May 14, 2021

How to Build Your First DevOps Lab – Part 2: Configuring the Ubuntu Master

Where do we start?

That was the main question in the first part, right? Well, we have to start somewhere, and that somewhere is with the Ubuntu master VM. Starting with this VM makes the most sense as it is hosting the majority of the products that will be used in this lab build. It is the central hub for information in this DevOps lab.

Normally, I would not recommend doing this as it creates a single point of failure. However, because I wanted to be mindful of those that may not have the resources to split everything out into separate components, this configuration is what I decided on. This part will go over the tasks to get the Ubuntu master VM setup with Jenkins and Ansible, with Splunk coming in a later part.

Responsibilities of the Ubuntu Master

The Ubuntu Master VM will handle communications to our Ubuntu node, allowing us to configure and deploy to it remotely and autonomously. We can achieve autonomy via Jenkins. Jenkins helps automate the building, testing, and deploying parts of the software development lifecycle, allowing for a Continuous Integration and Continuous Delivery (CICD) pipeline. Jenkins has great integrations with other external systems, including GitHub, Jira, and more; both of which will be shown in this series. The installation of Jenkins is quite straight forward as you'll soon see.

The other installation that will be covered here is Ansible, which is also just as simple. Ansible will help us run remote commands on the Ubuntu Node. If you are familiar with Jenkins, it is possible to add the Ubuntu Node as a Node in Jenkins, which would alleviate the need for Ansible altogether. However, this is not the best approach given how popular Ansible is. Knowing how to use it can be quite valuable, which is why we are using this approach instead.

Below you will find the necessary steps to get both Jenkins and Ansible installed on the Ubuntu Master VM:

1. Login to the Ubuntu Master VM and open a terminal window
2. Install Jenkins on Ubuntu master VM

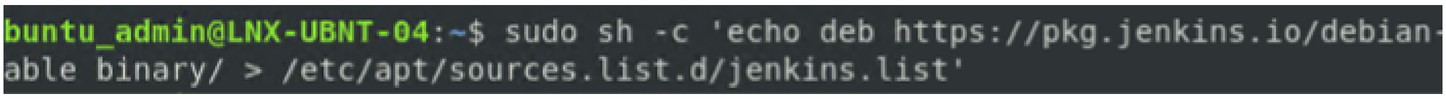
a. Get Jenkins Authentication Key and add it to the list of keys used by apt to authenticate packages

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key
| sudo apt-key add -
```

```
buntu_admin@LNX-UBNT-04:~$ wget -q -O - https://pkg.jenkins.io/debian-stable/j
kins.io.key | sudo apt-key add -
K
```

b. Add Jenkins source to the apt list

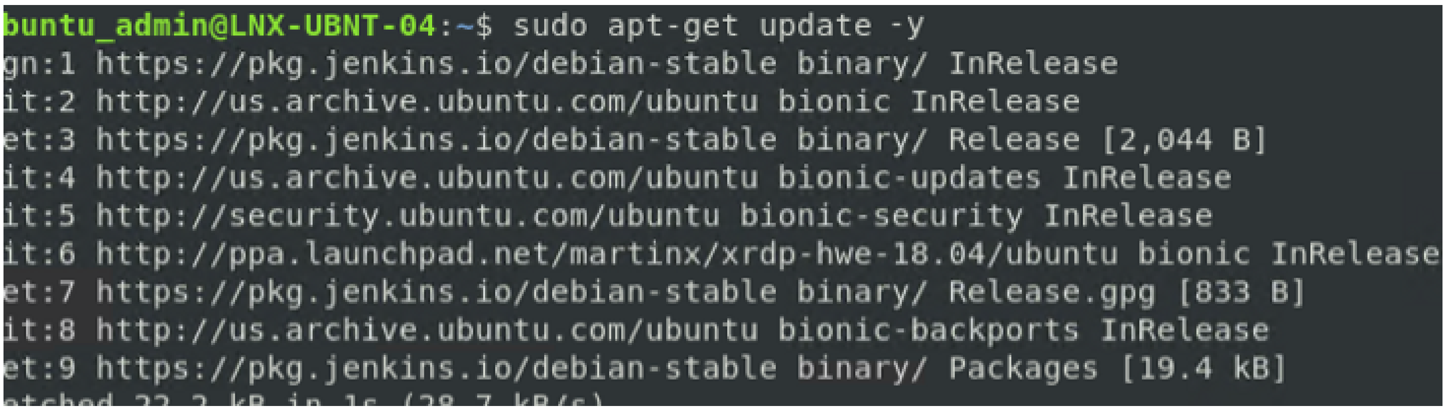
```
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/
> /etc/apt/sources.list.d/jenkins.list'
```



```
buntu_admin@LNX-UBNT-04:~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-
able binary/ > /etc/apt/sources.list.d/jenkins.list'
```

c. Update apt to make sure it uses the new source when installing Jenkins

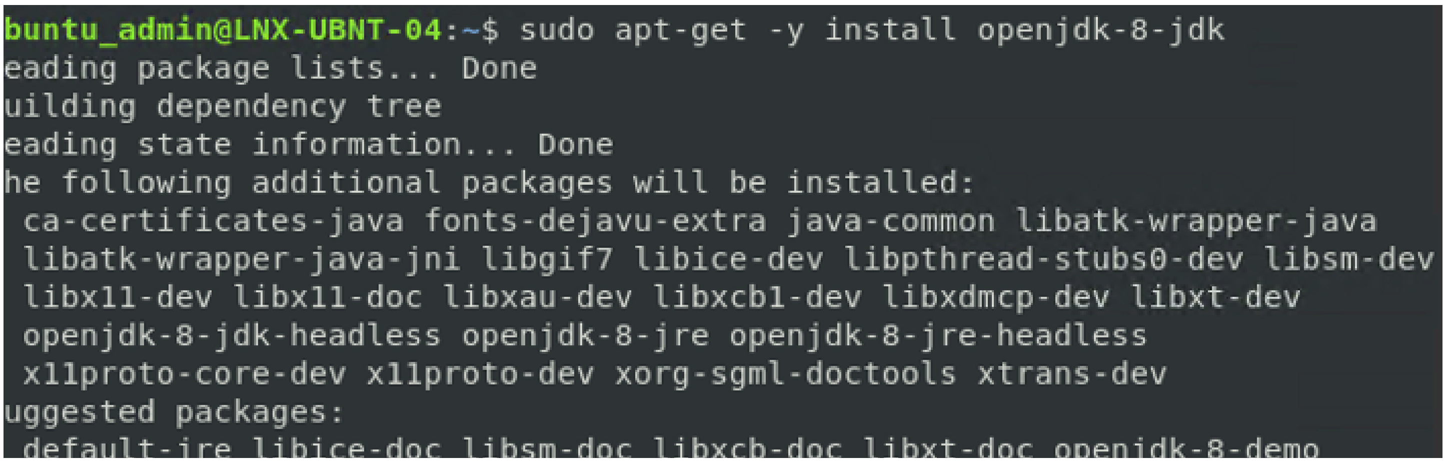
```
sudo apt-get update
```



```
buntu_admin@LNX-UBNT-04:~$ sudo apt-get update -y
gn:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
it:2 http://us.archive.ubuntu.com/ubuntu bionic InRelease
et:3 https://pkg.jenkins.io/debian-stable binary/ Release [2,044 B]
it:4 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
it:5 http://security.ubuntu.com/ubuntu bionic-security InRelease
it:6 http://ppa.launchpad.net/martinx/xrdp-hwe-18.04/ubuntu bionic InRelease
et:7 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
it:8 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
et:9 https://pkg.jenkins.io/debian-stable binary/ Packages [19.4 kB]
```

d. Install Java as a prerequisite for installing Jenkins

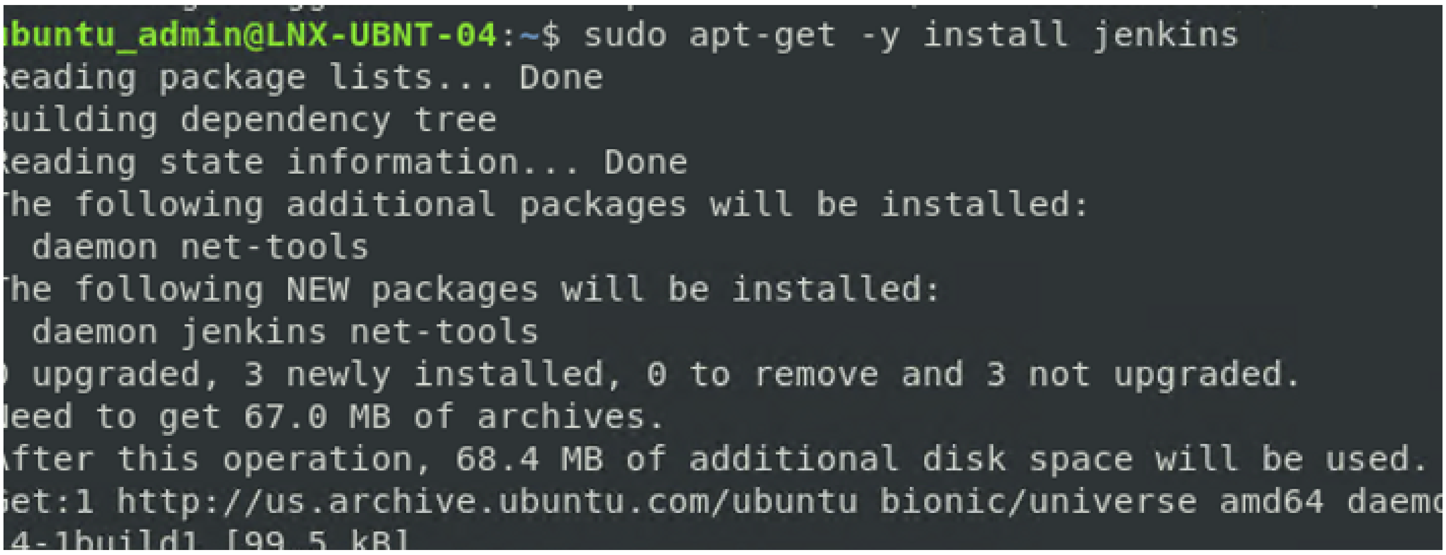
```
sudo apt-get -y install openjdk-8-jdk
```



```
buntu_admin@LNX-UBNT-04:~$ sudo apt-get -y install openjdk-8-jdk
eading package lists... Done
uilding dependency tree
eading state information... Done
he following additional packages will be installed:
 ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
 libatk-wrapper-java-jni libgif7 libice-dev libpthread-stubs0-dev libsm-dev
 libx11-dev libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
 openjdk-8-jdk-headless openjdk-8-jre openjdk-8-jre-headless
 x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
uggested packages:
 default-jre libice-doc libsm-doc libxcb-doc libxt-doc openjdk-8-demo
```

e. Install Jenkins

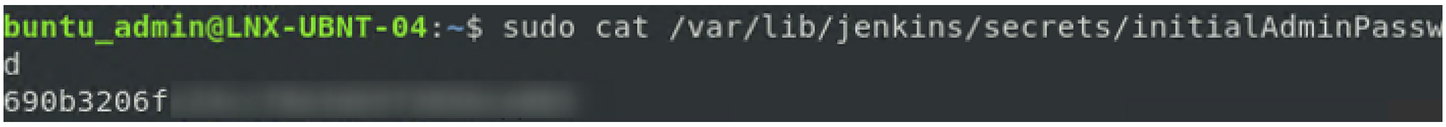
```
sudo apt-get -y install jenkins
```



```
ıbuntu_admin@LNX-UBNT-04:~$ sudo apt-get -y install jenkins
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
 daemon net-tools
The following NEW packages will be installed:
 daemon jenkins net-tools
) upgraded, 3 newly installed, 0 to remove and 3 not upgraded.
Need to get 67.0 MB of archives.
After this operation, 68.4 MB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu bionic/universe amd64 daemo
4-1build1 [99.5 kB]
```

f. View the contents of the InitialAdminPassword file for further steps

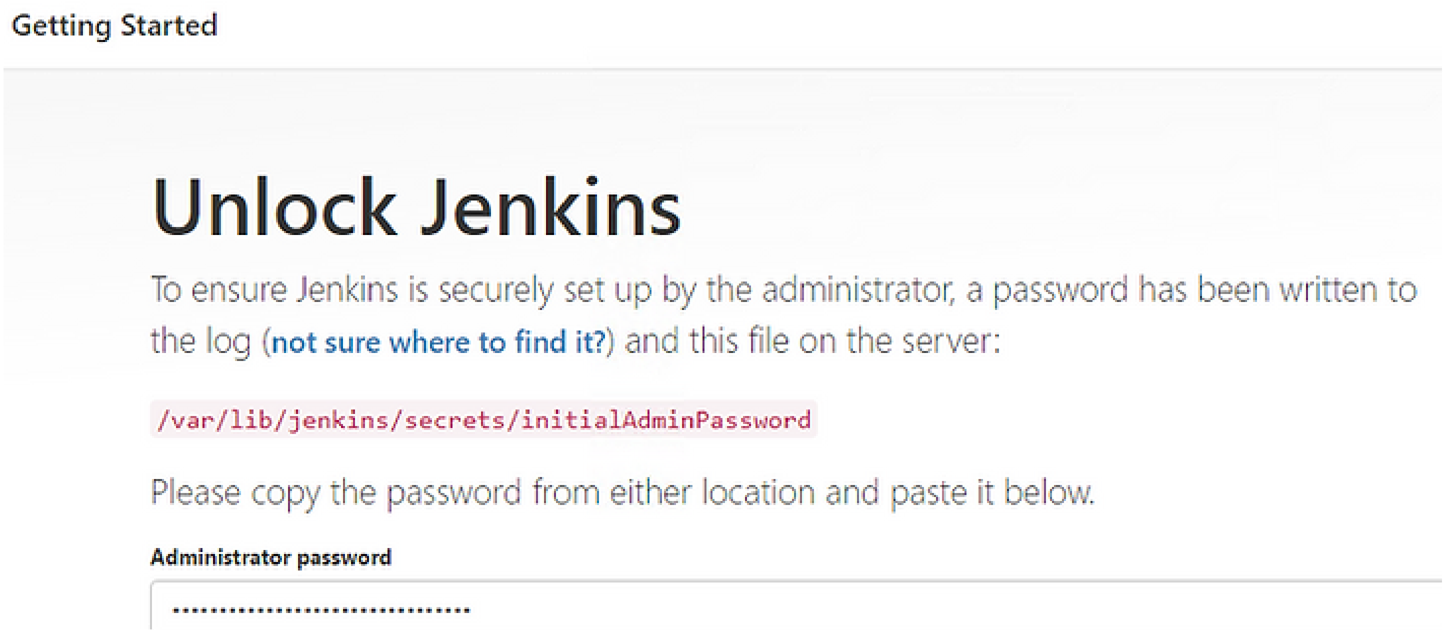
```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



```
buntu_admin@LNX-UBNT-04:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassw
d
690b3206f[redacted]
```

3. Connect to Jenkins web interface via http://[IP]:8080, replacing IP with the IP of your Ubuntu master VM

4. Input the password from the file in step 2f



Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

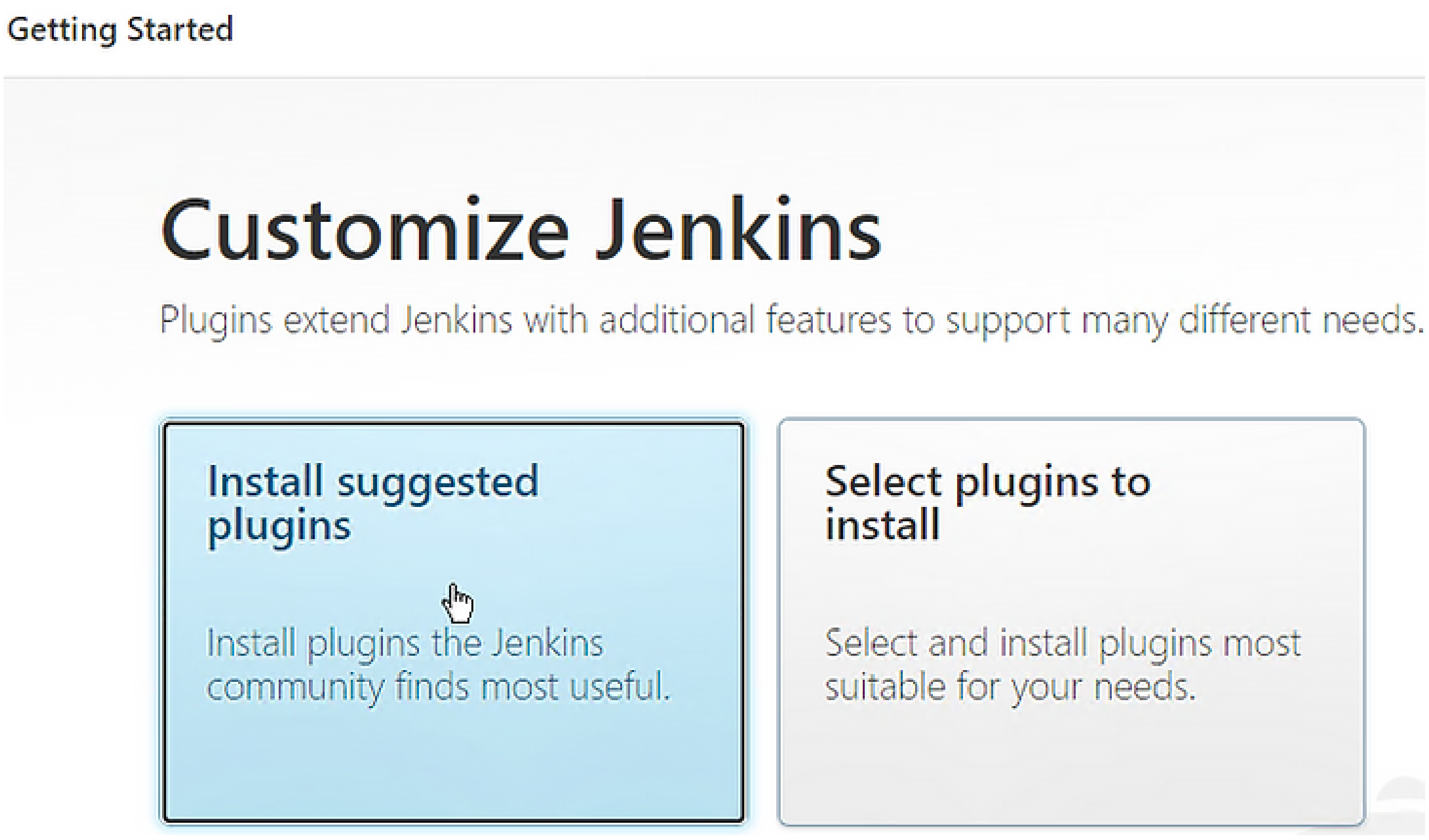
/var/lib/jenkins/secrets/initialAdminPassword

Please copy the password from either location and paste it below.

Administrator password

································

5. Install all suggested Plugins

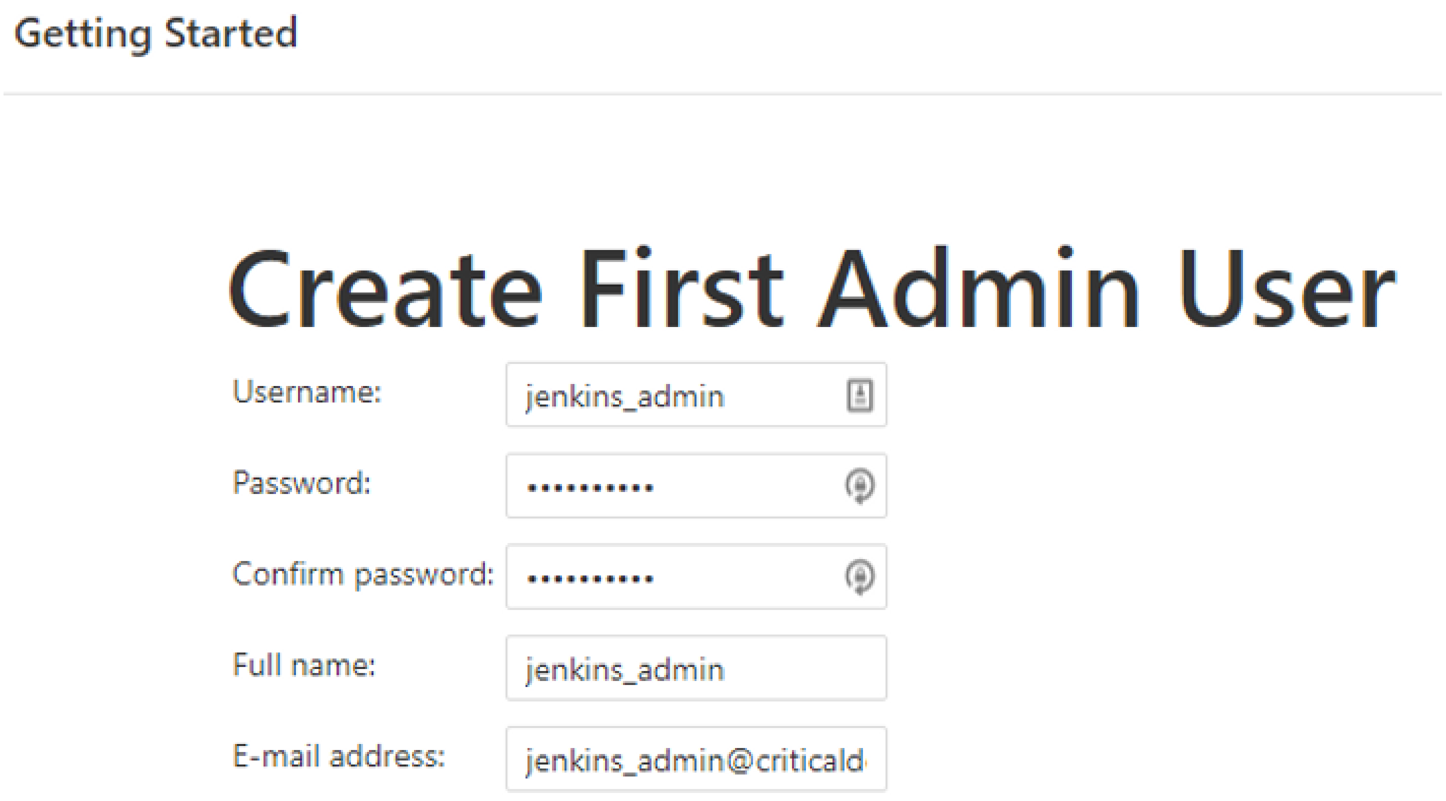

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.

6. Create Jenkins admin account and continue to Jenkins web interface


Getting Started

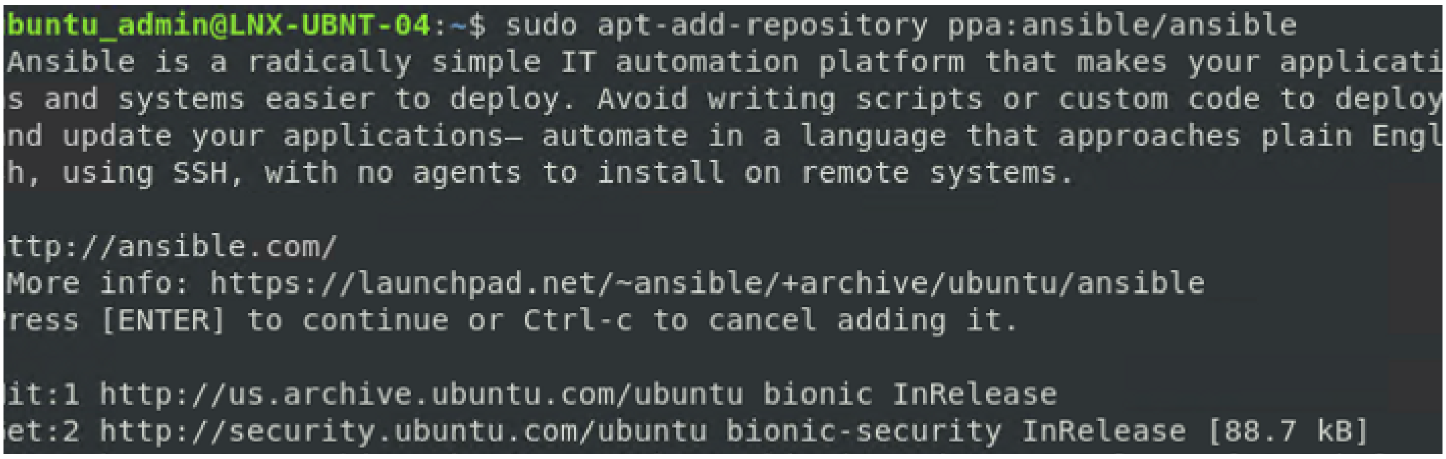
Create First Admin User

Username:	jenkins_admin
Password:	••••••••••
Confirm password:	••••••••••
Full name:	jenkins_admin
E-mail address:	jenkins_admin@criticald

7. Run through commands to install Ansible on Master device

a. Add Ansible apt repository

```
sudo apt-add-repository ppa:ansible/ansible
```



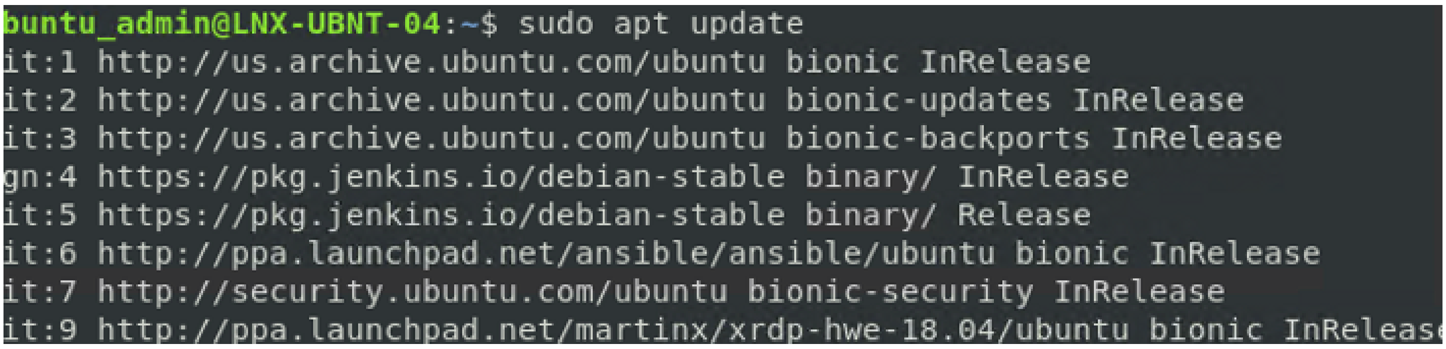
```
buntu_admin@LNX-UBNT-04:~$ sudo apt-add-repository ppa:ansible/ansible
Ansible is a radically simple IT automation platform that makes your applicati
s and systems easier to deploy. Avoid writing scripts or custom code to deploy
nd update your applications— automate in a language that approaches plain Engl
h, using SSH, with no agents to install on remote systems.

ttp://ansible.com/
More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
ress [ENTER] to continue or Ctrl-c to cancel adding it.

it:1 http://us.archive.ubuntu.com/ubuntu bionic InRelease
et:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
```

b. Update apt so that it can make use of that new repository

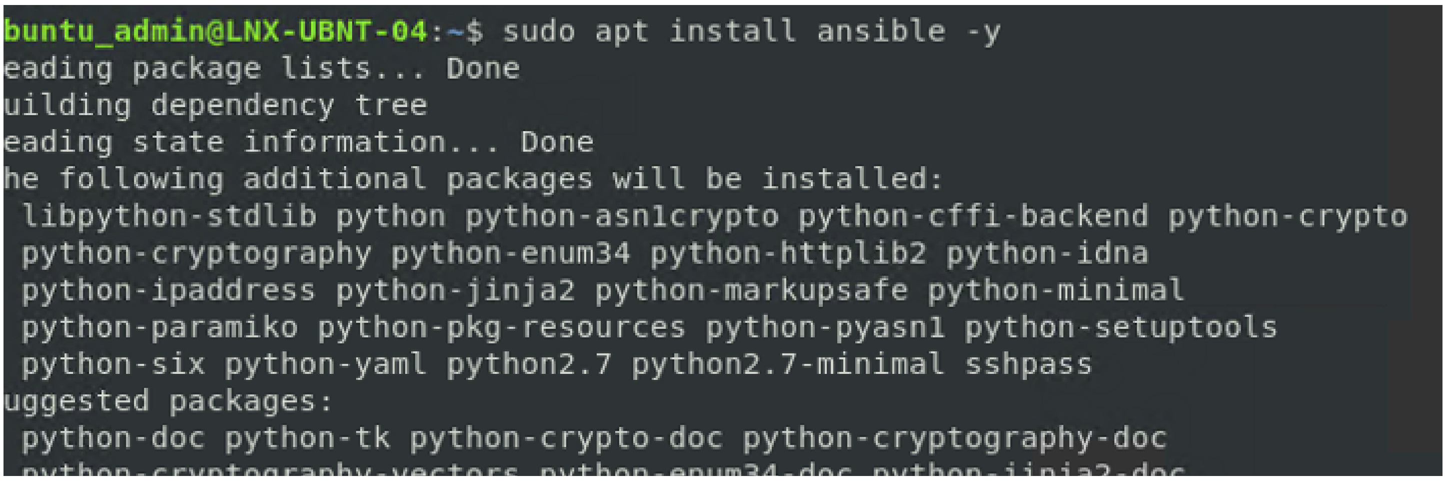
```
sudo apt update
```



```
buntu_admin@LNX-UBNT-04:~$ sudo apt update
it:1 http://us.archive.ubuntu.com/ubuntu bionic InRelease
it:2 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
it:3 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
gn:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
it:5 https://pkg.jenkins.io/debian-stable binary/ Release
it:6 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic InRelease
it:7 http://security.ubuntu.com/ubuntu bionic-security InRelease
it:9 http://ppa.launchpad.net/martinx/xrdp-hwe-18.04/ubuntu bionic InReleas
```

c. Install Ansible

```
sudo apt install ansible -y
```



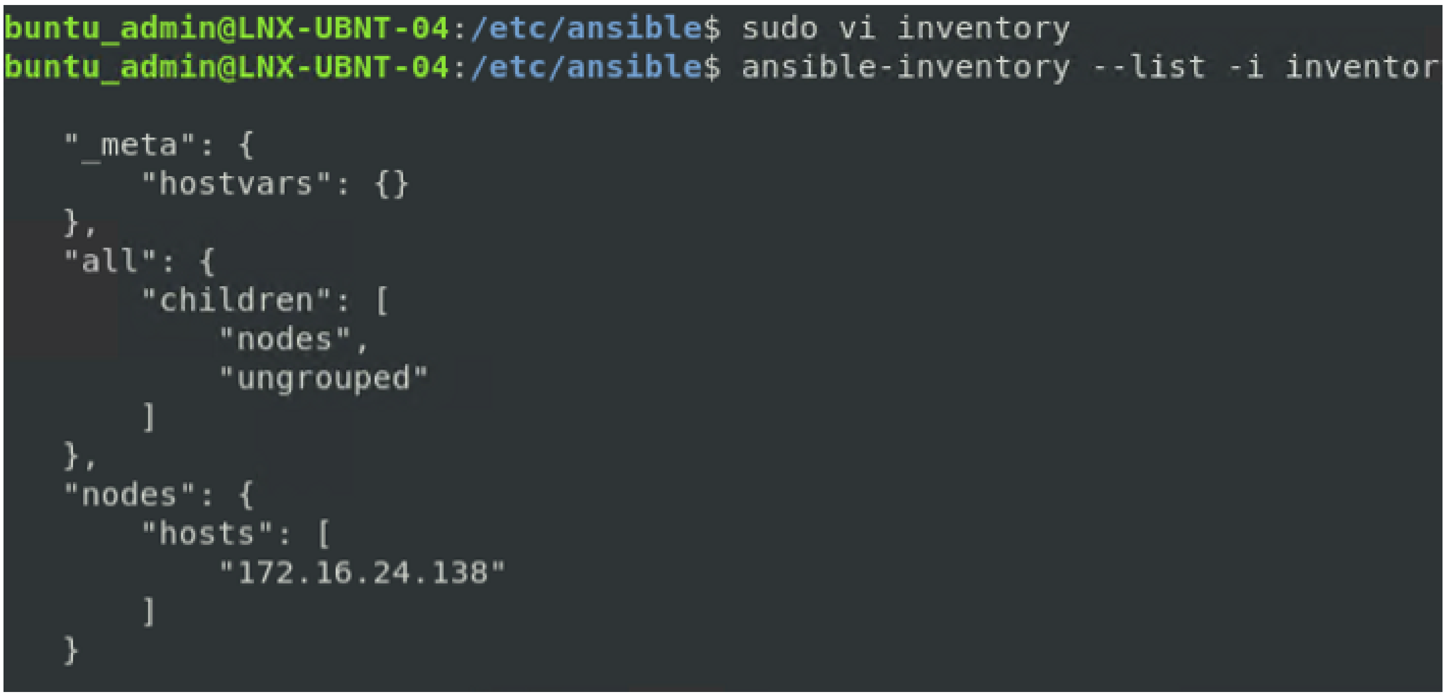
```
buntu_admin@LNX-UBNT-04:~$ sudo apt install ansible -y
eading package lists... Done
uilding dependency tree
eading state information... Done
he following additional packages will be installed:
 libpython-stdlib python python-asn1crypto python-cffi-backend python-crypto
 python-cryptography python-enum34 python-httplib2 python-idna
 python-ipaddress python-jinja2 python-markupsafe python-minimal
 python-paramiko python-pkg-resources python-pyasn1 python-setuptools
 python-six python-yaml python2.7 python2.7-minimal sshpass
uggested packages:
 python-doc python-tk python-crypto-doc python-cryptography-doc
```

8. Configure a new Ansible inventory file and make sure it is being read correctly

- **a.** Be sure to replace the IP shown here under File Contents with the IP address for your Node
- **b.** The argument in the inventory file is used to disable strict host checking for SSH; this can be removed after running the Jenkins pipeline for the first time in a future part if desired. This is done to prevent the Jenkins build from failing. If the argument wasn't included, the Jenkins build would fail unless you previously SSH'ed into the node with the account that will be running the ansible command in Jenkins.

```
cd /etc/ansible
sudo vi inventory
ansible-inventory –list -i inventory

File Contents:
[nodes]
172.16.24.138 ansible_ssh_common_args='-o
StrictHostKeyChecking=no'
```



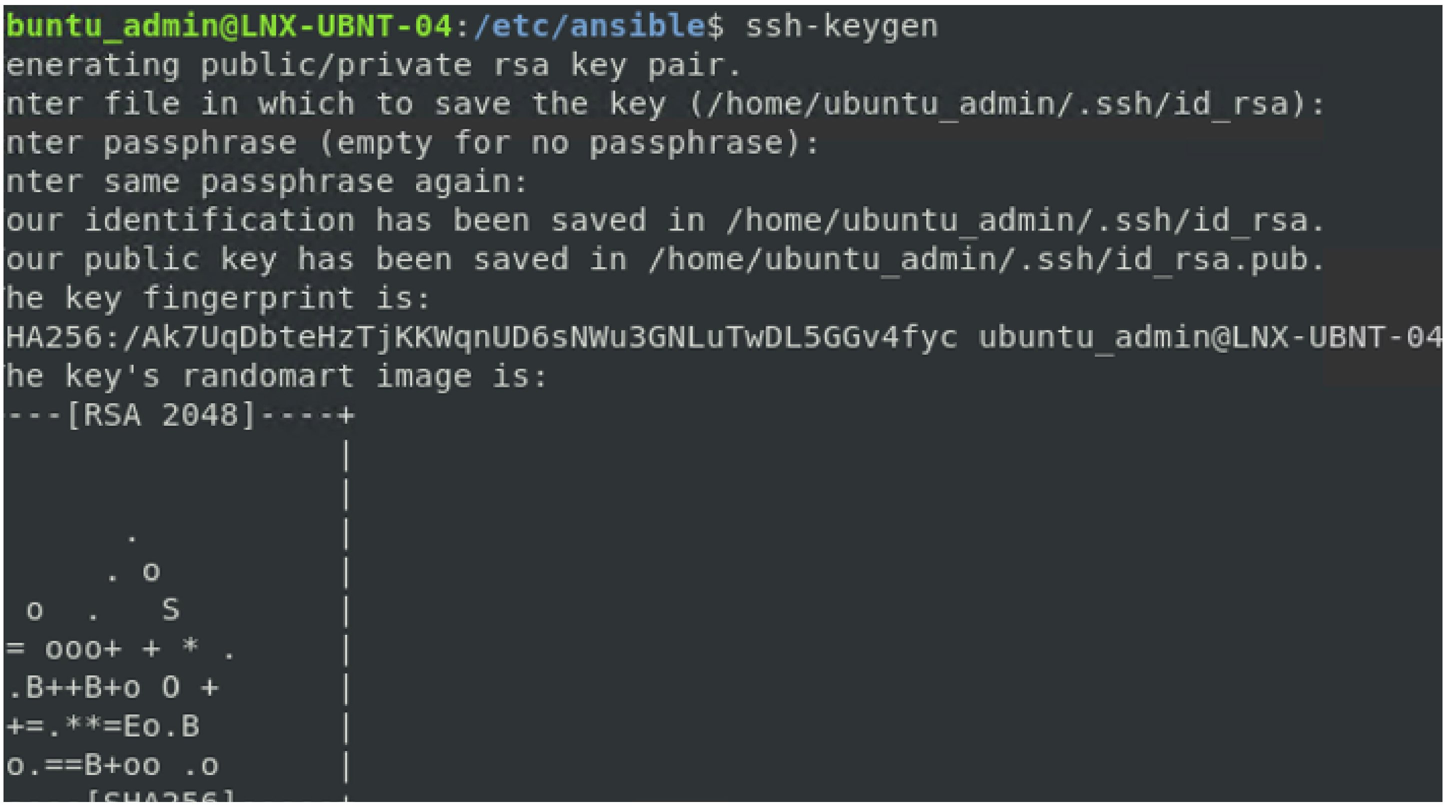
```
buntu_admin@LNX-UBNT-04:/etc/ansible$ sudo vi inventory
buntu_admin@LNX-UBNT-04:/etc/ansible$ ansible-inventory --list -i inventor

   "_meta": {
       "hostvars": {}
   },
   "all": {
       "children": [
           "nodes",
           "ungrouped"
       ]
   },
   "nodes": {
       "hosts": [
           "172.16.24.138"
       ]
   }
```

9. Create SSH keys

  a. NOTE: Do not put a password in when prompted to do so. If you do, you will not be able to automate the Jenkins deployment as it will ask for you to input a password when ansible does an SSH connection to the Node

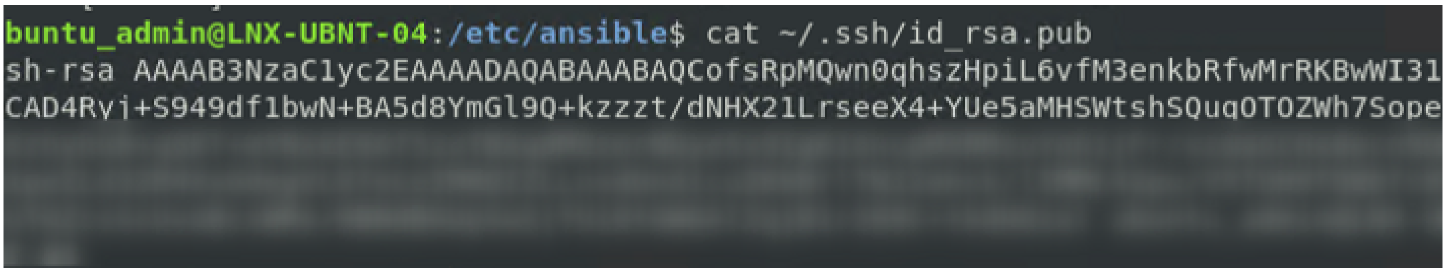
```
ssh-keygen
```



```
buntu_admin@LNX-UBNT-04:/etc/ansible$ ssh-keygen
enerating public/private rsa key pair.
nter file in which to save the key (/home/ubuntu_admin/.ssh/id_rsa):
nter passphrase (empty for no passphrase):
nter same passphrase again:
our identification has been saved in /home/ubuntu_admin/.ssh/id_rsa.
our public key has been saved in /home/ubuntu_admin/.ssh/id_rsa.pub.
he key fingerprint is:
HA256:/Ak7UqDbteHzTjKKWqnUD6sNWu3GNLuTwDL5GGv4fyc ubuntu_admin@LNX-UBNT-04
he key's randomart image is:
---[RSA 2048]----+
                 |
                 |
      .          |
     . o         |
 o  .   S        |
= ooo+ + * .     |
.B++B+o O +      |
+=.**=Eo.B       |
o.==B+oo .o      |
----[SHA256]-----+
```

10. View contents of the public key and copy it to an external text editor

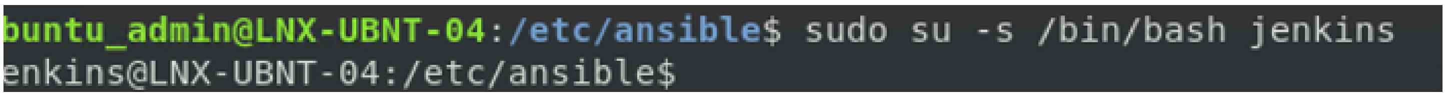
```
cat ~/.ssh/id_rsa.pub
```



```
buntu_admin@LNX-UBNT-04:/etc/ansible$ cat ~/.ssh/id_rsa.pub
sh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCofsRpMQwn0qhszHpiL6vfM3enkbRfwMrRKBwWI31
CAD4Rvj+S949df1bwN+BA5d8YmGl9Q+kzzzt/dNHX21LrseeX4+YUe5aMHSWtshSQuqOTOZWh7Sope
[redacted]
```

11. Login as the Jenkins user

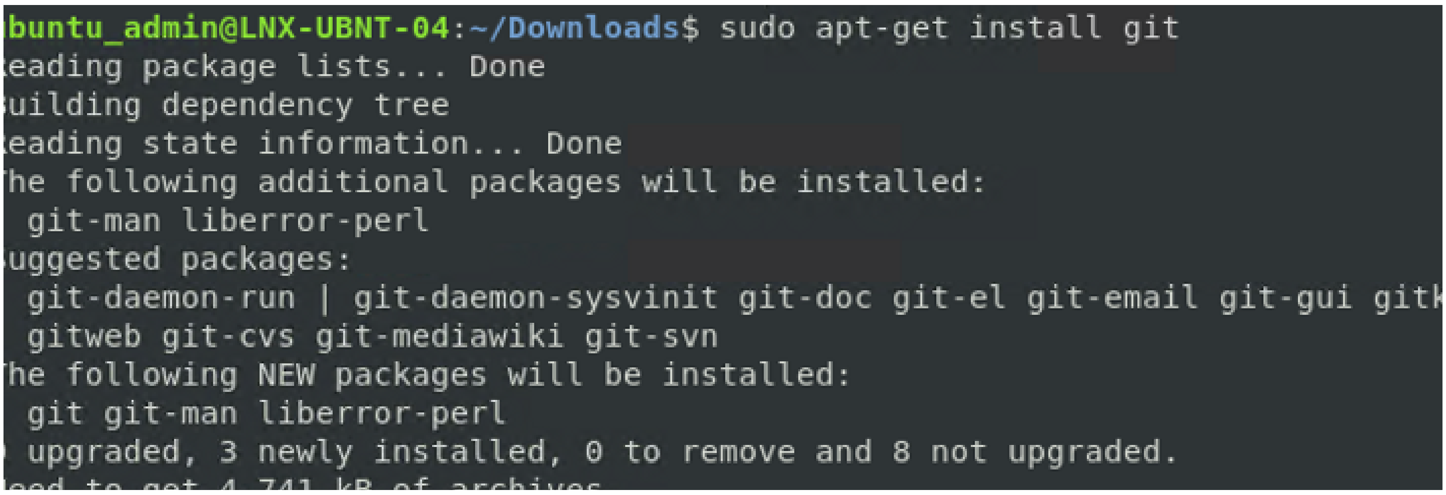
```
sudo su -s /bin/bash jenkins
```



```
buntu_admin@LNX-UBNT-04:/etc/ansible$ sudo su -s /bin/bash jenkins
enkins@LNX-UBNT-04:/etc/ansible$
```

12. Repeat steps 9 and 10 for the Jenkins user
13. Install Git for a later part, if not already installed

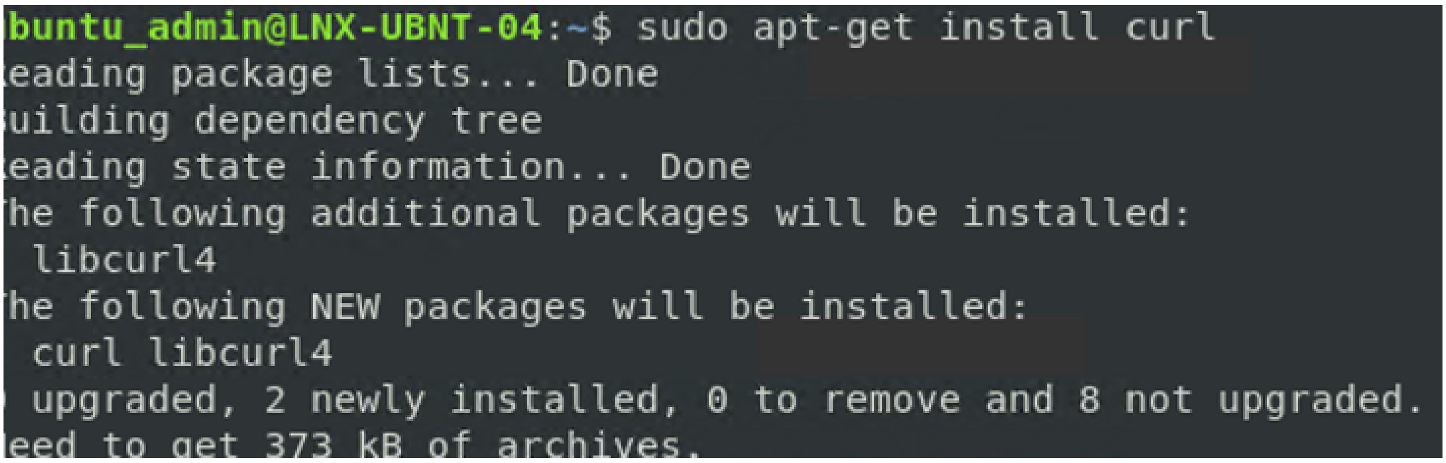
```
sudo apt-get install git -y
```



```
buntu_admin@LNX-UBNT-04:~/Downloads$ sudo apt-get install git
eading package lists... Done
uilding dependency tree
eading state information... Done
he following additional packages will be installed:
 git-man liberror-perl
uggested packages:
 git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
 gitweb git-cvs git-mediawiki git-svn
he following NEW packages will be installed:
 git git-man liberror-perl
 upgraded, 3 newly installed, 0 to remove and 8 not upgraded.
```

14. Install Curl for a later part, if not already installed

```
sudo apt-get install curl -y
```



```
buntu_admin@LNX-UBNT-04:~$ sudo apt-get install curl
eading package lists... Done
uilding dependency tree
eading state information... Done
he following additional packages will be installed:
 libcurl4
he following NEW packages will be installed:
 curl libcurl4
 upgraded, 2 newly installed, 0 to remove and 8 not upgraded.
eed to get 373 kB of archives.
```

What now?

Well, this series is far from over. In the next part, I'll discuss further the purpose of the Ubuntu node VM and walk through the configuration. Your Master node should be at a good spot moving forward to finish the rest of the configurations in a future part. The Ubuntu node is responsible for running our Docker containers, allowing us to achieve consistent runtime environments.

If you want to continue this configuration, check out part 3!

DevOps Lab Build Series Index