# NAME: DANIEL VILANOVA TITELLO

# SERVERS: GIT SERVER AND JENKINS

# GIT SERVER

## How to install

I installed git following the class tutorial. I used a VM with Ubuntu to install git server.

Create a VM for git server

To update ubuntu version

*Comand: sudo apt update*

To install git

*Comand: sudo apt install git*

A close up of a black background

Description automatically generated

I had git already installed from my previous installation.

Set up the git shell which is responsible for the following commands push, pulls and etc. Need to update the file with git shell.

A screenshot of a cell phone

Description automatically generated

My shell is located in /usr/bin/git-shell.

A black sign with white text

Description automatically generated

To open shells file:

*Comand: sudo vim /etc/shell*

Now we need to include the line “/usr/bin/git-shell”

A screenshot of a computer

Description automatically generated

To create a git user

*Comand: sudo useradd gitca*

*Comand: sudo passwd gitca: adminadmin*

A picture containing photo, black, phone, table

Description automatically generated

To generate a rsa key:

*Comand: ssh-keygen -t rsa*

A screenshot of a cell phone

Description automatically generated

To change my user

*Comand: sudo su gitca*



We create a new folder called gitproject  for this project. Within this folder, we initialize the git repository

*Comand: sudo git init --bare*

A picture containing photo, black, sign, red

Description automatically generated



## Testing

### Using git server

Using git bash, I cloned the directory to a folder in my local machine, opened the file that I had just cloned, made the changes, committed it and pushed it back to the git server. Initially, I was using my root user (dtitello) and later with a new user called git.

Install git in your local machine like Windows

A close up of a logo

Description automatically generated

Open git bash

Now we want to clone our host repository to our local repository

Comand: $ git clone dtitello@40.127.233.109:gitproject

Insert the vm password

A screen shot of a social media post

Description automatically generated

After cloning gitproject from the git master repository, I’m creating a new file and uploading to the master repository.

A screenshot of a cell phone

Description automatically generated

We did the same steps for the user git. We only need to make sure git has a home directory that will be used to create the repository.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

### Using github

I also decided to clone a few files from my github account, just to make sure that I had another scenario covered.

## Problems

### Cloning directory

I had a few issues cloning the directories because I wasn’t using the home directory. If I wanted to use a different directory, I should have changed the path of the command to clone.

### Pushing changes

I had problems pushing my changes to git server. At that point, I tried different users like git, dtitello and gitca but none of them worked. Fortunately, the user virtca worked well.

# JENKINS SERVER

## How to install

Luckily, installing jenkins was easier than git.

### Step 1 — Installing Jenkins

The version of Jenkins included with the default Ubuntu packages is often behind the latest available version from the project itself. To take advantage of the latest fixes and features, you can use the project-maintained packages to install Jenkins.

Command: sudo apt install openjdk-8-jdk

First, add the repository key to the system:

Command: wget -q -O - <https://pkg.jenkins.io/debian/jenkins-ci.org.key> | sudo apt-key add -

When the key is added, the system will return OK. Next, append the Debian package repository address to the server’s sources.list:

Command: echo "deb <http://pkg.jenkins.io/debian-stable> binary/" | sudo tee -a /etc/apt/sources.list.d/jenkins.list

When both of these are in place, run update so that apt will use the new repository:

Command: sudo apt update

Finally, install Jenkins and its dependencies:

Command: sudo apt install jenkins

Now that Jenkins and its dependencies are in place, we’ll start the Jenkins server.

### Step 2 — Starting Jenkins

Let’s start Jenkins using systemctl:

Command: sudo systemctl start jenkins



Since systemctl doesn’t display output, you can use its status command to verify that Jenkins started successfully:

Command:  sudo systemctl status jenkins

If everything went well, the beginning of the output should show that the service is active and configured to start at boot:

A screenshot of a cell phone

Description automatically generated

Now that Jenkins is running, let’s adjust our firewall rules so that we can reach it from a web browser to complete the initial setup.

### Step 3 — Opening the Firewall

By default, Jenkins runs on port 8080, so let’s open that port using ufw:

Command: sudo ufw allow 8080

A picture containing clock, meter, drawing

Description automatically generated

Check ufw’s status to confirm the new rules:

Command: sudo ufw status

A screenshot of a cell phone

Description automatically generated

**Note:** If the firewall is inactive, the following commands will allow OpenSSH and enable the firewall:

Command:  sudo ufw allow OpenSSH

Command: sudo ufw enable

A close up of a black background

Description automatically generated

### Step 4 — Setting Up Jenkins

To set up your installation, visit Jenkins on its default port, 8080, using your server domain name or IP address: <http://40.127.233.109:8080/>

A screenshot of a cell phone

Description automatically generated

In the terminal window, use the cat command to display the password:

* sudo cat /var/lib/jenkins/secrets/initialAdminPassword

A close up of a sign

Description automatically generated

Password: b5bba15c593c4d269cc9070551138831

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

### Step 5 - Authenticating on your GIT server from Jenkins

Scenario: you have 2 x VMs, one set up as a GIT server and one as a Jenkins server.

Assumption is that you have a user on your GIT server called git who owns your project repo (substitute your own username otherwise) and that you are logged onto your Jenkins server machine as the admin.

1. Give the Jenkins user a password you can remember

**sudo passwd jenkins**

2. Promote the jenkins user to be a sudoer

**sudo usermod -aG sudo jenkins**

3. Start running as that Jenkins user:

**sudo su - jenkins**

4. move to their home directory

**cd $home**

5. Generate authentication keys

**sudo -u jenkins ssh-keygen**

6. Send those keys to your GIT server (so that this jenkins user is trusted)

**ssh-copy-id virtca@52.170.198.175**

Now go back to your Jenkins UI and you should be able to set up your repo in your job config

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

Description automatically generated

### Step 6 – Connecting Jenkins to Apache Server

To allow Jenkins to upload the output file from git within apache server, I had to install one plugin allowing ssh access and configure access to the apache server and its path.

## Testing

### Using git server

I also used the same methodology to test Jenkins. Cloned a repository from git server, made the changes in a file called index.html and pushed it back to git. Ran the Jenkin’s job to get the file from git and uploads to the specific folder within Apache. The result, could have been seen accessing “host/index.html”

### Using github

I followed the same process above with one little change, I cloned from my github account.

## Problems

The only problem with Jenkins was configuring the right path for both servers: git and apache. Also, I forgot the username to access Jenkins so I had to set a security flag to false so I could access Jenkins and find out my username. Apparently, it’s something that happens more than it should.

# CONCLUSION

I believe that git and github are very useful tools to use for work however I am not sure about Jenkins. I definitely understand the usability however I think deploying changes to production requires more steps and security than the ones used in this class.