

<b>Name: Earl Kristian G. Villamor</b>	<b>Date Performed: August 23, 2022</b>
<b>Course/Section: CPE31S22</b>	<b>Date Submitted:</b>
<b>Instructor: Dr. Jonathan Taylar</b>	<b>Semester and SY: 1<sup>st</sup> sem 2022 - 2023</b>
<b>Activity 2: SSH Key-Based Authentication and Setting up Git</b>	
<b>1. Objectives:</b> <ul style="list-style-type: none"> <li>1.1 Configure remote and local machine to connect via SSH using a KEY instead of using a password</li> <li>1.2 Create a public key and private key</li> <li>1.3 Verify connectivity</li> <li>1.4 Setup Git Repository using local and remote repositories</li> <li>1.5 Configure and Run ad hoc commands from local machine to remote servers</li> </ul>	
<b>Part 1: Discussion</b> <p>It is assumed that you are already done with the last Activity (<b>Activity 1: Configure Network using Virtual Machines</b>). <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><b>What Is ssh-keygen?</b></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><b>SSH Keys and Public Key Authentication</b></p> <p>The <a href="#">SSH protocol</a> uses public key cryptography for authenticating hosts and users. The authentication keys, called <a href="#">SSH keys</a>, are created using the keygen program.</p> <p>SSH introduced <a href="#">public key authentication</a> 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

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`.

```
earl@workstation:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/earl/.ssh/id_rsa):
/home/earl/.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/earl/.ssh/id_rsa
Your public key has been saved in /home/earl/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:MWQu6hRI9WPA050wJnzMJ45V7vv1zbkx+1SP9bGZ2WQ earl@workstation
The key's randomart image is:
+---[RSA 3072]---+
| .+++++. |
| . +=B*+ |
| . *=0= |
| .+. + O |
| o S .E |
| o . =@ |
| . . ...*+ |
| . . .+00 |
| . .0=+ |
+---[SHA256]-----+
```

Figure 1.1

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.

```
earl@workstation:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.

Enter file in which to save the key (/home/earl/.ssh/id_rsa): /home/earl/.ssh/i
d_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/earl/.ssh/id_rsa
Your public key has been saved in /home/earl/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ACNWHdHmDfCfOKtQsQ8fA7odhDx/tUeiMmqVU1Ldu/U earl@workstation
The key's randomart image is:
+---[RSA 4096]---+
| o+=Boo . |
| .+..* . . |
| . . o . . |
| .. . . + o . |
| .o+ = ooS.o . |
| +o0 oo+. E |
| . +=0.. . |
| +.++++ . |
| o.o.o |
+---[SHA256]-----+
```

Figure 1.2

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.
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`.

```
earl@workstation:~$ ls -la .ssh
total 24
drwx----- 2 earl earl 4096 Aug 23 10:07 .
drwxr-x--- 15 earl earl 4096 Aug 23 09:00 ..
-rw----- 1 earl earl 3381 Aug 23 10:19 id_rsa
-rw-r--r-- 1 earl earl  742 Aug 23 10:19 id_rsa.pub
-rw----- 1 earl earl 2240 Aug 23 08:40 known_hosts
-rw----- 1 earl earl 1120 Aug 23 08:31 known_hosts.old
```

Figure 1.3

## 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`

```
earl@workstation:~$ ssh-copy-id earl@workstation
The authenticity of host 'workstation (127.0.0.1)' can't be established.
ED25519 key fingerprint is SHA256:5Sy9dtvVmor5X+5UkwrqPyvZi4HhcTkKfQFp8Ry3hGw.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? 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
earl@workstation's password:

Number of key(s) added: 1

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

Figure 2.1

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?

```
earl@workstation:~$ ssh-copy-id earl@server1
/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
earl@server1's password:

Number of key(s) added: 1

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

Figure 2.2

```
earl@workstation:~$ ssh-copy-id earl@server2
/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
earl@server2's password:

Number of key(s) added: 1

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

Figure 2.3

- It asked for a password of the server where I'm currently copying the SSH key to.

### Reflections:

Answer the following:

1. How will you describe the ssh-program? What does it do?
  - SSH program is often used by sys admins and developers to administer remote systems and applications, run commands, share files, and more because it offers strong encryption.
2. How do you know that you already installed the public key to the remote servers?
  - In order to determine if the public key is already instead among the servers from different devices, is to just insert the command: "cd ~/.ssh", followed by "cat id\_rsa.pub", and with that you'll be able to view the public key existing on a server.

## Part 2: Discussion

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

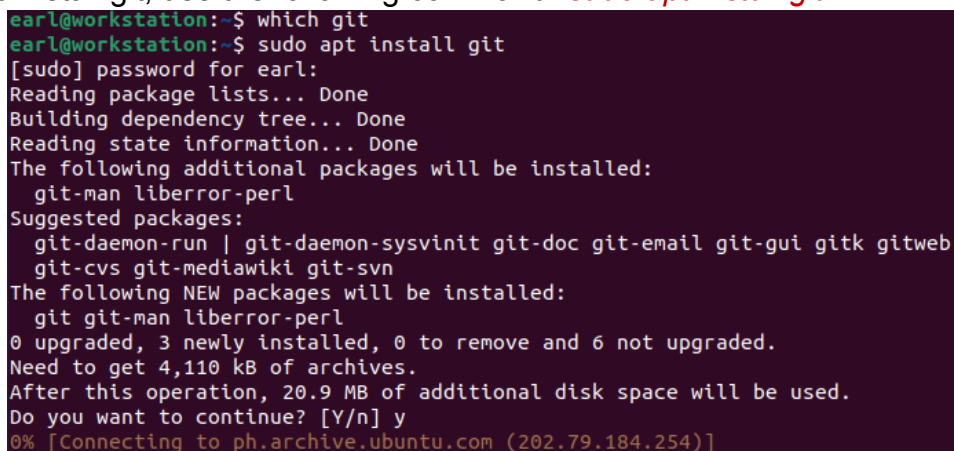
### Set up Git

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

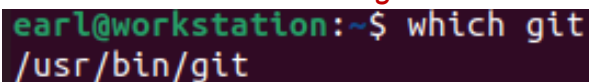
1. On the local machine, verify the version of your git using the command *which git*. If a directory of git is displayed, then you don't need to install git. Otherwise, to install git, use the following command: *sudo apt install git*



```
earl@workstation:~$ which git
earl@workstation:~$ sudo apt install git
[sudo] password for earl:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
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 6 not upgraded.
Need to get 4,110 kB of archives.
After this operation, 20.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
0% [Connecting to ph.archive.ubuntu.com (202.79.184.254)]
```

Figure 3.1

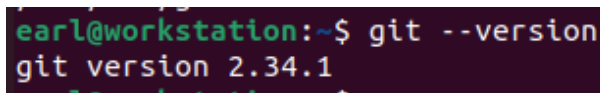
2. After the installation, issue the command *which git* again. The directory of git is usually installed in this location: *user/bin/git*.



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

Figure 3.2

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



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

Figure 3.3

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.
  - 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.

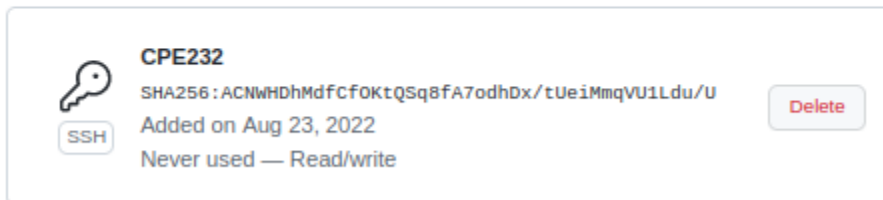
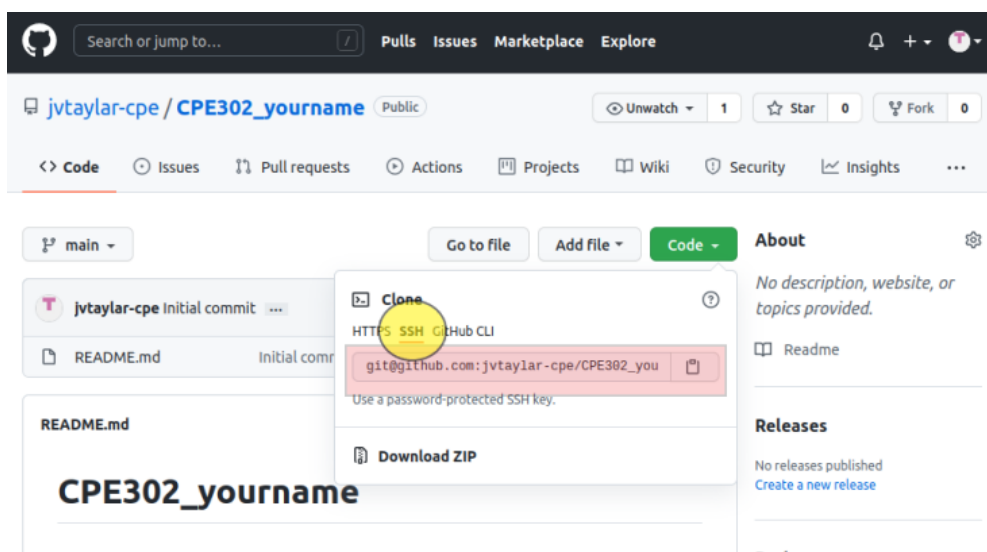


Figure 3.4

- 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.



- 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.

```
earl@workstation:~$ git clone git@github.com:Earl-Villamor/CPE232_EarlVillamor.git
git
Cloning into 'CPE232_EarlVillamor'...
The authenticity of host 'github.com (140.82.112.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhpZisF/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
Receiving objects: 100% (3/3), done.
```

Figure 3.5

- 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`.

```
earl@workstation:~$ ls
CPE232_EarlVillamor  Documents  id_rsa      Music      Public      Templates
Desktop              Downloads  id_rsa.pub  Pictures   snap        Videos
earl@workstation:~$ cd CPE232_EarlVillamor
earl@workstation:~/CPE232_EarlVillamor$ ls README.md
README.md
earl@workstation:~/CPE232_EarlVillamor$
```

Figure 3.6

- 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`

```
earl@workstation:~$ git config user.EarlV
earl@workstation:~$ git config --global user.name EarlV
earl@workstation:~$ git config --global user.email qekgvillamor@tip.edu.ph
earl@workstation:~$ cat ~/.gitconfig
[user]
    name = EarlV
    email = qekgvillamor@tip.edu.ph
earl@workstation:~$
```

Figure 3.7

- 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.
- 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?



```

earl@workstation:~/CPE232_EarlVillamor$ 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

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .README.md.swp

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

```

Figure 3.8

- 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.

```

earl@workstation:~/CPE232_EarlVillamor$ git add README.md
earl@workstation:~/CPE232_EarlVillamor$ git commit -m "This is it"
[main ce37ff3] This is it
1 file changed, 2 insertions(+), 1 deletion(-)

```

Figure 3.9

- 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`.

```

earl@workstation:~/CPE232_EarlVillamor$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 286 bytes | 286.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Earl-Villamor/CPE232_EarlVillamor.git
   326c68e..ce37ff3  main -> main

```

Figure 3.10

- 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 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.



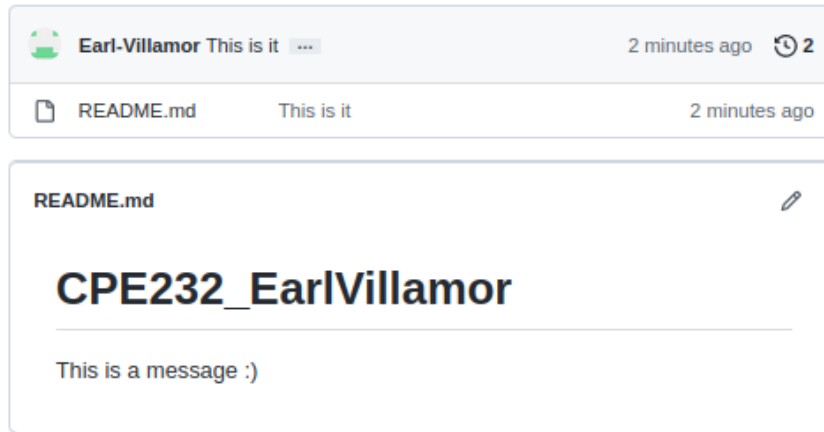


Figure 3.11

### Reflections:

Answer the following:

1. What sort of things have we so far done to the remote servers using ansible commands?

- Ansible is a configuration management application that makes it easier to set up and keep up distant servers. You may manage one to hundreds of systems from a single location thanks to its simple design, which is meant to get users up and running quickly.

-

2. How important is the inventory file?

- Managing an inventory file is just as essential as having playbooks or roles since, without one, you'll always find yourself confused and asking yourself a lot of questions. In addition to the preceding, the inventory file's definition of the required variables reduces the task's code in scripts and speeds up interpretation.

### Conclusions/Learnings:

- I have discovered various commands involving generating Keys, GitHub, especially when it comes to managing the file system itself. Learning the use of GitHub can be convenient in terms of being able to continue your work from any device you use as long as you access your GitHub account.