DT211C-3 Introduction to DevOps Lab 01

Setting up a VirtualBox Network Environment

The idea of this lab is to create a basic networked environment for future labs.

It is recommended that you download a copy of VirtualBox and run it on your own laptop. Go to https://www.virtualbox.org/wiki/Downloads and download the binary packages for your platform. You may also want to download the Extension Packs for additional functionality (driver support). Install VirtualBox and Extension Packs if necessary.

Create a New Virtual Machine

For this installation we will use **Ubuntu Server 14.04.5** LTS 32-bit/64-bit (depending on your hardware) or higher version. Please use only the LTS version i.e. Even-number-year.Even-number-release (e.g. 14.04, 16.04, 18.04. Download the .iso from Ubuntu website or from a local mirror (e.g. Heanet http://old-releases.ubuntu.com/releases/14.04.5/)). The file name will be something like ubuntu-14.04.5-server-i386.iso and will be about 700 MB in size. (i386 indicates the 32-bit version; AMD64 refers to the 64-bit version).

In VirtualBox, create a new virtual machine, called ***base***. Two parameters that you must choose for your VM are:

Memory (RAM): 512 MB

Hard disk size: 8 GB

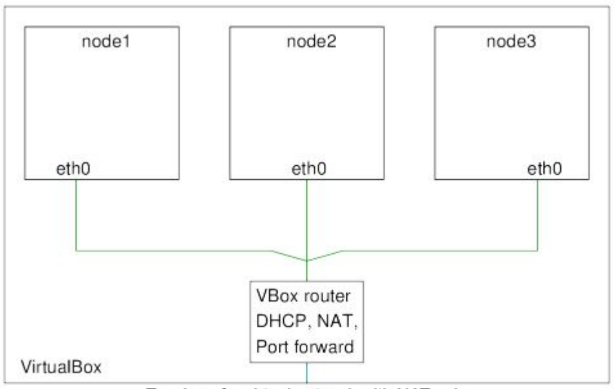
Note: These are recommendations for a minimal setup. In most cases this will be sufficient. If you think you may want to run something intensive on your setup, consider increasing the size of both as necessary. This can be increased afterwards, but it requires a bit more work.

Configure Network Adapter Settings

Before you start the virtual machine, edit the settings for the network adapters. By default, VirtualBox enables only the first adapter, but allows up to four network adapters. We will configure 2 of them now.

Adapter 1 is configured by default to use NAT. This allows the virtual machine to have Internet access, but not act as a server or contact other virtual machines.

For our virtual network we want our nodes to communicate with each other. However to simplify the management of the VMs, they should each also have Internet access and allow remote SSH connections (e.g. so you can connect to the guest via the host using SSH).



Topology for virtual network with NAT only

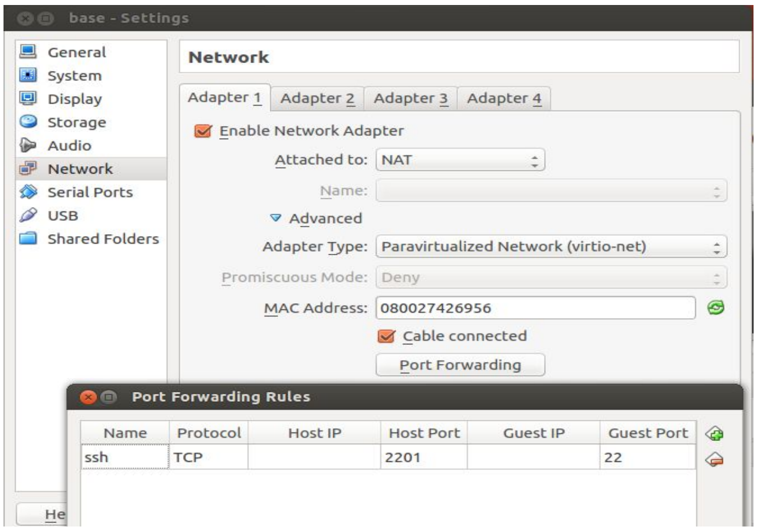
The first step is to give the base virtual machine extra network adapters. In VirtualBox, find the Network settings to see a list of four potential network adapters. Adapter 1 is already enabled and configured to use NAT. Expand the *Advanced* options. There are two things to change:

1. Press the *Port Forwarding* button and add a new rule with the following parameters:
   Name: ssh
   Protocol: TCP
   Host IP: *empty*
   Host Port: 2201
   Guest IP: *empty*
   Guest Port: 22

Port forwarding is setup so that we can access the virtual machine using SSH. You don't have to use port 2201 - any port number unused by the host should be ok.

For example, to connect to the virtual machine from the command line in Linux (I've used port 2200 on my base installation):

```
brian@ulysses:~$ ssh -p 2200 -l devops localhost
devops@localhost's password:
```



Network

Enable Network Adapter
Attached to: NAT
Name:
Advanced
Adapter Type: Paravirtualized Network (virtio-net)
Promiscuous Mode: Deny
MAC Address: 080027426956
Cable connected

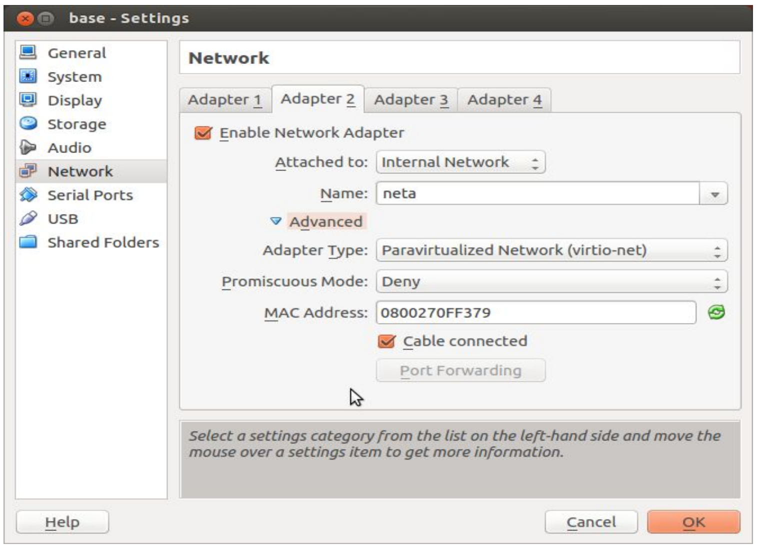
Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
ssh	TCP		2201		22

VirtualBox settings for network adapter 1 on base

Now enable adapter 2 to use the following values:

- Attached to: Internal Network
- Name: ***neta***
- Promiscuous Mode: Deny
- MAC Address: default value (it should be different across all adapters)
- Cable Connection: yes (checked) for Adapter 2
- Port Forwarding: no rules

The figure below shows the settings for Adapter 2:



Network

Enable Network Adapter: ✓

Attached to: Internal Network

Name: neta

Advanced

Adapter Type: Paravirtualized Network (virtio-net)

Promiscuous Mode: Deny

MAC Address: 0800270FF379

Cable connected: ✓

Port Forwarding

Select a settings category from the list on the left-hand side and move the mouse over a settings item to get more information.

VirtualBox settings for network adapter 2 on base

Configuring all four network adapters in the **base** virtual machine will make it easier in configuring nodes cloned from the base.

When you start the VM for the first time, VirtualBox prompts for a disc (iso) to boot from. Select the .iso image you downloaded previously.

Install Linux:

The Linux installation is rather straightforward. Most default options can be chosen or you can chose values that suit you (e.g. language, keyboard, time zone). There are however several important values to be chosen:

- On the first install menu (the black screen with the menu in the middle, which is displayed after you've chosen the language), press **F4** and select ***Install a minimal virtual machine***.
- Username and password: Chose **devops** and **devops**. The virtual machines are only intended to be used on your computer, so securing them with a strong password does not bring many benefits; choose a simple password and re-use it when necessary (e.g. MySQL, root user).
- No *automatic upgrades* in order to keep the versions of software used in virtual networks the same for different users.
- In the *Software Selection* menu, select: ***Basic Ubuntu server*** and ***OpenSSH server***

Start and Login to the base virtual machine

At the end of the install procedure you are presented the option to *Continue* to reboot into the virtual machine. Do so, and login using the username and password you created, e.g. *devops* and *devops*. You should see a prompt like:

```
devops@base:~$
```

meaning you are logged in as the user *devops* on the host *base* and currently in your home (~) directory.

Install and Update Software

Once Ubuntu is installed and running, you should update the existing software packages and install some additional package.

Use apt-get to update and install packages as follows:

```
devops@base:~$ sudo apt-get update

devops@base:~$ sudo apt-get upgrade

# Essential Install

devops@base:~$ sudo apt-get install vim nano wget build-essential man-db
manpages manpages-dev

# Optional Install

devops@base:~$ sudo apt-get install iperf lynx iputils-tracepath traceroute
tcpdump ethtool telnet iptables iputils-arping ettercap-text-only dnsutils
iptraf whois
```

Configure Networking on the Guest

The base system will have multiple network interfaces, however we may not use all of them:

1. eth0 is used for normal Internet access. It is configured by the special DHCP server provided by VirtualBox. There is nothing to change with this interface; it should be set correctly already.
2. eth1 is used for the internal network (to connect to other virtual machines). We will manually set the address for this interface.
3. eth2 is used in the same way as eth1. However initially in VirtualBox we will set the cable to be disconnected. This interface will only be needed on some virtual machines (usually routers).
4. eth3 is used in the same way as eth2.

Now configure addresses for the interfaces. The first interface, eth0, should be configured to use DHCP. VirtualBox includes a DHCP server that allocates IP addresses to guests.

The other interfaces should be configured with static IP addresses.

The configuration of the eth1 interface is performed in the file /etc/network/interfaces. Edit the file with nano by typing:

```
devops@base:~$ sudo nano /etc/network/interfaces
```

The lines for the lo and eth0 interfaces do not need changing. You need to add new lines for the eth1 interface. Add the following lines to the bottom of the file:

```
auto eth1
iface eth1 inet static
   address 192.168.1.11
   netmask 255.255.255.0
   network 192.168.1.0
   broadcast 192.168.1.255
   post-up route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1 dev eth1
   pre-down route del -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1 dev eth1


#auto eth2
#iface eth2 inet static
  # address 192.168.2.2
  # netmask 255.255.255.0
  # network 192.168.2.0
  # broadcast 192.168.2.255
  # post-up route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.2.2 dev eth2
  # pre-down route del -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.2.2 dev eth2


#auto eth3
#iface eth3 inet static
  # address 192.168.3.3
  # netmask 255.255.255.0
  # network 192.168.3.0
  # broadcast 192.168.3.255
  # post-up route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.3.3 dev eth3
  # pre-down route del -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.3.3 dev eth3
```

What does this do?

When the computer (or more precisely the network interface eth1) starts, it is given the IP address 192.168.1.11 with subnet mask 255.255.255.0. The post-up line also adds a route to the routing table when the interface starts (the last line removes this route when the interface goes down). This route will act as the default route inside the virtual network (i.e. to all networks within 192.168.0.0/16).

The set of lines for eth2 and eth3 are all currently commented out - they do nothing. They follow the same format as for eth1. They are included for convenience when you create a virtual machine with two or three internal interfaces, e.g. router. You can just uncomment the lines and edit the addresses as needed.

Note that the all lines are not necessarily needed for all interfaces. For example, the post-up and pre-down lines may not be needed when configuring a router. Also the values used may not be appropriate for your network. The lines provided are just defaults - they should be changed appropriately depending on the virtual machine/network you are trying to configure. For a description of the interfaces file type man interfaces on the command line.

Restart the Virtual Machine

Reboot the VM using the reboot command:

```
devops@base:~$ sudo reboot
```

Prepare Interfaces for Cloning

The next step will be to clone this base virtual machine to create multiple other virtual machines. When we clone, VirtualBox can change the hardware (MAC) address of the network interfaces (so the new machine has a different hardware address to the original machine - this is desirable). However, Ubuntu is currently configured to give interface names (such as *eth0* and *eth1*) based on hardware addresses. If we changed the hardware address, then Ubuntu will not be able to configure the interface upon booting, which is a problem. To avoid this we need to disable the current mapping from hardware address to interface name.

NOTE: If this rule does not exist in your installed version, then you have nothing to do here.

Edit the file *`/etc/udev/rules.d/70-persistent-net.rules`* and comment out the two lines that start with SUBSYSTEM by inserting the # character. (Alternatively, you can delete the lines).

```
devops@base:~$ sudo nano /etc/udev/rules.d/70-persistent-net.rules
```

Now shutdown the VM using the poweroff command:

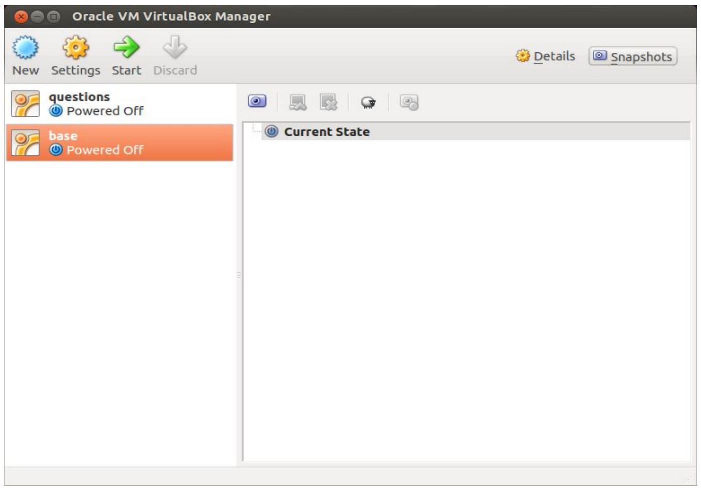
```
devops@base:~$ sudo poweroff
```

It's important that you comment out the lines in */etc/udev/rules.d/70-persistent-net.rules* just before you poweroff and start the cloning with VirtualBox. If you start the virtual machine again before cloning, then Ubuntu will automatically add the lines to */etc/udev/rules.d/70-persistent-net.rules* and you will have to remove them again. So now that the virtual machine is off, move to the next step of cloning it.

Cloning with VirtualBox GUI

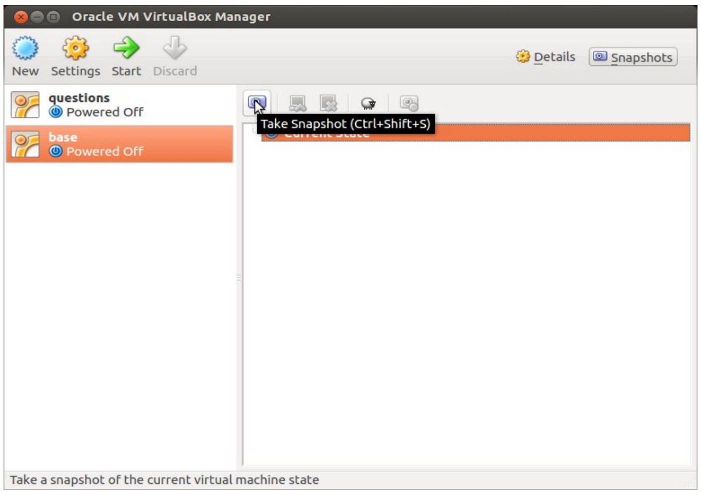
First create a snapshot of the base virtual machine. This snapshot, which is a version of the base virtual machine at specific time instance, will be cloned.

Select the base VM and then click the *Snapshots* button:



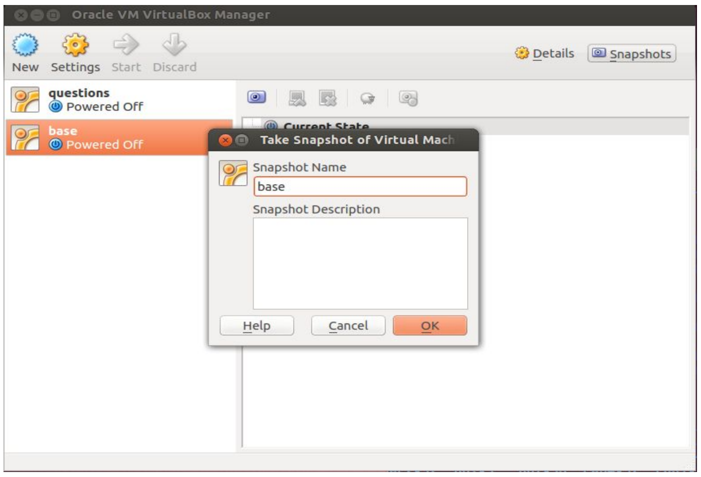
VirtualBox cloning of base VM: View snapshots of base VM

Click the *Take Snapshot* button:


Oracle VM VirtualBox Manager
New
Settings
Start
Discard
Details
Snapshots
questions
Powered Off
base
Powered Off
Take Snapshot (Ctrl+Shift+S)
Take a snapshot of the current virtual machine state

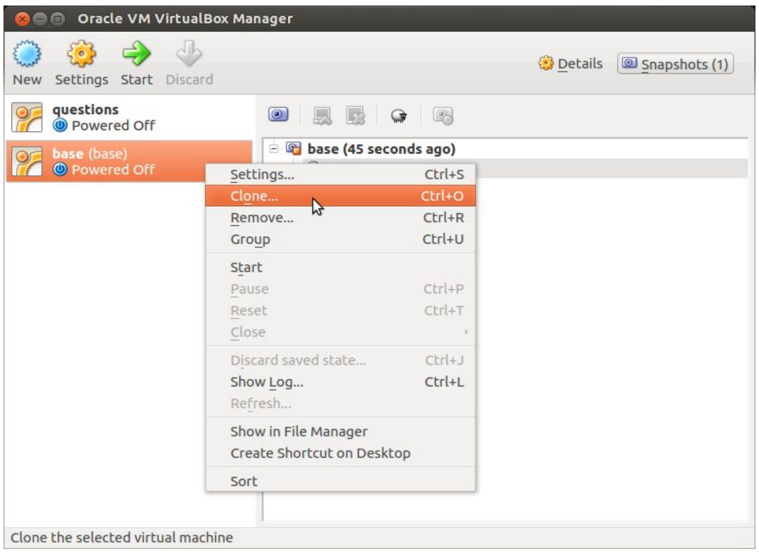
VirtualBox cloning of base VM: Take a snapshot of base VM

Name the snapshot ***base*** (i.e. same name as the VM):



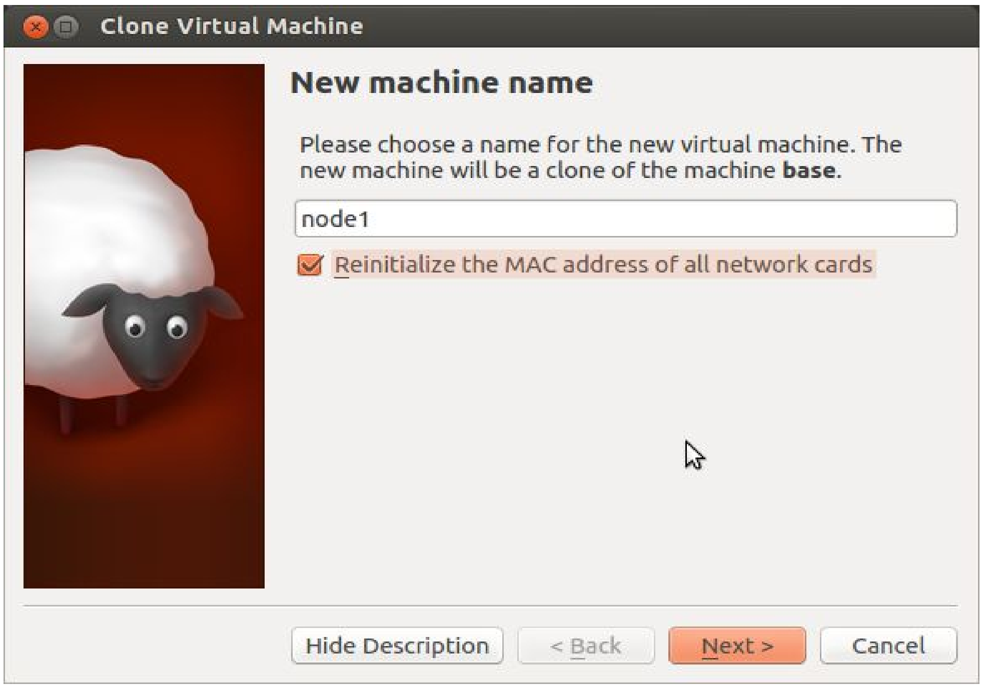
VirtualBox cloning of base VM: Name the snapshot base

Now that the snapshot is created, right click on the base VM and select *Clone...*:



VirtualBox cloning of base VM: Clone the base VM

Name the clone *node1* (and subsequent clones, *node2*, *node3* etc.) and check *Reinitialize the MAC address of all network cards*:


Clone Virtual Machine

New machine name

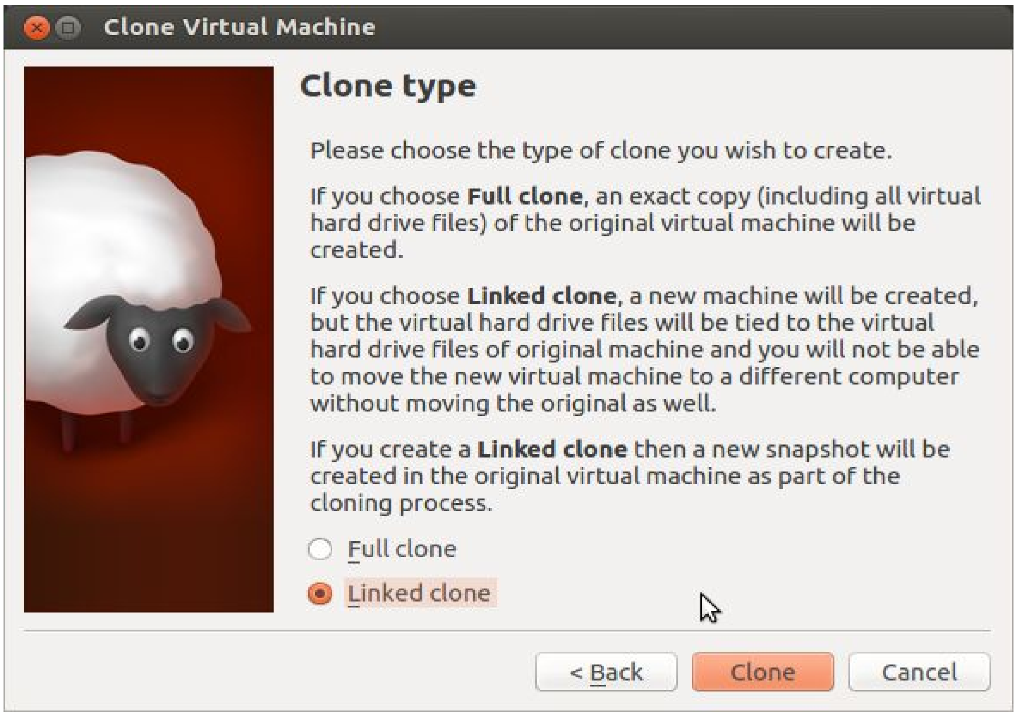
Please choose a name for the new virtual machine. The new machine will be a clone of the machine **base**.

node1

☑ Reinitialize the MAC address of all network cards

VirtualBox cloning of base VM: Name clone and reinitialize MAC addresses

Select *Linked clone* as the clone type. With a linked clone, only the *changes* between the cloned VM and the base VM hard disks are saved. With a full clone, the entire base VM hard disk is copied. Using a linked clone saves significant hard disk space on the host (at the expense of performance if the clone is used for a long time and many disk changes are made compared to the base).


Clone Virtual Machine

Clone type

Please choose the type of clone you wish to create.

If you choose **Full clone**, an exact copy (including all virtual hard drive files) of the original virtual machine will be created.

If you choose **Linked clone**, a new machine will be created, but the virtual hard drive files will be tied to the virtual hard drive files of original machine and you will not be able to move the new virtual machine to a different computer without moving the original as well.

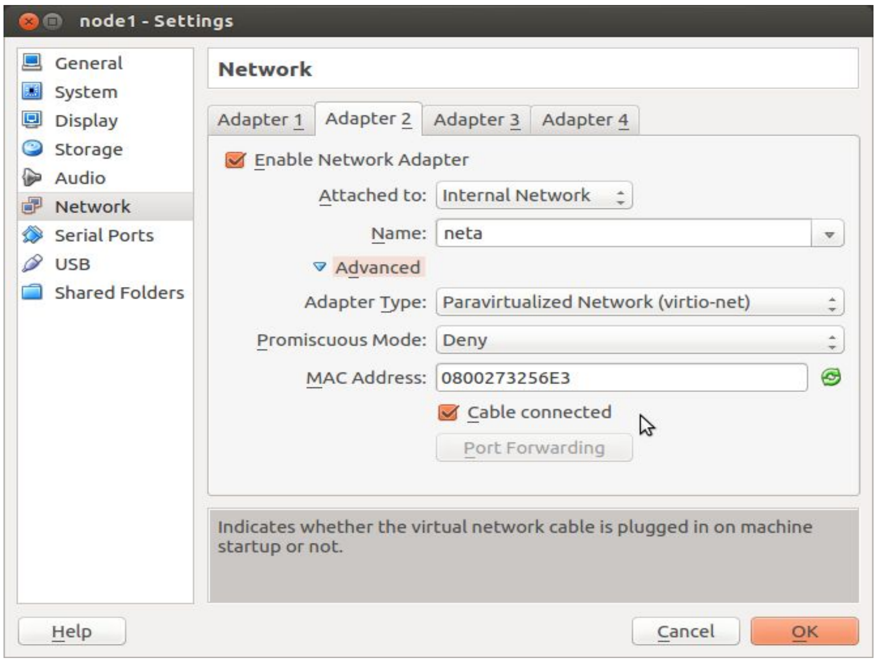
If you create a **Linked clone** then a new snapshot will be created in the original virtual machine as part of the cloning process.

Full clone [✓ Linked clone]

VirtualBox cloning of base VM: Linked clone

Now the new virtual machine has been created, you may need to configure the network adapter settings. This depends on what role this node has in the virtual network. Recall that in our base virtual machine we configured four network adapters:

- Adapter 1: NAT, used for connection to host and internet.
- Adapter 2: internal, used for connecting to other nodes. Initially configured to use internal network *neta*.
- Adapter 3: internal, used for connecting to other nodes. Initially configured to use internal network *netb* but with the cable disconnected.
- Adapter 4: internal, used for connecting to other nodes. Initially configured to use internal network *netc* but with the cable disconnected.


node1 - Settings

Network

✓ Enable Network Adapter

Attached to: Internal Network

Name: neta

Advanced

Adapter Type: Paravirtualized Network (virtio-net)

Promiscuous Mode: Deny

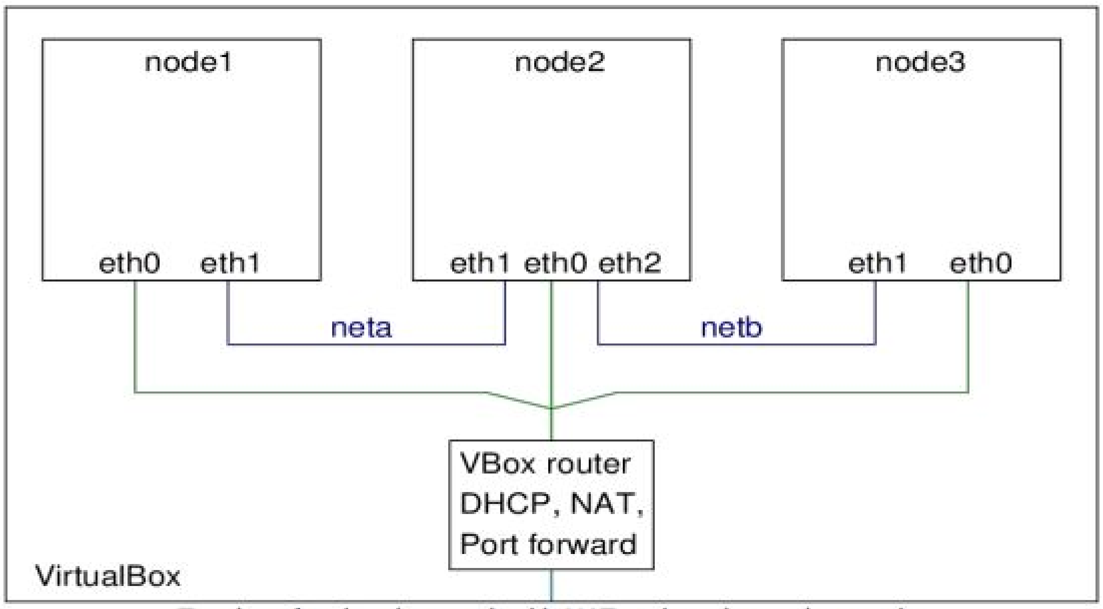
MAC Address: 0800273256E3

✓ Cable connected

Indicates whether the virtual network cable is plugged in on machine startup or not.

Network Adapter Settings of Cloned VM: set the network name and cable connected to appropriate values

You may need to change the settings for adapters 2, 3 or 4, depending on how you intend this node to connect to other nodes in the virtual network. For example, if this node is node1 in the network below, then no changes to the adapters are needed. However if this node is node2, then you need to connect the cable (by checking Cable connected for Adapter 3).



Topology for virtual network with NAT and two internal networks

In summary, for the new virtual machine, configure the settings for adapters 2, 3 and 4 by setting the appropriate internal network name (e.g. *neta*, *netb*, *netc*) and connecting/disconnecting the cable depending on whether the adapter is needed or not.

The final change is to set the port to be used for port forwarding on adapter 1 (that NAT interface). For the base virtual machine the port 2201 was set to forward to the SSH port 22. You should change the port so that it is unique for each node. For example, on *node1* use port 2201, on *node2* use port 2202 and so on. (You can use other ports; they don't have to be 2201, 2202, etc.)

Now you can start the new virtual machine and proceed to the next step to configure it.

Configure Each Node

When you have created a node by cloning the base virtual machine, start and login to that node. There are several steps needed to finalize the node configuration. Also, depending on the role of the node in the virtual network, you may need to make further configuration changes.

First change the host name in both the */etc/hostname* and */etc/hosts* files. You can edit them manually with the text editor *nano* (also using *sudo*) or change them directly with the following lines (replacing *node1* with your node name):

```
devops@base:~$ sudo sed -i 's/base/node1/' /etc/hostname
devops@base:~$ sudo sed -i 's/base/node1/' /etc/hosts
```

Next set the appropriate addresses for your node interfaces. Open */etc/network/interfaces* with nano and change the values to suit your node. Once you have saved the file, restart the interface(s) that you changed by either rebooting or:

```
devops@base:~$ sudo ifdown eth1
devops@base:~$ sudo ifup eth1
```

Reboot and Test

Now reboot your node, login and test. Then repeat the steps of cloning and configuring for other nodes to create your virtual network.

The Lab is completed when . . .

The associated Google form is completed.