

Cloud Computing and Deployment

EFRAIN FERNANDEZ SANGRADOR

2. N02-Linux-Cluster

Instructions

Clone the virtual environment from the exercise of this environment N01 to a total of 10 instances

Then edit the network settings for each cloned instance so that it has its own local Internet Protocol (IP) address number and the communication to the OpenSSH server is enabled between these instances and to them.

Then show that you can connect to each hosted environment (computer 1) from the host computer (computer 0), and from this hosted environment to another hosted environment (computer 2), and then run the `hostname -I` command at the very latest (computer 2).

Show this for all combinations, and since there are 90 combinations of links, make this printout using a Bash script and command `parallel-ssh` (example for a few combinations: `ssh comp1 parallel-ssh -i -H comp2 -H comp3 -H comp4 -H comp5 -H comp6 -H comp7 -H comp8 -H comp9 -H comp10 hostname -I`).

Material for hints during the explanation of the task:

```
guest# apt-get update; apt-get install pssh
```

```
host$ for comp in comp{1..10}; do echo ssh $comp parallel-ssh
```

```
$(echo comp{1..10} | sed -e 's/'$comp'//g'); done
```

```
host$ vboxmanage clonevm Ubuntu
```

```
host$ vboxmanage controlvm Linux savestate
```

```
host$ vboxmanage startvm Linux -type headless
```

Submit a report about this exercise that demonstrates execution of the solution, its implementation, and scripts/code.

FOR HALF POINTS: Without creating a script for all 90 combinations (only 1 complete combination)

First, we clone the existing VM with a new name:

```
vboxmanage clonevm N01-Linux-VM --name=N02-Linux-Cluster-1 --register
```

```
glifing@glifing-laptop ~/Escritorio/Cloud Computing$ vboxmanage clonevm N01-Linux-VM --name=N02-Linux-Cluster-1 --register
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-1"
```

We start the virtual machine and we:

update → sudo apt-get update

pssh (parallel-ssh) → sudo apt-get install pssh. I had installed ssh, so I only have to update

```
There are services installed on your system which need to be restarted when certain
libraries, such as libpam, libc, and libssl, are upgraded. Since these restarts may cause
interruptions of service for the system, you will normally be prompted on each upgrade for
the list of services you wish to restart. You can choose this option to avoid being
prompted; instead, all necessary restarts will be done for you automatically so you can
avoid being asked questions on each library upgrade.
```

```
Restart services during package upgrades without asking?
```

```
<Yes>
```

```
<No>
```

```
Restarting services possibly affected by the upgrade:
```

```
Services restarted successfully.
```

```
Processing triggers for man-db (2.8.3-2) ...
Setting up libpython2.7-stdlib:amd64 (2.7.17-1~18.04ubuntu1.6) ...
Setting up python2.7 (2.7.17-1~18.04ubuntu1.6) ...
Setting up libpython-stdlib:amd64 (2.7.15~rc1-1) ...
Setting up python (2.7.15~rc1-1) ...
Setting up pssh (2.3.1-1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
efra@n01-linux-vm:~$
```

We shutdown the machine and prepare the environment to use multiple parallel instances.

We will connect this environments with IP automatically, so we create a new network interface that will be named vboxnet0 automatically (only for hosts):

```
vboxmanage hostonlyif create
```

Now we add DHCP (Dynamic Host Configuration Protocol) for assing automatically the Ips numbers:

```
vboxmanage dhcpserver add --ifname vboxnet1 \
```

```
--ip 192.168.57.2 --netmask 255.255.255.0 \  
--lowerip 192.168.57.3 \  
--upperip 192.168.57.254 \  
--enable
```

Later, we configure network access via the host interface

```
vboxmanage modifyvm N02-Linux-Cluster-1 \  
--nic1 hostonly  
vboxmanage modifyvm N02-Linux-Cluster-1 \  
--hostonlyadapter1 vboxnet1
```

We reduce the memory for later simultaneous execution of instances:

```
vboxmanage modifyvm N02-Linux-Cluster-1 \  
--memory 1024
```

The VM is then started in screenless (headless) mode:

```
vboxmanage startvm N02-Linux-Cluster-1 \  
--type headless;
```

```
glifing@glifing-laptop ~/Escritorio/Cloud Computing$ vboxmanage startvm N02-Linux-Cluster-1 \  
--type headless;  
Waiting for VM "N02-Linux-Cluster-1" to power on...  
VM "N02-Linux-Cluster-1" has been successfully started.
```



We see the ip with the command: `$ ip a` → 192.168.56.101

We try to connect ssh

```

efra@n01-linux-vm:~$ ssh 192.168.56.101
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be established.
ECDSA key fingerprint is SHA256:fL67SVYVF57yd25Xfn6X7Aqp5yFhrBbq2dIzn1wk0bI.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.56.101' (ECDSA) to the list of known hosts.
efra@192.168.56.101's password:
Welcome to Ubuntu 18.04 LTS (GNU/Linux 4.15.0-166-generic x86_64)

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

System information as of Mon Jan 17 10:22:09 UTC 2022

System load:  0.09          Processes:            93
Usage of /:   30.1% of 8.05GB Users logged in:       1
Memory usage: 15%          IP address for enp0s3: 192.168.56.101
Swap usage:   0%

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

167 packages can be updated.
61 updates are security updates.

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jan 17 10:19:19 2022
efra@n01-linux-vm:~$

```

We save the RSA public key

```

cat ~/.ssh/id_rsa.pub | ssh 192.168.56.101 \
"cat >> ~/.ssh/authorized_keys"

```

Now, we clone the virtual environment N02-Linux-Cluster-1 nine times

```

for i in {2..10}; do
vboxmanage clonevm N02-Linux-Cluster-1 \
--name N02-Linux-Cluster-$i --register;
done

```

```

X glifing@glifing-laptop ~/Escritorio/Cloud Computing for i in {2..10}; do
vboxmanage clonevm N02-Linux-Cluster-1 \
--name N02-Linux-Cluster-$i --register;
done
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-2"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-3"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-4"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-5"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-6"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-7"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-8"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-9"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Machine has been successfully cloned as "N02-Linux-Cluster-10"
glifing@glifing-laptop ~/Escritorio/Cloud Computing

```

Because of the space of my disk, I am forced to do only two virtual machines (a three machines clusters)

I'm going to use N01-Linux-VM like host.

```
for i in {2..3}; do
```

```
vboxmanage clonevm N02-Linux-Cluster-1 \
```

```
--name N02-Linux-Cluster-$i --register;
```

```
done
```

We can do the operations with scripts, but my computer can't work it.

Also, we can do it with docker (and create a new network for this cluster):

```
for i in {1..10}; do
```

```
    docker run -it image_id
```

```
done
```