Automated Jenkins Install for Centos

By Jimmy Orcutt for `Puppet`

Table of Contents

Technical Challenge Questions

1. *Describe the most difficult hurdle you had to overcome in implementing your solution.*

   - The biggest hurdle I encountered was figuring out that Vagrant reserves eth0 for NAT and this wasn't the IP Jenkins was being served from. It took me a bit to figure out that I needed to add an interface.

2. *Please explain why requirement (d) above is important.*

   ```
   (d) Subsequent applications of the solution should not cause failures or
   repeat redundant configuration tasks
   ```

   - If an automated solution causes failures and repeats redundant configuration tasks then it defeats the purpose of automating a manual process. Automation should remove human-error, save time, and streamline a process.

3. *Where did you go to find information to help you?*

   - Google, StackExchange, ServerFault, and random posts on forums about random Linux commands and switches I could not remember off the top of my head.

4. Briefly explain what automation means to you, and why it is important to an organization's infrastructure design strategy.

   - Automation frees up staff to spend time on other value-adding activities that only humans are capable of rather than updating/configuring servers, installing programs and many other tasks. Not only does automation itself save time (therefore money), but also allows staff to spend time on solutions, new product ideas, features, and anything else that humans are good at. Automation also allows for great disaster recovery options. If an infrastructure goes down in part or entirety and can be fired up without manual intervention downtime can be severely minimized.

Compatibility

`centos-jenkins-provision.sh` is compatible with the following Operating Systems out of the box -

1. Centos 7
2. Centos 6

Description

`centos-jenkins-provision.sh` will perform the following out of the box -

1. Check if Jenkins is installed.
2. Install `Java 1.8`
3. Install Jenkins from URL
4. Set Jenkins to serve on port `8000`
5. Start Jenkins and enable it on boot
6. Provide you with a link to the server in the console.

How-To

Use this script with your favorite configuration management platform, as a `Vagrant` provisioning script, or run it on an already standing server. See Arguments for additional options.

- Vagrant
  1. Drop the script into your Vagrant folder and make sure `config.vm.provision :shell, path: "centos/centos-jenkins-provision.sh"` is set. Otherwise you can copy the contents into your existing provisioning script.
- Existing system
  1. Copy the script to your system.
  2. Give execution permission with `chmod a+x centos-jenkins-provision.sh`
  3. Execute with `./centos-jenkins-provision.sh`

Notes

- `Vagrant` uses the first network device (`eth0` in this case) for NAT and Jenkins will not be accessible on this address. There is no way around this without specifying a default router in your `Vagrantfile` which may be too complicated for a simple deployment. https://github.com/hashicorp/vagrant/issues/2093#issuecomment-23458455-permalink
- If using `Vagrant` be sure to specify `config.vm.network "public_network"` or `config.vm.network "private_network", type: "dhcp"` to be sure an additional accessible network interface is added to your server for Jenkins.
- The script will provide you with `eth1` if it exists, otherwise you will be given the `eth0` IP.

```
if [ -z "$(hostname -I | awk '{print $2}')" ]
then
    IPADDR="$(hostname -I)"
```

```
else
    IPADDR="$(hostname -I | awk '{print $2}')"
fi

echo "Access Jenkins via browser from http://$IPADDR:8000 in about 10 seconds."
```

Arguments

This script has 3 options that can be passed as arguments into the script. The order does not matter and 1, 2, or all 3 can be used.

```
./centos-jenkins-provision.sh -fw
./centos-jenkins-provision.sh -fw -update
./centos-jenkins-provision.sh -fw -update -skipwiz
./centos-jenkins-provision.sh -fw -skipwiz -fw
```

To use arguments on your provisioning script in Vagrant see the following example.

```
 config.vm.provision :shell, path: "centos-jenkins-provision.sh", :args =>"-update
-fw -skipwiz"
```

-fw

`WARNING` Will enable firewall if it is not on already.

`./centos-jenkins-provision.sh -fw` will enable the firewall in Centos 6/7 and open port 8000.

In Centos 7

```
sudo systemctl start firewalld
sudo firewall-cmd --permanent --zone=public --add-port=8000/tcp
sudo firewall-cmd --reload
```

In Centos 6

```
sudo iptables -A INPUT -m state --state NEW -p tcp --dport 8000 -j ACCEPT
sudo iptables-save | sudo tee /etc/sysconfig/iptables
```

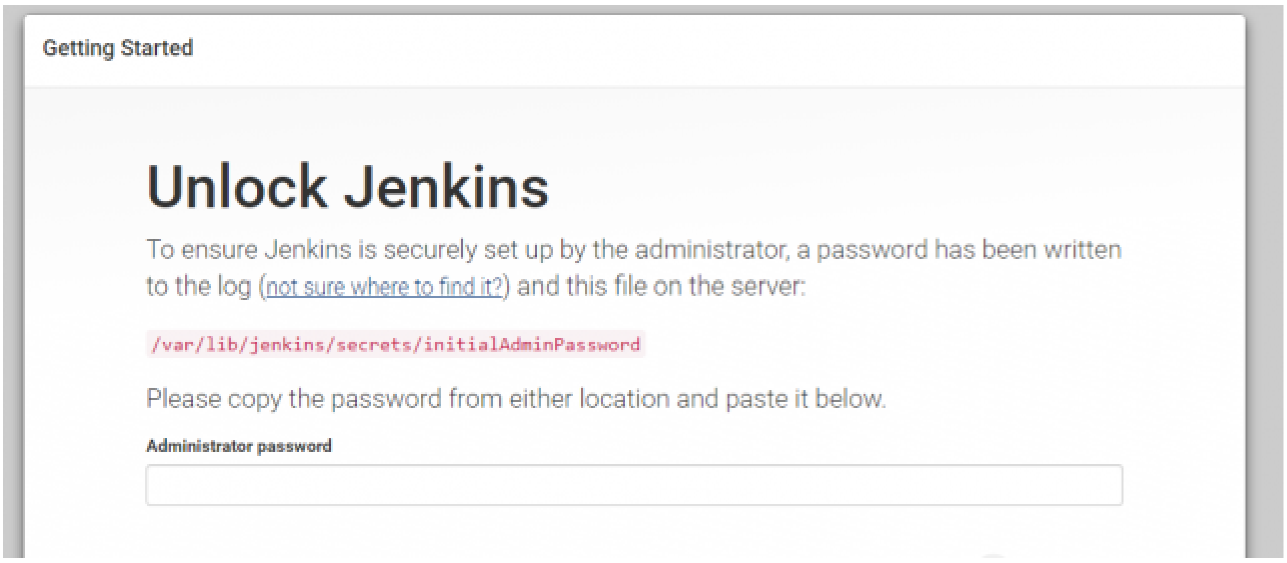
-update

`./centos-jenkins-provision.sh -update` will first update the OS before Jenkins and prerequisites are installed. You may only want to do this on a fresh machine.

```
sudo yum update -y
```

-skipwiz

`WARNING` Insecure if you do not set a secure admin username and password immediately after installation.


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`

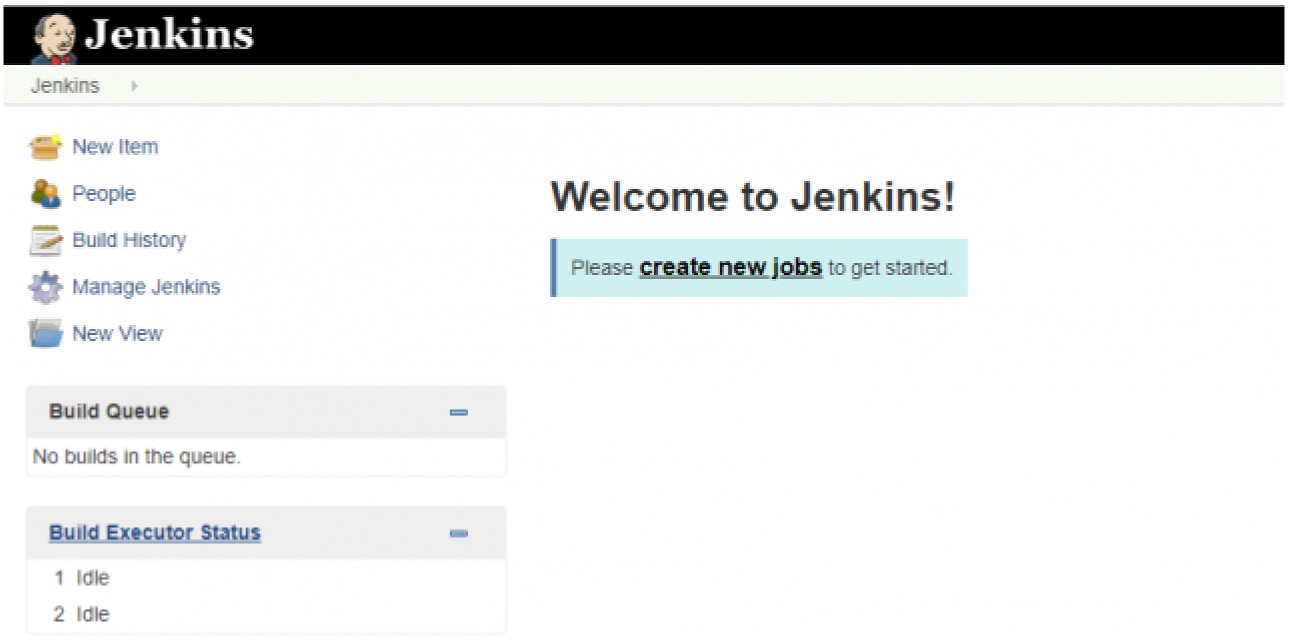
Please copy the password from either location and paste it below.

Administrator password

Normally Jenkins will require you to paste a generated password to "unlock" the server. This can be circumvented to allow for a fully automated deployment getting you straight to an unlocked Jenkins dashboard after install.

```
sudo sed -i 's/JENKINS_JAVA_OPTIONS="-
Djava.awt.headless=true"/JENKINS_JAVA_OPTIONS="-Djava.awt.headless=true -
Djenkins.install.runSetupWizard=false"/g' /etc/sysconfig/jenkins
```

After install completes you will see the unlocked dashboard



Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Welcome to Jenkins!

Please **create new jobs** to get started.