

|   |  |
|---|--|
| <b>Name:</b> Zamora, Angelo E.            | <b>Date Performed:</b> 09 - 04 - 2024            |
| <b>Course/Section:</b> CPE31S2            | <b>Date Submitted:</b>                           |
| <b>Instructor:</b> Engr. Robin Valenzuela | <b>Semester and SY:</b> 1st Semester 2024 - 2025 |

### **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**

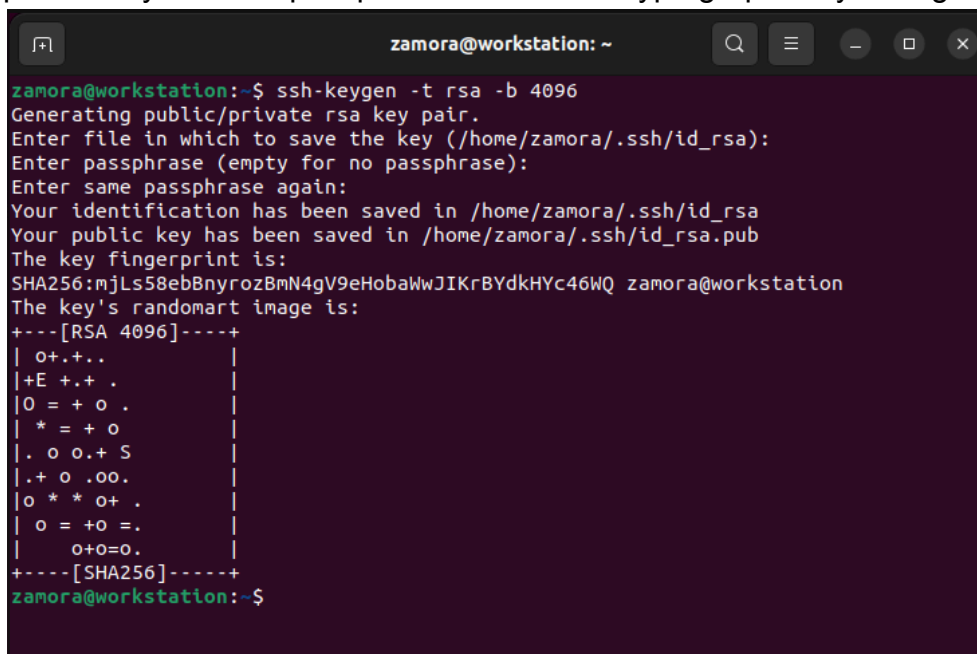
The SSH protocol uses public key cryptography for authenticating hosts and users. The authentication keys, called SSH keys, are created using the keygen program.

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.

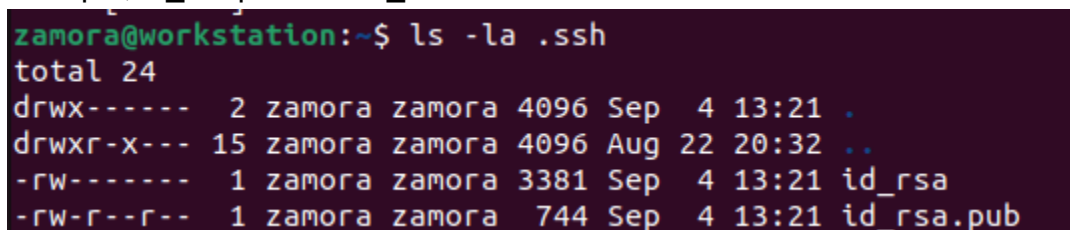
### Task 1: Create an SSH Key Pair for User Authentication

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`.
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.



```
zamora@workstation: ~  
zamora@workstation:~$ ssh-keygen -t rsa -b 4096  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/zamora/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/zamora/.ssh/id_rsa  
Your public key has been saved in /home/zamora/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:mjLs58ebBnyrozBmN4gV9eHobaWwJIKrBYdkHYc46WQ zamora@workstation  
The key's randomart image is:  
+----[RSA 4096]-----+  
|  o+...      |  
|+E +.+. .   |  
|O = + o .   |  
| * = + o     |  
|. o o.+ S    |  
|. + o .oo.   |  
|o * * o+ .   |  
| o = +o =.   |  
|   o+o=o.    |  
+----[SHA256]-----+  
zamora@workstation:~$
```

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`.



```
zamora@workstation:~$ ls -la .ssh  
total 24  
drwx----- 2 zamora zamora 4096 Sep  4 13:21 .  
drwxr-x--- 15 zamora zamora 4096 Aug 22 20:32 ..  
-rw----- 1 zamora zamora 3381 Sep  4 13:21 id_rsa  
-rw-r--r-- 1 zamora zamora  744 Sep  4 13:21 id_rsa.pub
```

## 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*
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? **It logs straight to the server.** Did the connection ask for a password? **It didn't ask for a password.** If not, why? **Public key serves as a key for recognizing the workstation to server 1 and 2 as a safe connection. It's like adding a machine to the list that the workstation can control or access through the Public Key Authorization.**

### Server 1 Copying Public Key:

```
zamora@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa zamora@server1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zamora/.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 prompted now it is to install the new keys
zamora@server1's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'zamora@server1'"
and check to make sure that only the key(s) you wanted were added.
zamora@workstation:~$
```

### Server 2 Copying Public Key:

```
zamora@workstation:~$ ssh-copy-id -i ~/.ssh/id_rsa zamora@server2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/zamora/.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 prompted now it is to install the new keys
zamora@server2's password:

Number of key(s) added: 1

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

### Testing Workstation to Server 1:

```
zamora@workstation:~$ ssh zamora@server1
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Thu Aug 22 20:35:09 2024 from 192.168.56.101
zamora@server1:~$
```

### Testing Workstation to Server 2:

```
zamora@workstation:~$ ssh zamora@server2
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.8.0-40-generic x86_64)

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

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Thu Aug 22 20:33:09 2024 from 192.168.56.101
zamora@server2:~$
```

### Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?

SSH (Secure Shell) is a network protocol that allows you to securely access remote computers. It is intended to protect communications by encrypting all data sent between the client and the server. This protects against illegal access and eavesdropping. What it does is that it gives you access to remote servers through Remote Login. Next it allows you to file transfers remotely through SSH, even Command Execution etc.

2. How do you know that you already installed the public key to the remote servers?  
To check, you can verify through the authorization file, check if the public key is in the file. Next is test the SSH connection to verify if you have access.

## 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*

```
zamora@workstation:~$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libwpe-1.0-1 libwpebackend-fdo-1.0-1
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 3 not upgraded.
Need to get 4,146 kB of archives.
After this operation, 21.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ph.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all 0.1
7029-1 [26.5 kB]
Get:2 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man all 1
:2.34.1-1ubuntu1.11 [955 kB]
Get:3 http://ph.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64 1:2
34.1-1ubuntu1.11 [4,146 kB]
```

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

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

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

```
zamora@workstation:~$ git --version
git version 2.34.1
zamora@workstation:~$
```

4. Using the browser in the local machine, go to [www.github.com](https://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.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

Required fields are marked with an asterisk (\*).


Owner \* Repository name \*


 GeloaceRT / CPE212\_ZAMORA\_Angelo

 CPE212\_ZAMORA\_Angelo is available.

Great repository names are short and memorable. Need inspiration? How about [super-duper-goggles](#) ?

Description (optional)

☒  **Public**  
Anyone on the Internet can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file

- 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

CPE212 Key

Key type

Authentication 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.

### Add new SSH Key

Title

CPE212 Key

Key type

Authentication Key

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQACQC3Wd4Y1fNyViGHqww4PXj+foaNB44yW9k3kfe+chd75l6gCeeHQiVRMQ2j3erohU4SnaVr
tTXskVTkoO2GXaDq8wroc6pUdJXICcsPscdKRPVK9DnzPHYqZs07PVHbH3OU1JqL3Rlqugg7v0RDYl2s65MCj2X1upHRQDAndg1Gj
TizBJ1C+zHu3u1esglWAploFsiilwLYDShyxUT3sR6FPA3Q5477ovSWLe02PkXprYIQxsBhFKLXhOuEYV5GEUK/92bwKdO0+Wp
/wOc3lVJBxNsuVfkAtSr4SZhABp22CM9NaR1HqabSycS3ThTL
/dOs1E6qdcfdj246FPnZqgPkibhy5mw0FHS1HzVRVCg1f6Vp4a0Y2qnUxaTPi3tPAeGdP1FQFMMUL7WB8ccVnk/30nSYsgeJoiWA
/BTl/2XgqqWwe6FMexj9FeV1uE7l95bPV3D5rtmH406RK53ESXD+XEeraKr4Ll4jtpex4l4h5f1yWOzzTA5dPFEHBBi8ZDF8oVofbNrOc
XQ3VvaNy0csPSsAxiHIEEmeRq+KuyyvXQk2CL06mbrbaEIH8R4uJoAAIL3K1OvMjMsb7WATUWf1PiSyluufV+5ccZTCnjsLgMLAeN
RsiDB8daaVo+F4iuZzo7xIW5NKqUcPGSxXx9OsF38tRPhZiYIMbEQ== zamora@workstation
```

Add SSH key

### Authentication keys



CPE212 Key

SHA256:mjLs58ebBnyrozBmM4gV9eHobaWwJIKRBYdkHYc46WQ

Added on Sep 4, 2024

Never used — Read/write

Delete

- 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

About

No description, website, or topics provided.

Readme

Releases

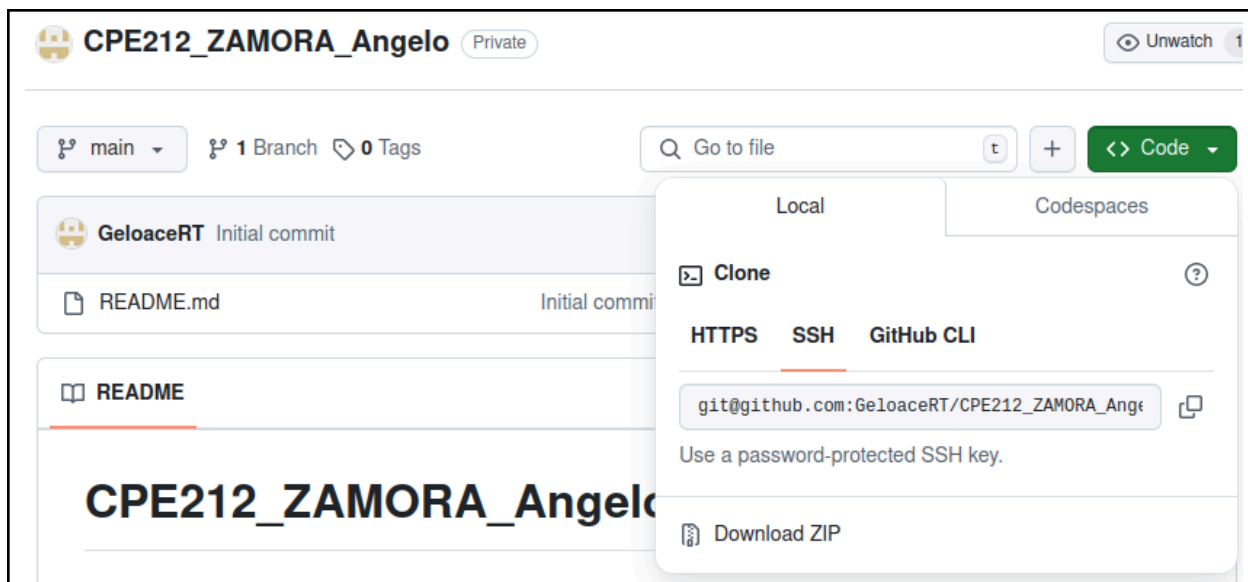
No releases published

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.

```
zamora@workstation:~$ git clone git@github.com:GeloaceRT/CPE212_ZAMORA_Angelo.git

zamora@workstation:~$ git clone git@github.com:GeloaceRT/CPE212_ZAMORA_Angelo.git
Cloning into 'CPE212_ZAMORA_Angelo'...
The authenticity of host 'github.com (20.205.243.166)' can't be established.
ED25519 key fingerprint is SHA256:+DIY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdKr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known 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 (from 0)
Receiving objects: 100% (3/3), done.
zamora@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`.

```
zamora@workstation:~$ ls
CPE212_ZAMORA_Angelo  Desktop  Documents  Downloa
zamora@workstation:~$ cd CPE212_ZAMORA_Angelo
zamora@workstation:~/CPE212_ZAMORA_Angelo$ ls
README.md
zamora@workstation:~/CPE212_ZAMORA_Angelo$
```

- 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`

```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git config --global user.name "Angelo Zamora"
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git config --global user.email angelozamora65@gmail.com
zamora@workstation:~/CPE212_ZAMORA_Angelo$ cat ~/.gitconfig
[user]
  name = Angelo Zamora
  email = angelozamora65@gmail.com
zamora@workstation:~/CPE212_ZAMORA_Angelo$
```

- 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.

```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ nano README.md
GNU nano 6.2
# CPE212_ZAMORA_Angelo
My name is Angelo Zamora and I created this 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?

```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ 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")
zamora@workstation:~/CPE212_ZAMORA_Angelo$
```

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

```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git add README.md
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git commit -m "I, Angelo Zamora created the repository."
```

- 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.

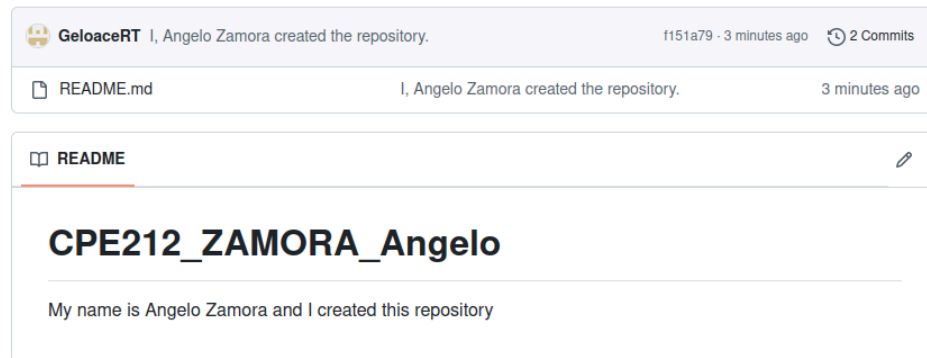
```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git add README.md
zamora@workstation:~/CPE212_ZAMORA_Angelo$ git commit -m "I, Angelo Zamora created the repository."
[main f151a79] I, Angelo Zamora created the repository.
1 file changed, 2 insertions(+), 1 deletion(-)
```

- 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*.

```
zamora@workstation:~/CPE212_ZAMORA_Angelo$ 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), 333 bytes | 333.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:GeloaceRT/CPE212_ZAMORA_Angelo.git
68f41bc..f151a79  main -> main
zamora@workstation:~/CPE212_ZAMORA_Angelo$
```

- 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? First we were able to create a public key to give access to the remote servers. Because whenever I establish an SSH Connection to the remote servers there will be no password to be prompted. With SSH, I have the access to work on the remote servers in just a single VM through the public key authorization access of SSH. Lastly, we were able to link a git repository in our very own VM, through the git commands to make it happen. We were able to modify, edit, and commit to the README.md through a terminal which is amazing.
4. How important is the inventory file?  
The inventory file is essential in Git repositories that use SSH for authentication. It serves as a centralized area for storing remote server IP addresses or host names. This simplifies the management and maintenance of connections to

numerous servers by eliminating the need to explicitly declare their details for each action.

**Conclusions/Learnings:**

This exercise successfully illustrated how to configure distant and local machines for a secure SSH connection utilizing public-key authentication, which eliminates the need for passwords. The development of a public and private key pair was a critical stage in this procedure, guaranteeing that only authorized users had access to the remote workstations.

Furthermore, successfully establishing a Git repository on both local and remote workstations enabled effective version control and collaboration. The ability to execute ad hoc commands from a local system to remote servers demonstrated SSH's power and flexibility, allowing for remote administration and job execution. Overall, this exercise provided useful insights into how SSH may be used to administer remote systems securely and efficiently.