# Week 6 Cloud Computing - Implementing OpenStack Transcript

Hello, welcome to week six of the virtual and cloud computing module. I'm Mark Comerford.

I'll be your guest lecturer for this session, and we will be covering a few things today. We'll be covering how to deploy equal to server in VirtualBox and then deploying the open stack environment within that virtual machine.

Before we begin, we'll just cover a few things about OpenStack and what it involves. It's underlying architecture and open stocks cloud system platform is used to control and pull computing power, provisioning of storage, and the virtualization and provisioning of network resources. In a data center environment, these resources are usually managed and provision through application program interfaces. Lots of developers make certain modules available to open start using these APIs. That we have available to us. And they can do all kinds of different things. It's very modular, and it's very adaptable to customer requirements. So, when we manage OpenStack, we usually manage it through what's known as a dashboard.

This is a web based interface. Usually with the web based interfaces you can control most, if not all of the underlying aspects of OpenStack. And this is the same for other virtualization platforms such as Amazon, AWS, or Microsoft as your open stack has its own implementation of this. It arrives great flexibility, because they're all they can create and destroy what's known as instances, instances are virtual machine templates. And we can basically create and modify our instances as we like. The instances can have various different amounts of RAM, processor cores, storage. All these things are adaptable to customer requirements. And when we deal with OpenStack, what we're dealing with is what's known as infrastructure as a service. Now you have already covered some of these aspects in previous lectures. So this is really a practical demonstration of what an IS platform would look like actually in the real world. An OpenStack is basically broken down into numerous services. Now, I'm not gonna go over each one of these, but we'll cover the three fundamental aspects of Opentack, one of them is compute. This is where the virtual machine computation is involved processing.

Then we have the networking aspect of OpenStack. This is where one virtual machine may communicate with another, the data for the decisions routing are made on this particular stock. Then we have the storage part of the deployments. This is where obviously the backing store for your virtual machines is held on. Basically we have always three different processes that work in tandem with each other. This is what creates our infrastructure. There are other services that feed into this, that are essential for operation, but these are the. Basically the core aspect of OpenStack and what are needed to make it function. So before we start, I just thought I'd quickly go over some of the different things you can do with OpenStack. Now, we not actually going to cover these in this tutorial but I thought it'd be interesting for you to know so you are aware of these things that you can then you go away and experiment with them.

So you can see on the OpenStack website on the software section of the website, there is a sample configuration option. So we click on this and gives you different modules that we can actually implement. Afterwards, if we want to, and these are all optional. We don't have to have them but they are there. If we need to expand our functionality, we just, for example, go to the first option Cinder, dealing with block storage. Using APIs to interface with your OpenStack infrastructure and

we can use basically LVM logical volume management to manage our storage facilities within the OpenStack. And which tells us gives us a lowdown of what we actually need to have these models running for this particular module. It depends on keystone to operate. It also tells you about related services that will may be useful if you're implementing this module.

If we go on the documentation here, it'll tell us what it is and what actually involves what we need to actually implement these modules. So it's interesting to see what these modules are about. And there's lots of development going on, a lot of open standards that we can follow which developers used to make their own modules and to adapt OpenStack. So the actual options are quite diverse. We can do a lot with them, and it just opens up functionality to LVM logical volume management.

And though I was just to do different things that in its default form we can't do. So moving on now we're going to get started with the actual deployment of OpenStack.

So, to begin with, we need to actually deploy underlying operating system. Now ordinarily, OpenStack will be deployed on a bare metal environment. That is, they won't be any virtual machine in between OpenStack on the actual machine it will be running on in this instance will actually be wearing virtual box. So it is one layer of virtualization that we have to deal with which wouldn't be there otherwise. So, the performance of open stack will be inhibited by the use of virtual box. Unfortunately, this is the only way we can demonstrate the use of open stock in this environment. By all means, if you have a actual computer that you can spare with that you can install OpenStack and you can follow same tutorial but just natively install it on a real installation. This is just for flexibility so that you don't have to mess with your existing machine if you are limited with resources, but it is encouraged actually run the operating system the Ubuntu operating system on an actual real machine. So present you may or may not have VirtualBox 6.1 installed. Now, for this particular module, we need to actually have 6.1 installed, because what we need to do is enable a feature called nested virtualization. Now this feature was only implemented in VirtualBox very, very recently. And it's likely you will have the old version of VirtualBox, possibly 6.0.

So what you'll have to do is download the VirtualBox 6.1 from their website. Now if we go onto the website, now depending on what platform you're using, Linux or Windows, we can basically download the correct Implementation for our operating system so download the right one, install it, you may be prompted to upgrade to the newer version. They may be older extensions for 6.0 just to upgrade them. Just go along with the upgrade process. You may need to reboot your computer and then once that is up and running, then we can allow a virtual box will appear now, just close this window.

Now as you can see here gone, help about on 6.1. So this is what we want to see, make sure you're on this. Otherwise, we're not going to have this reading properly. The first thing you're going to do is go to machine and then go to new. This will add a new virtual machine to your eventually. Next we want to select Linux and then next we want to select 64 just call it a name, keep them to server. And then ideally, we want to have the machine running with sufficient RAM. Now this is obviously going to be limited by what you are running currently in your machine. So just change the actual RAM to whatever as you can realistically, don't over do it so that your main host machine is oversaturated with lack of RAM. So this computer has eight gigabytes of RAM. So I'm gonna just assign the actual virtual machine to half of what I actually got. So if I set this to 4096, that's four gigabytes of RAM for our virtual machine. So we'll go ahead and click Create. Next, we'll be prompted to create a virtual hard drive. Now, ten gigabyte is probably is gonna be sufficient, but I'm gonna put it to 20 just to be safe. Again, it depends on what actual space you have left on your device, set this accordingly. So if we correct create num and what you'll see is the virtual machine will be actually showing in the virtual machine list.

So now what we need to do is we need to enable nested virtualization. The reason why we need nested virtualization is because, by default, when we run a virtual machine, the actual virtualization extensions within the processor of your machine aren't passed on to the virtual machine. Now, the OpenStack environment requires virtualization enabled. So as you can tell if OpenStack is running within a virtual machine, it won't know it's actually a virtualized environment. So it will actually not be able to run a virtual machine within a virtual machine. This is why we call the actual implementation the nested virtualization is one machine running within another ordinarily, this is not good practice because you are basically holding two layers of performance degradation in the way of the actual hard underlying hardware. But this is just for demonstration purposes. So if we go to settings, go to system and go to processor. We can see here we have a few options. Now if we select the Enable Master to VT-X AMD-V, that will enable nested virtualization. We may also want to increase the number of processor cores with our virtual machine one may not be sufficient. So in my instance, I have eight virtual CPUs actual real CPUs and then I want four virtual cores within this VM. This just gives processing power to be able to do everyday tasks.

We click on OK. And then we have then this correct settings for the rest of virtualization.

The next thing we're gonna do is download the Ubuntu ISO. Now, we're using Ubuntu server because it will likely have those services we need to deploy OpenStack. Because you get a way with using Ubuntu desktop but I don't know what the actual outcome will be and it may not run as expected. So, I'm going to follow good practice here and we're going to use Ubuntu to serve this particular module. So let's go into Ubuntu server click on the 18.04.4 link. And then you'll see here and it's starting to download the ISO. So let the ISO download, and then we'll all load the VirtualBox up, and map this image in VirtualBox. Okay, now that we've downloaded the ISO,we need to actually point this ISO in the virtual machine. So that when we boot it up, we'll actually boot the Ubuntu server setup. So it's pretty straightforward. All we do is go back to settings, go to storage.

And you'll see here you'll have two different storage devices. We have control IDA with a little CD drive. This is what we actually want. This is where we load the disk image. So basically if we just we just click on the empty option here. You're given another option here attributes. So all we do is just clicking this little CD here and click Choose and create a virtual optical desk. Then we use to create at desk image.

Now I've saved my Ubuntu server ISO to my desktop and you can see here it shows up as the only file that slides because it knows damn only usable optical disk image files which ISO files are part of that category. Now you may have download this to other locations so just navigate to wherever you've got that ISO located. So we flipped the ISO and click on Open and then what we do is then we have this showing here as the actual ISO we've just selected. So we just click then on choose and then we can see here now this ISO has been assigned to our IDA controller which will then boot up into the Ubuntu setup environment to actually cover next. So the next thing we need to do before we actually start the setup board for Ubuntu is change the network settings to work to how we want. The actual OpenStack implementation to run. By default the network setting is set to NAT, Network Address Translation. Now NAT is ordinarily, Works fine in any ordinary implementation of a virtual machine. But when we actually want to connect from this host machine to our virtual machine, it becomes problematic because there's no end-to-end connectivity. So only the traffic within the virtual machine can get out to the Internet. But the reversal traffic, say, if wanted to access the IP address of the virtual machine from my host machine, this doesn't work very well, or not at all. So what we want to use is something called a bridged adapter. Now, the bridge adapter uses the actual underlying network adapter used on your machine to access your network and the Internet. This may vary depending on what actual connection you're using. So for example, if you're using a laptop with a wireless card in it to connect your network, then you want to select the actual

adapter you want to bridge to the virtual adapter using your wireless card. In my particular instance, I'm using a wired Ethernet connector on my machine. I've got a desktop machine. So I'll be using the actual Ethernet connection here. But as I say, this will vary depending on your machine. So what this will do is it'll bridge our virtual adapter to my real physical adapter to enable a realistic and open connection to my network. So it's fully visible, there's no blocking of using that.

So next process is to actually start the virtual machine and get the setup up and running.

So it's a pretty straightforward task, all we need to do is press the Start button here. We'll be presented with another window which shows the virtual environment booting up. As you can see here, the actual ISO now is booting up as though we have a real CD in our CD drive and we're boosting Ubuntu upon actually a real server. But this is actually virtualized. So we'll just wait for this to boot up, shouldn't take too long. The one thing to note with Ubuntu server is the actual setup processes in a text-based environment is not graphical like Ubuntu desktop. But don't worry too much about this, it's still pretty straightforward to follow, you're just using the keyboard rather than a mouse. But don't get too disheartened, it's pretty straightforward. I'll walk you through it anyway, so it's not a big problem. So the text-based interface of the Ubuntu server is relatively straight forward to use, it's just the arrow keys on your keyboard. Most options it's up and down, but you may get some options where it's left or right. In this instance, we're gonna select the language,

choose whatever language you wish to select. And then, you just basically press Enter on the option you wanna select. So if I press Enter on English, and then it will go to the next stage. So just go up with your up arrow, and then just select which keyboard layout you want to have installed. So I'm just gonna select the UK because my keyboard is mapped out to that particular layout. And then, we press the down arrow, and then once the option's hovered over, it'll go green. Just press Enter over that, and press Done. And then, it will go on to the next stage.

Now, the next section of this install will show you the actual network adapter setup.

So if the actual bonded bridged adaptermhas actually been successful, we'll see the IP address of our virtual machine. Been given an IP. Now usually, this will be from a DHCP server. And this DHCP server will be running on your router at home. So if you've got something on the lines of a 192 dot 168 address, then you will know pretty certain that it's actually been successful. Of course, your network may be different, you may have a different IP address structure to what you see here. But for the most part, if you see an IP address, you know it's working. If you don't, then you have to troubleshoot. Have you actually got a network connection functioning properly? Is it getting all the information from your DHCP server? So double-check this. If you've got an IP address showing here,

then it's sure fire way of knowing that it's actually working. So once we see this is actually fine and it's working, go to press down and Done. Make sure Done's hovered over, and press Enter. Proxy addresses, you don't need to worry about, just ignore them. Mirror address, also, you don't need to worry about that. Just press Enter. Next section will be about using the actual virtual disk you set up earlier on. Just select entire disk. We don't need to mess around with all the other options. And then, it'll ask you which disk to install it to. There's only one 20 gig drive that we set up earlier on.

Press Enter to that. And then, you'll basically summarize what you've just selected. It's mostly automated, you don't need to actually change any other options. Press Enter and then Done. And then, it says do you want to continue? Now, it'll automatically be hovering over No. We just press the down arrow. Go an Continue, and then press Enter to continue. Okay, next section, we are entering our login credentials. So this will basically be what you see on the screen. So I'm just gonna enter in my information here. Server name, it can be whatever you want.

I'm gonna call it OpenStack, but it can be anything you desire. Same with the username, just choose whatever you want, doesn't have to be specific. I'm just gonna abbreviate my name. And then, just enter in a password and confirm it, make sure it's correct. And then, press the down arrow and press Enter on the Done. Next section, installing an OpenSSH server. We can select this, press the Space bar on that, and it'll mark it with an X. You don't have to have OpenSSH, but if you're familiar with Secure Shell, you can actually use Secure Shell clients within your host machine and access the virtual machine from this. Makes administration of the command line quite a lot easy, cuz you can copy and paste commands into the actual shell across the network. So install it, and if you're familiar with using OpenSSH, then I would certainly encourage you to use this. If not, then just use the actual terminal window here within virtual boxes will suffice. So you hover to Done and press Enter.

Next, we're presented with all of the packages that we can install within the Ubuntu server environment. Now, we don't actually need any of these options, these are just there if you want to expand your functionality of Ubuntu. But we don't actually have any services within this specific option that we need to actually install. These prerequisites for OpenStack are actually installed when we deploy it within a pre-built script provided by OpenStack themselves. So if we just keep going down, we can actually use Tab as well to skip between. If we see here, press Tab, we can skip between major options very quickly. So just get rid of that. So if we just hover over Done and press Enter. And then, what will happen is then Ubuntu will setup, as you can see here, and we will have a fully working system. So I'm just gonna leave this running. It may take a while to install. Another thing to take into consideration is that when Ubuntu is being setup, it will download updates and packages. So just bear in mind this may take a while depending on how many packages need to be downloaded and installed. It also depends on the speed of your machine and the speed of the hard drive in which the virtual machine is running on. If it's a mechanical hard drive, this could take a while. SSDs may not take so long, so just bear this in mind. And we'll carry on after this installation is finished. Okay, now that we have the installation finished, we can now proceed to reboot. We press Enter on the reboot option, and it'll ask us to remove the installation medium, if we just press Enter on that. And I should just reboot back into the startup. If we just wait a minute now and it will actually boot up into the login prompt. We don't use the username and password that we set before in the setup, and we log into the Ubuntu environment with this information. So we just let this boot up now, shouldn't take too long.

You'll get all this tech showing up. This is just the initial startup. And as you see here, we are presented with an OpenStack login prompt. So if we just enter in the same credentials as what we entered in before in the setup. Just ignore all this text that's showing up here, this just happens on the first initial setup. Once you reboot it all won't come up again. So when we login successfully,

we can see here this is the command line interface with Ubuntu. We're not dealing with any graphical interfaces with this particular tutorial. This is all console based, as I was saying before. So when we first load up, we see certain statistics about the Ubuntu environment. We can see the actual IP address that we've been assigned to this machine, and the amount of storage space left on the drive. So we can see here the 28% of the actual hard drive being used by the virtual environment. So if we carry on, though, and we'll proceed to download the actual OpenStack software, and we can proceed with the deployment of this software. So the first thing we want to do before we start anything is to ensure that Ubuntu is actually up to date. So there's two commands we type for this. It's apt-get update, and apt-get upgrade. Now, apt-get is a package manager within Ubuntu which allows you to install, remove packages and update. There's also another command that we need to be familiar with, it's called sudo. I type it in. Basically, sudo enables a non-privileged user, that is, a user that can't make changes ordinarily, to be able to make changes as a super user.

Now, a super user is a user who is able to make changes to the underlying system. The main lowest level account that we can use to make changes is called the root. Root is the penultimate account for this, and we can do whatever we want. Sudo basically enables the current user to have most of these privileges. So right, if we get on and type in apt-get update. Need to type in our password for the username that we created. And what this will do is it will grab all the latest package information from Ubuntu's servers. And then, what we can do is if we just press the up arrow, we can repeat the command that we just typed in. If we just type in then upgrade. And as you can see here, there are a few packages which need to be upgraded. And yeah, if we just press Yes to that, Y, and then press Enter. And then, it'll download these packages and it will install them.

It may take some time, so I'll just pause this and resume once these have installed.

So now that these packages are installed, we can then download OpenStack from their website.

So what we use to get OpenStack is a package within Ubuntu called Git. Now, this is probably something that you may have heard of in the past, in other instances, within the Linux distribution. Git is basically a distributed version control system. And what it does is it tracks changes in source code during software development. It's good for coordination work amongst programmers, and

it can detect changes in a files, including regressions in software, there's maybe a bug. We can file bug reports, and then those changes then can be tracked within the software, and we can see this in the public domain. It's often used in Linux packages to determine what's going on. So we will use Git to get OpenStack. And that's basically involves a few commands. So if we look at the command line here, and what we'll do is we'll navigate to the root directory of our Ubuntu installation. Now, the root directory is the lowest level of hierarchy a directory can change to. If we think of the file systems like a tree structure, well, basically the root directory is it right at the top of that tree. So if we list this as a command, we type in ls dash l, we can see all the files or the folders that comprise of the root directory. A lot of these are system files crucial to the operation of Ubuntu and

the Linux operating system.

So now that we have actually navigated to the root directory We'll actually use Get to create a new directory within this route, to create our OpenStack environment. So what we do is we use pseudo then we use Get, then we use a feature within get called clone.

Now this clones the repository from open stacks website. So if we type in HTTPS get open stack.org, openstack-dev, DevStack. I just copy what exactly what I'm typing in here, and this will work. And then what we do is then the command after what URL is the name of the folder that we're gonna create within the root directory. So we're gonna call it DevStack. If I press Enter here now, what this will do is get will then clone the actual repository from open stack servers and it will make the actual file structure for OpenStack. So that cloning process is now complete. What we do then is we change the directory into our newly created DevStack directory which gives us created. And if we do a CD DevStack that means change directory, we can see we're doing with our salad. Within this directory there is now a set of files that have been created. These are the files that get has just clone from the repository. So now what we need to do is actually configure OpenStack. What OpenStack includes within the repository is a example configuration file that we can use as a template to our own configuration.

But what we need to do is copy that configuration from the samples directory. We can see the samples they're actually quite clearly in this recording. So what we need to do is issue the command pseudo CP, which means copy in Linux, samples, because we wanna copy from the sample directory. Now the configuration file for OpenStack is called local.com. And what we want to do is we wanna copy that to our current directory, which is DevStack. So to do this we just type in the name of the file, we want to call it which will be the same as the sample configuration file. So if we issue this, that now then creates the copy of the sample file within dev start the show by doing LSL. We can see clearly that local.com is now present within the DevStack that directory and I know we can configure this to our own. So now what we need to do is run a text editor to edit the configuration file.

What we usually use for this is a text editor called Nano. And this is ordinarily included in most Linux distributions, and it's very straightforward to use. It's one of the easiest text editor you can use within the command line. So if we issue the command Sudo, Nano local.com, and then what we're presented with is the actual template configuration file that we copied over from the samples directory. If we never get down with the arrow keys, you can see a flashing or cursor. That's where our current position is of our, when we want to edit our text. So the only thing we wanna be changing in this directory, in this configuration file is the where the flashing cursor is which has admin password. We wanna change this to a password of our own choosing. Now I'm not gonna pick of strong password because this is just for demonstration purposes. So I'm just gonna say password with a capital P, one, two, three, four, five. But I say you can change to whatever you desire.

So once that is completed, once you've changed up that, tou then press Ctrl+X. Now Ctrl+X exits Nano. But what it also does is it detects changes that have occurred within

the actual file you've opened.

And it will ask you where you want to save the file. If you type in yes or Y to this and then press Enter, then this will make the changes. If you press no or N and press Enter, it will not save the changes. In this case, we do want to save the changes last is the name of the file to write. This is obviously gonna be the same name as what the file is already created. This will overwrite the existing file with a new updated version of the file. So we press Enter on that was his file name to right. And then it will quit the app and then our changes have been saved. Now we can go ahead with the deployment of OpenStack. So the first thing we need to do is actually create a separate user for OpenStack. There is a script included within the DevStack repository which because for us.

So we're just gonna run this script very quickly. So we do this by running sudo, In the DevStack directory. Then in tools, and then there's a create-stack-user. So if we hit Enter on that, and then what you'll find is a new user has been created for you automatically. It sets certain parameters, which save us having to do it manually. It just makes the process a lot more straightforward. Now this newly created username, the script's just done for us.

We need to then change the ownership of the DevStack directory to the new stock user that we've just created. If we do a list LSL within the directory so far, the current owner is actually route.

We don't want this, we want the owner to actually be the stack user because this is the user that will be used to administer DevStack. So we do this by using a tool called CHO, change ownership. So if type sudo CHO and we wanna do this recursively know what that means is it all go through each directory within the DevStack directory tree and it will change those parameters according to the username we want on the group that we want to set. So we use capital R and then we basically specify their username, and the user group. Now the username and user group are the same, so it's just a stack, colon, stack. And we specify the directory we want to play those ownership rules.

So we take DevStack and what you'll find is if we do another LSL then, it's changed from root, root to stack, stack. So that means that the ownership of these files has been changed. And then we can now proceed with the actual installation of the OpenStack environment.

So now that we've changed the permissions for the DevStack directory to the stack user and group, we now need to log into that stack user in order to initiate the setup process for OpenStack. So to do this, we'll use a command called SU. That basically means superuser, it allows us to change to whatever user we desire as long as we have the administrative credentials. So if we type in sudo or just click on the window sudo, and as you enter the name of the lady user that we want to change too. So we press Enter now and we can see that our username has changed from what we were

logged into to stack, which is the new user that was created just before. From here, and now we can actually initiate the script which will install OpenStack. So to do this, it's pretty straightforward.

We run the DevStack. I now move on stack.sh. Now, if you're wondering what an sh file is, it's basically a script. It's a script that runs within the command line of Linux terminals and has a set of instructions which tells the command line what to do. Scripts can be very simple, they can be very complicated. It depends on what the actual script's trying to achieve. Know that this script will be very advanced. There's a lot of processes involved with it. And just to be aware, this process could take a while to actually finish.

So we run this now, and you'll see a lot of text showing in your console window. This is normal, this is to be expected. And what's actually happening now is the script is downloading the required packages for OpenStack to operate. There's a lot of different packages. There's also a lot of compiling that needs to be done. There's a lot of source code that OpenStack needs to have compiled in order for it to operate. An OpenStacl is primarily a Python-based distribution. So a lot of Python code will be sort of integrated into this. We can see here, clearly this Python development package is currently installing. So this installation could take at least half an hour, possibly even longer, depending on the speed of your machine. So I'm gonna pause this video now, and once we've finished here, we'll move on to the actual administration of the OpenStack environment. Once the OpenStack installation is finished, we all see a set of information that will be useful for us when we're accessing the dashboard within OpenStack. We can see here that it generates the IP address of our machine. And it will tell us the URL for accessing the user interface that you're gonna use to configure different aspects in OpenStack. So the piece of information for most prevalence was most important to us is the horizon dashboard. We can see here the URL is actually provided for. So if we actually open a web browser, we can now access this Website from the URL actually given to us. So we just type in 192.168.10.199. Of course, this will differ on your machine. You have a different IP address than what you see here.

Let's load up. And you can see now we're presented with a OpenStack login portal. So now what we have to do is, is login using the admin account that we are provided with OpenStack. So all we have to do is look at the information here on the console, and the default user admin and demo. So we wanna login with admin of course. If we type in admin and the password will be the password we set up in the local.com file, we setup a while back. So that in my case is password, one, two, three, four, five. We sign in there now, and then we should be presented with the dashboard of OpenStack. So here we can see the initial view of the dashboard as a summary of the actual environment. So at the moment, we have no instances running the course, this is the first time we've run OpenStack. We've got no virtual CPUs in use and no RAM in use of course. But this is a useful dashboard to summarize what is going on in OpenStack. Where it's allocated and what's in use is a good way of determining

what resources he left on the machine and whether or not they need to be

distributed amongst other computers within the cloud. The final part of the tutorial, we're just going to set up two instances. And we're gonna try and pick between these instances using a shared private network within the actual OpenStack environment. So what we do is just we click on instances here and we're presented with a blank screen. So we just click on launch instance. And we're presented with instance name and the description. We only need to enter a name, we'll just call the first machine Test 1, I'll click on Next. Leave all these options as they are and we are going to use a nimble image within the OpenStack environment. It's a Linux-based distribution called Sarah.

Now this is usually for testing purposes, we don't use this in production environments. We would actually deploy our own image. Something like Ubuntu or Debian, or one of the Linux distributions, or even Windows. We could have Windows Server, for example. But we'll use Cr OS in this example. So to deploy Cr OS, we have to have this little arrow click here. This will then move from available machines to allocated. Keep everything off default, create new volume. Yes, and if we go on to the Next. Now the next screen determines what resources we have available for this instance. Now, you may or may not have seen this. If you're familiar with Amazon AWS, they have similar sorts of setups, where you can choose different tiers of computing power. And this will depend on what you require. Do you need a lot of memory? Do you need a lot of processor cores? And of course, the more of these specifications you add to your machine. They're obviously they're higher the cost will be to run.

So we will just choose the least specification of template. We don't need anything significant cuz we are running a very low powered operating system. So we click Next. Now, we put shared network because we want to communicate between a local subnet within our open stock network. We click on Next over that. We won't touch anything else in this actual tutorial. This will all leave for you to actually experiment with. So if we click here Next, we don't need to touch these, so we'll just go and launch Instance now. And then you'll have to wait a while and it should show as it has here now. And what is actually happening here is the virtual machine is being built as I'm speaking. So we know it's been built because we look at the task and it's in the process of actually doing something. When this has actually completed as it as just a note, the power stays at running that's what we want to see. We're also be given an IP address. Now, you may notice that the IP address provided here is not the same subnet, as was before provided from your dashboard. Of course, this is a virtual network. This is a network running within the OpenStack environment. It's a virtualized network. So remember we're talking about software defined networks and network function virtualization. Well, this is in effect, what we're seeing here, this is open start using open V switches to actually create a virtual network.

So what I've done is I've created another instance, identical process to before, but I've called it Test2. You can see here now they're both running in the console. And what we are going to do is, we are going to pick from Test1 to Test2. And if all goes well, we should be able to get a response that shows our virtual network is actually working and we can communicate. So we go to the Test1 console. Now, the console is a similar concept to what we have seen in VirtualBox. So you'll have a command line interface, you'll be able to interact with it. This of course, is in OpenStack, so don't get it mixed up with what you've been seeing in your VirtualBox window because this is not the same.

This is actually the instance running within OpenStack. So if we click on the virtual machine, go to the Console. And as you can see here, we are presented with a command line interface. If we just press Enter, and you can see we can actually interact with it with the keyboard. So as we can see here the default login details are provided for us. So if we just type them in, just the username is Xerox and the password gocubsgo We are then now logged into the instance we made. So to confirm we have actually got an IP address that corresponds with what we've seen in the console, we'll just type an ifconfig. And as you can see there in the EF north stat that we actually do have IP address, which is good, that's what we wanna say. And this is the instance basically participating within the network stack of what we've been basically assigned to with the private network that we set up earlier.

So if we just go back, get the IP address of the other machine, so we go to Instances. So the IP address is very similar, of course, as you expect in the same subnet, but the last IP address is 247. So if we go back to Test1. Just wait for that to load. If we just actually type in ping Now, as you can see here, we're successful. So what's actually happening is the Test1 instance is sending ICMP packet to the test to instance. So we have network communication now between the two instances. So this concludes the tutorial on how to deploy OpenStack within Ubuntu. To record quite a few things today we've deployed a Ubuntu virtual machine. We've actually incorporated OpenStack within it and we've created some instances were able to communicate between those instances.

So we encourage you to narrow take a walk in further depth on what you can actually do it over the stock, experiment with different things. Look online and Look at tutorials and see what you can find them what you can actually do with it. What we've done today is only scratched the surface on what this environment is actually capable of. So take a look and see what you can discover from your findings and try to implement your findings into the actual environment and see what you can actually achieve. So thank you for watching this tutorial, and we'll look forward to seeing you in the next week's tutorial.