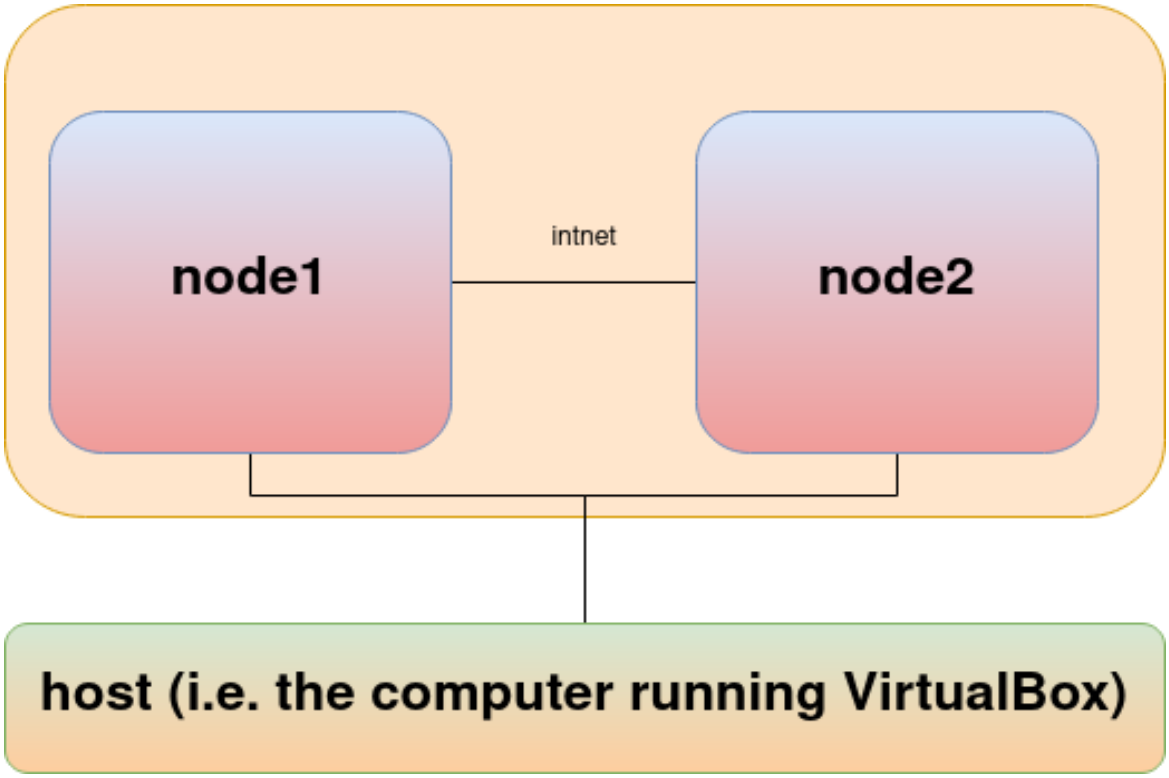
# Introduction to DevOps

## (27th January 2021)

# Setting up an Ubuntu Linux VM

## The goal of today's lab

We're trying to set up two virtual machines and network them together. We would like to set up a network with the following topology:




Here we see two VMs (node1 and node2), all of which will be connected to the host (the computer running the VMs, i.e. your own computer) via a network. This will allow the each VM to connect to the internet. In addition, the two nodes are connected to each other via an internal network, called intnet.

## Installing VirtualBox

The virtual machine/virtualisation software that will be used for this lab is called VirtualBox. You should download and install VirtualBox if it isn't on your computer already. The main download site is:

https://www.virtualbox.org/wiki/Downloads

If you know of any other way to install it (for example via a package manager on a Linux system), you can use that method also.

## Downloading Ubuntu

To keep the VM small and fast, we're going to use the server version of Ubuntu, since we only need a basic Linux system with a command line interface.

Download an ISO file for the latest version of Ubuntu Server from

https://ubuntu.com/download/server

We will be installing this on the VM later.

## Create and cofigure a base VM

Since we will be setting up 2 virtual machines, it's a good idea to set up a base VM, and then clone it twice to create the nodes. This avoids having to install Ubuntu twice, which would be far too time consuming.

Run VirtualBox, and click 'New' at the top to create a new VM. You will be asked to make a few choices when setting up the VM:

- Use 'base' as the name of the VM
- Set the type to 'Linux' and the version to 'Ubuntu (64-bit)'
- Unless you are using a very low-end computer, try to set about 1GB (1024MB) of memory for the VM
- Create a hard disk that's about 10GB

The two nodes will each have 2 network adapters – one to connect to the host, and one to connect to the other VM. To make this work, we need two network adapters in each machine (one to connect to each network). This should be set up before you start up the server -- click 'Settings' and then 'Network' to change some of its network settings:

- In the 'Adapter 1' tab, make sure:
  - 'Enable network adapter' is checked
  - 'Attached to' is set to 'NAT'
  - 'Cable connected' is checked (you might need to click 'Advanced' to see this)
  - Click 'Port forwarding' and add a new entry to the table – give this entry the name 'ssh', the protocol 'TCP', the host port '2200' and the 'guest port' 22
- In the 'Adapter 2' tab make sure:
  - 'Enable network adapter' is checked
  - 'Attached to' is set to 'Internal Network'
  - Set the 'Name' field to 'intnet'
  - 'Cable connected' is checked

## Installing Ubuntu

Save the settings and then run the VM by clicking 'Start' at the top of VirtualBox. The first time you run the VM, you will be given the option to choose an ISO file to boot from. Use this to select the Ubuntu ISO you downloaded earlier.

You can generally just use all of the default settings when installing, but there are a few things to look out for:

- Make sure you select the right keyboard layout – Ubuntu will default to an American keyboard, so make sure you select the Irish layout (or whatever keyboard you have on your computer).
- Make sure you remember the name you gave the server, as well as the username and password you set up. Since you will be the only person using the VM pick a password you'll remember instead of one that is secure! :-)

When the installation is complete, you should be able to reboot the VM and log in to it.

## Configuring the VM

Once you can log in, there's a few things you should do to set up the VM.

First, install some useful software that we'll be using later. Use the following command:

```
sudo apt install nano wget build-essential net-tools ifupdown
```

Modern versions of Ubuntu use a custom-developed network stack, which works differently to most other Linux distributions. It's a good idea to switch it back to the network stack used in most other Linux distributions. To do this, start off by opening the network interfaces file with the following command:

```
sudo nano /etc/network/interfaces
```

This should open an empty file. Add these lines:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
```

Press Ctrl + O to save, and Ctrl + X to exit.

Next, open the Grub configuration file:

```
sudo nano /etc/default/grub
```

Near the top of the file, there should be line that looks like:

```
GRUB_CMDLINE_LINUX=""
```

Change this line to:

```
GRUB_CMDLINE_LINUX="net.ifnames=0 biosdevname=0"
```

Then update Grub by running:

```
sudo update-grub
```

Save the file and then reboot the VM with the following command:

**`sudo reboot`**

When you've rebooted, login and run:

**`ifconfig`**

You should see the network connection using a device called eth0 if everything went well!

## Cloning the VM

Cloning lets you make an exact copy of the VM. Before cloning, you'll need to shut down your VM. This can be done at the command line with:

**`sudo poweroff`**

When back at the VirtualBox GUI, right-click 'base' and select 'Clone'. You will be given a number of options to choose from for the cloned VM.

- Call the clone 'node1'
- Set the 'MAC address policy' to 'Generate new MAC addresses for all network adapters'
- Choose 'Linked clone' when prompted to choose a clone type. This will make the cloned VM smaller
- Repeat the above two steps again to make node2

## Configuring the cloned VMs

There's a small number of changes we need to make to the two VMs to set up the intnet network.

We'll use static IP addresses in for the internal network, so we don't have to keep looking up the IP address every time we boot up our VMs. To do this, boot up the two VMs, and edit their /etc/network/interfaces file. For node1, add the following lines (note the indentations!):

```
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
```

Add the following to the file in node2:

```
auto eth1
iface eth1 inet static
  address 192.168.1.12
  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
```

You'll need to restart each machine after you've done all of this.

## Testing

At this point, you should be able to ping nodes2 from node1. Start up both VMs again, and run the following on node2:

```
ping 192.168.1.11
```

Remember, 192.168.1.11 was the IP address we just gave to node1. If you get IP packets back, this is working. Now try pinging node2 from node1:

```
ping 192.168.1.12
```

If this works, you're done!

## Submission

Submit via Brightspace before midnight 8th February.

The submission should consist of screenshots of:

- The VirtualBox manager screen, showing the three VMs: base, node1 and node2 (3 marks)
- The output from the ifconfig command for both node1 and node2, showing the network is connected (3 marks)
- The output of the ping command from one of the two nodes, showing the other node being pinged successfully (2 marks)