screenshots for each task.

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



The screenshot shows the GitHub web interface for creating a new classic personal access token. The browser address bar shows the URL `https://github.com/settings/tokens/new`. The page has a sidebar with 'Fine-grained tokens' (marked as Beta) and 'Tokens (classic)' selected. The main content area is titled 'New personal access token (classic)'. It includes a description: 'Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).' Below this is a 'Note' section with a text input field containing 'CPE232'. A section for 'Expiration' shows a dropdown set to 'Custom...' and a date picker set to '12 / 25 / 2023'. The 'Select scopes' section explains that scopes define access and provides a link to 'Read more about OAuth scopes'. A table of scopes is visible, with 'repo' selected, granting 'Full control of private repositories'.

Personal access tokens

Fine-grained tokens Beta

Tokens (classic)

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

CPE232

What's this token for?

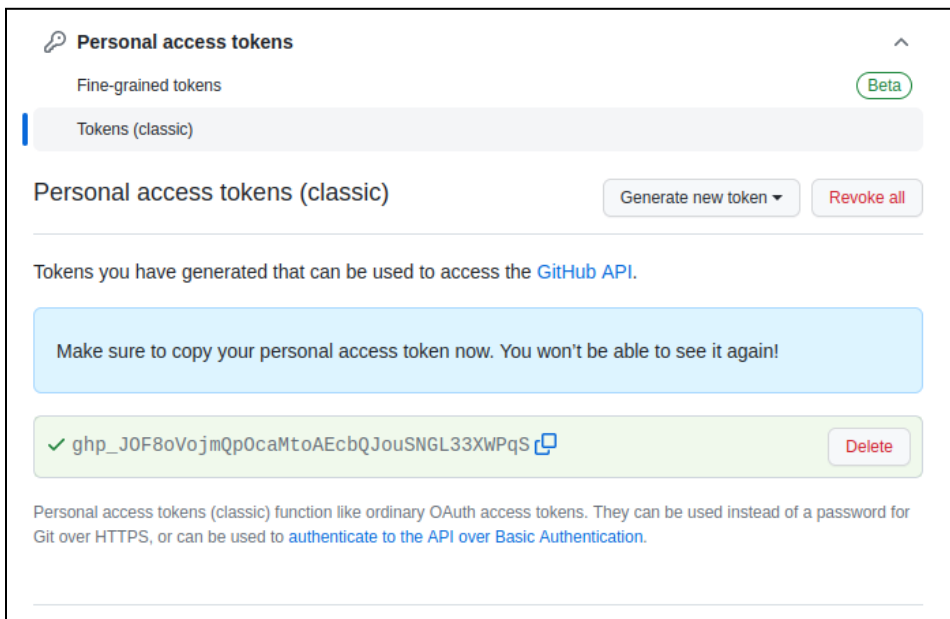
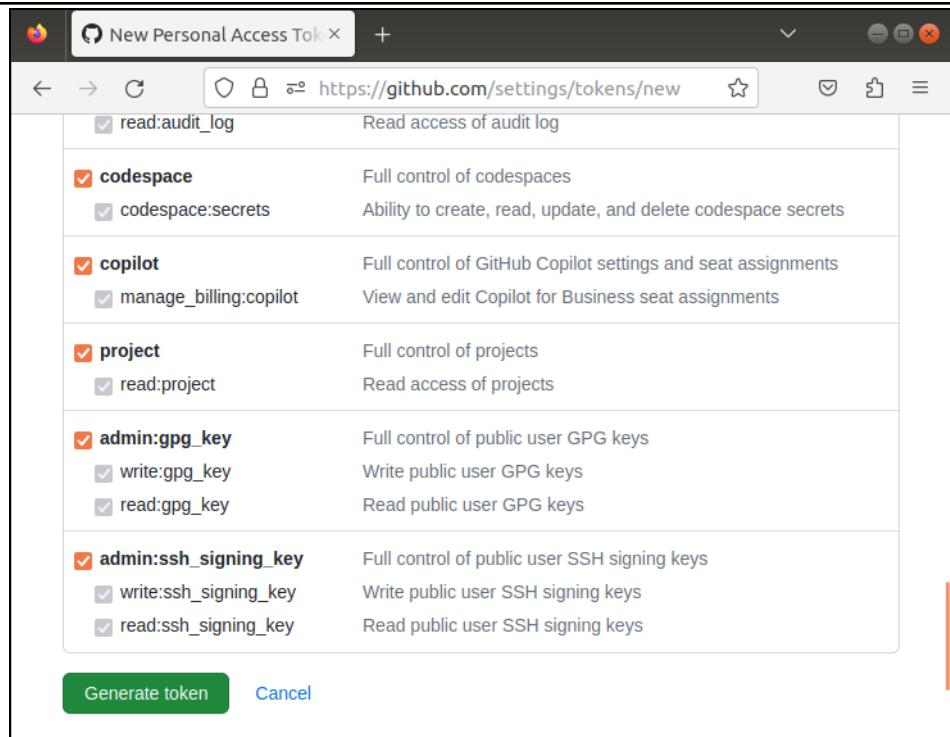
Expiration *

Custom... 12 / 25 / 2023

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repository:status	Access commit status



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

- 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.
 - 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.

SSH keys

[New SSH key](#)

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

Authentication Keys



CPE232

SHA256:1h5k0Cd1uDioB+izQFiWgmHpAaRwsUViLysb7iP/Zgw

Added on Aug 22, 2023

Never used — Read/write

SSH

Delete

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.

Search or jump to... Pulls Issues Marketplace Explore

jvtaylor-cpe / CPE302_yourname Public

Unwatch 1 Star 0 Fork 0

Code Issues Pull requests Actions Projects Wiki Security Insights

main

Go to file Add file Code

Clone

HTTPS SSH GitHub CLI

git@github.com:jvtaylor-cpe/CPE302_you

Use a password-protected SSH key.

Download ZIP

About

No description, website, or topics provided.

Readme

Releases

No releases published

Create a new release

Packages

CPE302_yourname

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

```
rose@workstation: ~  
File Edit View Search Terminal Help  
rose@workstation:~$ git clone git@github.com:qrattaler/CPE232_TALLER.git  
Cloning into 'CPE232_TALLER'...  
The authenticity of host 'github.com (20.205.243.166)' can't be established.  
ECDSA key fingerprint is SHA256:p2QAMXNIC1TJYWeI0ttrVc98/R1BUFWu3/LiyKgUfQM.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'github.com,20.205.243.166' (ECDSA) to the list of k  
nown hosts.  
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.  
rose@workstation:~$
```

- 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`.

```
rose@workstation:~$ ls  
CPE232_TALLER  Documents  examples.desktop  Pictures  Templates  
Desktop        Downloads  Music             Public    Videos  
rose@workstation:~$ cd CPE232_TALLER  
rose@workstation:~/CPE232_TALLER$ ls  
README.md  
rose@workstation:~/CPE232_TALLER$
```

- 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`

```
rose@workstation:~/CPE232_TALLER$ git config --global user.email qrattaler@tip.edu.ph  
rose@workstation:~/CPE232_TALLER$ git config --global user.name "Rose Ann"  
rose@workstation:~/CPE232_TALLER$ cat ~/.gitconfig  
[user]  
  name = Rose Ann  
  email = qrattaler@tip.edu.ph  
rose@workstation:~/CPE232_TALLER$
```

- 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.

```
rose@workstation: ~/CPE232_TALLER
File Edit View Search Terminal Help
GNU nano 2.9.3 README.md

# CPE232_TALLER
CPE31S4

[ Wrote 3 lines ]
^G Get Help ^O Write Out ^W Where Is [ Wrote 3 lines ] ^J Justify ^C Cur Pos ^U Undo
^X Exit ^R Read File ^\ Replace ^M Cut Text ^_ Uncut Text ^T To Spell ^G Go To Line ^E Redo
```

- 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?

```
rose@workstation:~/CPE232_TALLER$ sudo nano README.md
[sudo] password for rose:
rose@workstation:~/CPE232_TALLER$ 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")
rose@workstation:~/CPE232_TALLER$
```

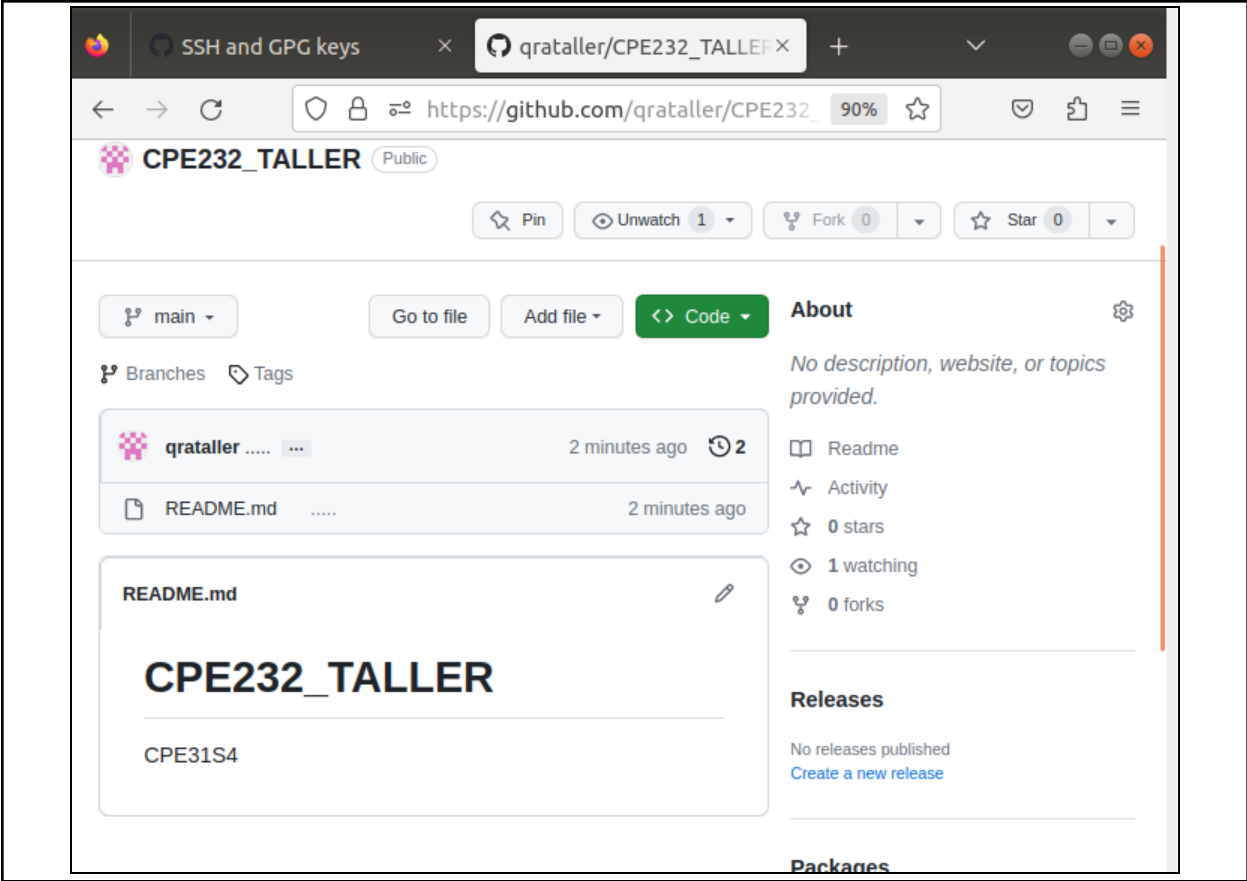
- j. Use the command *git add README.md* to add the file into the staging area.
- k. Use the *git commit -m "your message"* to create a snapshot of the staged changes along the timeline of the Git projects history. The use of this command is required to select the changes that will be staged for the next commit.

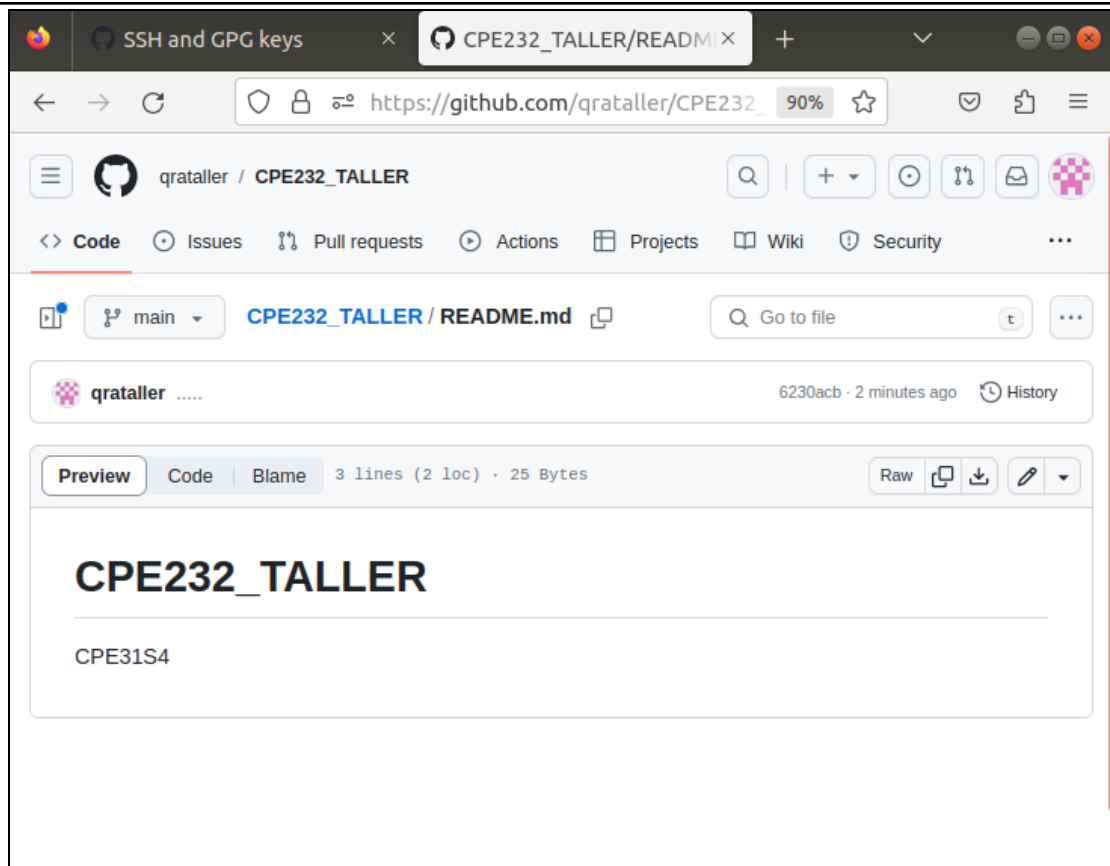
```
rose@workstation:~/CPE232_TALLER$ git add README.md
rose@workstation:~/CPE232_TALLER$ git commit -m "....."
[main 6230acb] .....
1 file changed, 3 insertions(+), 1 deletion(-)
```

- I. 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*.

```
rose@workstation:~/CPE232_TALLER$ git push origin main
Counting objects: 3, done.
Writing objects: 100% (3/3), 259 bytes | 259.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:qrattler/CPE232_TALLER.git
837bed0..6230acb main -> main
rose@workstation:~/CPE232_TALLER$
```

- 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.





Reflections:

Answer the following:

3. What sort of things have we so far done to the remote servers using ansible commands?
 - ***We already***
 - ***Created a key pair for SSH authentication,***
 - ***Cloned public key to remote servers,***
 - ***Set up git repository using local and remote repositories, and***
 - ***Configured and executed ad hoc commands from local machine to remote servers***
4. How important is the inventory file?
 - ***The inventory file is important since it lists all remote hosts, organizes server information, and enables efficient management. It also gives a detailed record of how and when a file is modified.***

Conclusions/Learnings:

- *In this activity, I've learned how to set up secure connections between local machines and remote servers using keys instead of passwords. Having knowledge about this topic can help in ensuring security, enhancing automation and efficiency, and remote management. This activity also wants to emphasize the cruciality of security when it comes to server management or system administration.*

Honor Pledge for Graded Activity

"I affirm that I shall not give and receive any unauthorized help on this activity, and that this work is my own."