How to Set Up A Personal Testing Environment

A Guide by Mr. Gibson

This guide is intended to be used to set up a home lab testing environment. I had difficulties in getting this setup and am writing this guide in the hopes that it can help save someone the headache in the future. My personal philosophy when it comes to setup is to make sure all of my home lab equipment is segregated from my network, and to allow expandability in the future when/if more resources become available to me while keeping it relatively simple.
In this environment I am using proxmox as my bare-metal hypervisor. Why? Proxmox is a free open source hypervisor that has a lot of support online. It was also an opportunity for me to learn a new platform. You don't have to use Proxmox to accomplish these tasks, however the steps listed below are Proxmox specific. Listed below is my guide on how to get a simple setup going for you to try at home.

Hardware considerations

First we will talk about the hardware requirements and whether or not you will have the resources to build a homelab. Keep in mind, this setup is very scalable so even if you meet my minimum requirements there is more that you can do. If you're worried you don't have the resources to make your own home lab, skip to the bottom where I list the required minimum specifications, you may be surprised!

In my environment there are 6 Operating systems and accompanying hardware considerations that you have to be aware of.

1. Proxmox host itself
2. A Kali machine
3. A firewall like PFsense (optional)
4. A Windows machine
5. A metasploitable machine (optional)
6. Security Onion v2 (optional)

Listed below we will go in-depth with each OS and see the requirements and I will summarize below the hardware requirements and different deployment scenarios based on what you're looking to accomplish and the resources you may have available to you. Something to note is, just because you allocate an amount of resources doesn't mean they will be used in their entirety. What this means is you can effectively over-slot how much resources you allocate because certain machines won't be using all the resources you allocate, however if you over-allocate and don't have the resources when the machine needs them you will run into issues.

1. Proxmox
    a. Proxmox is the bare-metal hypervisor needed to run the OS's and manage them on one platform. I recommend having a dedicated server/spare computer with these components installed.
    b. CPU: Any newer generation AMD/Intel CPU should be fine as long as it has virtualization support. Typically Proxmox itself isn't CPU intensive and will only require 1 CPU core.
    c. Ram: Again, proxmox is very lightweight and you won't have to dedicate hardly any Ram at all, recommended 1GB.
    d. Storage: For proxmox itself you will need to have around 100GB of storage for the OS and various other requirements.
2. Kali
    a. Kali is my preferred choice for a red team emulation OS. You can use others like Parrot OS but to me Kali is the most lightweight and feature-packed OS.
    b. CPU: 4 CPU cores. You may want to increase this number if you plan on doing any type of password cracking or scripting.
    c. Ram: 8 GB. I went overkill with the amount of Ram I gave and you can easily get by with 4 GB.

    d. Storage: I gave 200GB but you can easily get away with 100GB. If you're really worried about space 50GB is the absolute lowest I would recommend.
3. PFsense
    a. PFsense is a stateful firewall that has the ability to do packet inspection and provide routing functionality. Due to its open-source design and community support it's the firewall of choice for my build. However, you do not need to incorporate a firewall into your build, however this will mean you will have a flat network with everything in 1 subnet so be wary.
    b. CPU: 4 Cpu cores. If you plan on expanding in the future, routing is typically CPU intensive, but you can watch the resource monitor in the future to see if you need more.
    c. Ram: I allocated 12 GB of Ram to PFsense. However this is definitely overkill and you can get by with 4GB. The reason you would want to allocate more Ram is if you want to enable certain IDS/IPS features built into the firewall like Suricata. However you will typically be safe with 4GB.
    d. Storage: You don't need a lot of storage allocated to PFSense and can get by with around 50-100 GB of storage allocated.
4. Win10
    a. I am using a slightly out of date version of Windows 10, version 1607. I may replace this with a newer version of Windows in the future but for red-teaming purposes I am trying this version out first.You can use any version of Windows, but I think it's important you have a windows machine in your environment.
    b. CPU: 4 CPU cores, but could be as little as 2.
    c. Ram: 4 GB, wouldn't recommend anything lower and ht
    d. Storage: 60 GB.
5. Metasploitable
    a. Metasploitable is a purposely built insecure server where a majority of exploits will be successful. It purposely has misconfigurations and security vulnerabilities to test and try out! It is also very lightweight and is a good tool to practice basic penetration testing against.
    b. CPU: 1 CPU Core
    c. Ram: 512Mb (1 GB to be safe)
    d. Storage: 8GB
6. SecOnionv2
    a. My SIEM of choice. I'm running version 2.3.61 and will be configuring it in standalone mode. Security Onion 2 is a major resource hog, and if you have concerns about whether or not you will have the resources you may want to look into cutting SecurityOnion2 first. My philosophy when it comes to this environment is I won't have constant network traffic, and will only have traffic when I initiate it. With that in mind, minimum specifications are viable. If you incorporate more machines or have an emulation engine inside your environment you will need to increase these specifications.
    b. CPU: 8 CPU cores is the minimum, if you have an active environment you may want to increase the number of cores.

    c. Ram: The minimum to run SecOnion2 is 12GB of ram. If you have any extra available ram after you complete this guide, I would dedicate extra ram here if you notice it being tapped out.
    d. Storage: 200GB is the minimum storage requirements and what I recommend.
7. Ubuntu/Kubernetes Server
    a. My Kubernetes server is primarily for testing and gaining an understanding of docker containers. Right now I'm running a Wireguard VPN server exposed to public internet that allows me to connect to my home server securely over VPN. Wireguard is running in a docker container.
    b. CPU: 4 Cores is the minimum I would give but you can give more if you have more plans with docker containers. Again, the great thing about docker containers is the stripping of unnecessary services.
    c. RAM: 4GB is again the minimum I would give. But like stated with CPU requirements, you can attribute more if needed.
    d. Storage: I designated 40GB of storage as I thought that would be more than required.

Summary

If you are wanting to build a bare-bones homelab setup using proxmox you will need (Only Kali, Win10 and PFsense)
CPU: 8 Core Processor
RAM: 16 GB
Storage: 200 GB of dedicated storage space

If you are wanting to replicate my environment with aforementioned OS's you will need (All OS's)
CPU: 12 Core Processor
RAM: 32 GB
Storage: 2TB of dedicated storage space

My current specifications when typing up this document (All OS's plus extra services I run)
CPU: 24 Core Processor
RAM: 64 GB
Storage:14 TB of dedicated storage space

Setting Up Proxmox

Setting up and deploying Proxmox is a simple setup. I will walk you through the process of installing Proxmox onto your local dedicated server. Additionally once it's setup we need to do some additional configurations to make it suitable for a threat testing environment.

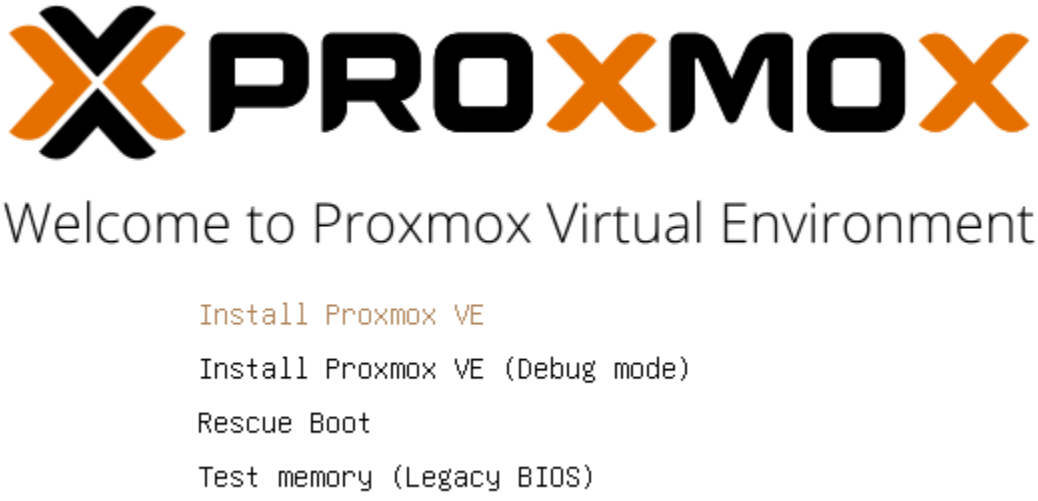
Step 1. Downloading Proxmox OS and creating bootable media

1. The first step will be to download the latest version of Proxmox from here:
   https://www.proxmox.com/en/downloads
2. After download you will need a thumb drive with nothing on it that you will make bootable and copy the contents of the file onto. One piece of software I recommend is Etcher that you can download here:
   https://www.balena.io/etcher/

Step 2. Running through the install of Proxmox on your local server

NOTE Before we begin it's important to understand that you will be wiping the contents of any OS you have during this process. That's why it's recommended you have a separate physical server when doing this. Additionally in this server should be the recommended list of components listed earlier in hardware requirements. Once you have confirmed these factors you can begin installing Proxmox.

1. Connect your server to a keyboard and monitor, connect your server to your network via a wired ethernet connection, plug in the USB device that you created earlier and power on your server.
2. You should then boot into the Proxmox VE install screen. Select "Install Proxmox VE".
   a. You may have to hit your BIOS motherboard and select your USB device as top boot priority if you aren't immediately presented with this screen.



3. You will be greeted with a EULA. Just select I agree to continue installation.
4. Next you will install Proxmox onto your server. You will be asked where you want to install it. I had multiple drives in my environment so make sure you pick the appropriate drive in the bottom drop down menu.
5. You will be prompted to select your location and time zone. Pick accordingly.

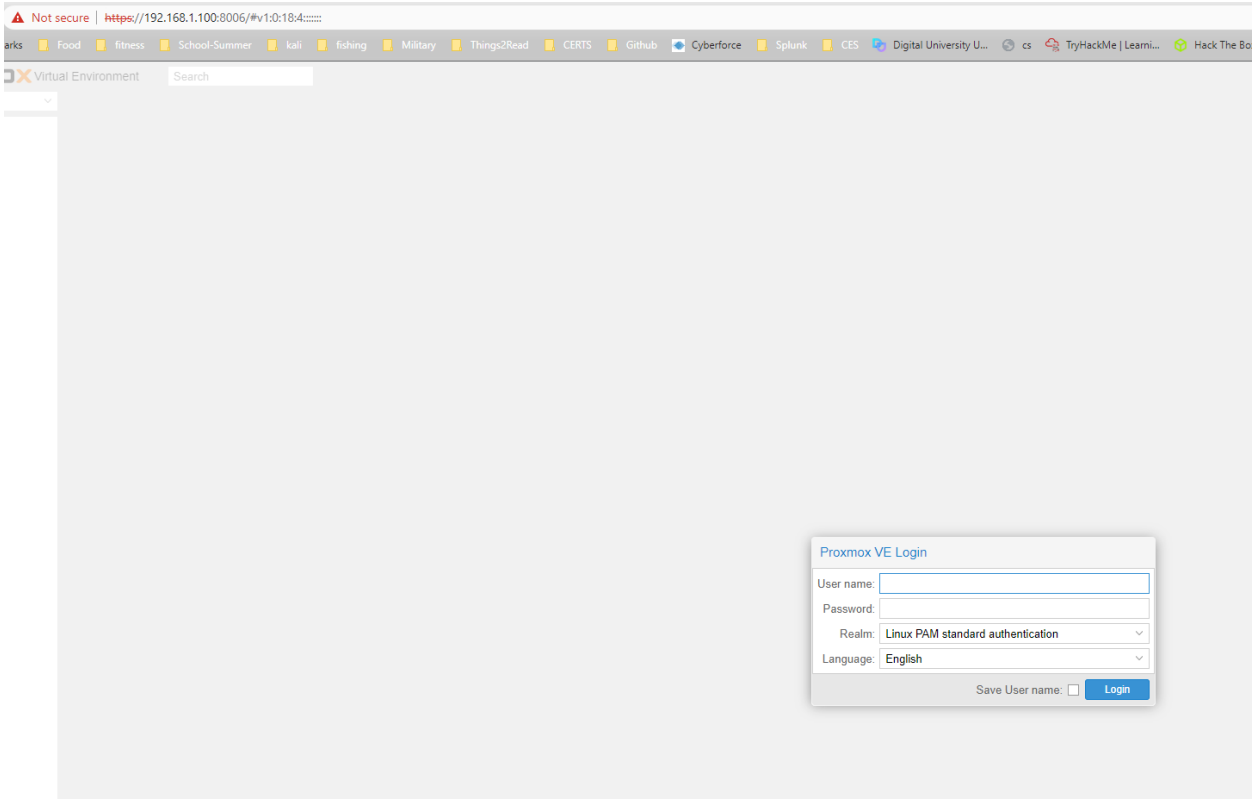
6. You will then be prompted to select and administrative password and email address. I would advise you use a real email account as you can tailor Proxmox to send you important emails about your server if need be.
7. Next you will be prompted to input your Network settings. For your hostname you can put essentially anything, however I put proxmox.mitchell.test, something equivalent will be fine. For your IP address, netmask, gateway and DNS use relevant information from your home environment. My network is 192.168.1.0/24. I used those parameters and gave my promox the static IP address of 192.168.1.100 with a netmask of 255.255.255.0 and my gateway/DNS of 192.168.1.254 based off my network settings. If you're unsure, go to any local host on your network, open up command prompt and type ipconfig /all and find the connection from your ISP and you will be able to configure your network settings.

```
Ethernet adapter Ethernet 4:

   Connection-specific DNS Suffix  . : attlocal.net
   Description . . . . . . . . . . . : Realtek Gaming GbE Family Controller #2
   Physical Address. . . . . . . . . : 24-4B-FE-01-E5-A0
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.1.206(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Wednesday, December 22, 2021 9:25:04 PM
   Lease Expires . . . . . . . . . . : Tuesday, December 28, 2021 10:13:52 AM
   Default Gateway . . . . . . . . . : 192.168.1.254
   DHCP Server . . . . . . . . . . . : 192.168.1.254
   DNS Servers . . . . . . . . . . . : 192.168.1.254
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

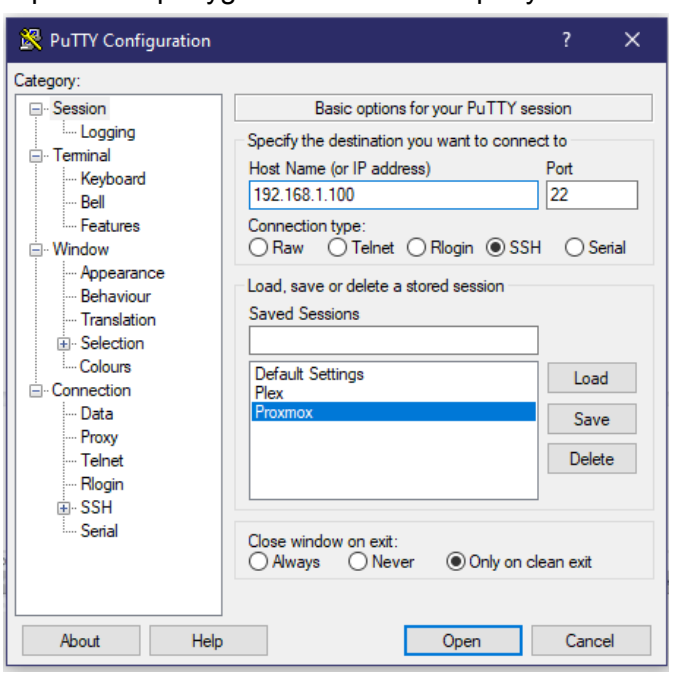
8. Lastly you will be presented with a summary of your install configurations. Verify that everything is correct and begin install.

Step 3. Validating Install and Post Install best practices

1. After the installation completes and it restarts, you should now be able to access your server via a web browser over port :8006 and/or putty.
    a. If you aren't able to reach your server via the IP address you assigned to it, you may want to check that your installation was successful and you have a valid network connection.

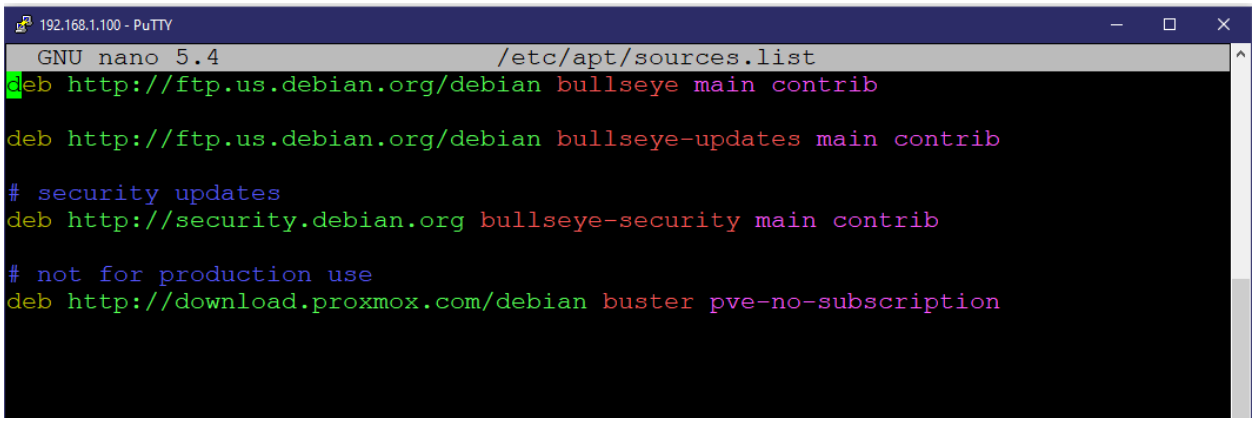


2. From here, you will want to download a remote management tool like putty and login to your server for some extra post installation configurations.
    a. Putty can be installed here https://www.puttygen.com/download-putty#Download_PuTTY_073_for_Windows



3. Upon logging in, use the credentials you created, you can now access your server remotely!

    a. A helpful tip is to save your session in Putty like I did at the top, that way you don't have to keep inputting in the same content.
4. First we will add a website to pull updates from.
    a. Add a line named not for production use in /etc/apt/sources.list
       It should look similar to below



5. Now we will edit the enterprise list to not pull the repo considering we won't be paying for an enterprise license.
    a. Comment out a line in /etc/apt/sources.list.d/pve-enterprise.list
       It should look similar to below

6. You will next want to update your distribution list
    a. Type: apt dist-upgrade
        i. Type y to confirm any updates
7. You will then want to reboot your server
    a. Type: reboot
        i. Note you will lose connectivity and will need to reconnect after it has finished rebooting
8. Next you will want to go into your WebGui to create some network interfaces.
    a. Once you have logged in click on your node and go to the network tab on the left hand side.
    b. You should see 2 interfaces but you will want to create a 3rd and 4th interface.
        i. Click the create tab at the top and then select OVS bridge
            1. Give the bridge a name
                a. I named mine vmbr15

2. Give it network specifications which will be used in your environment (give it different network specifications than your home net)
    a. I gave mine a 192.168.3.100/24 network
3. Make sure you have auto-start selected

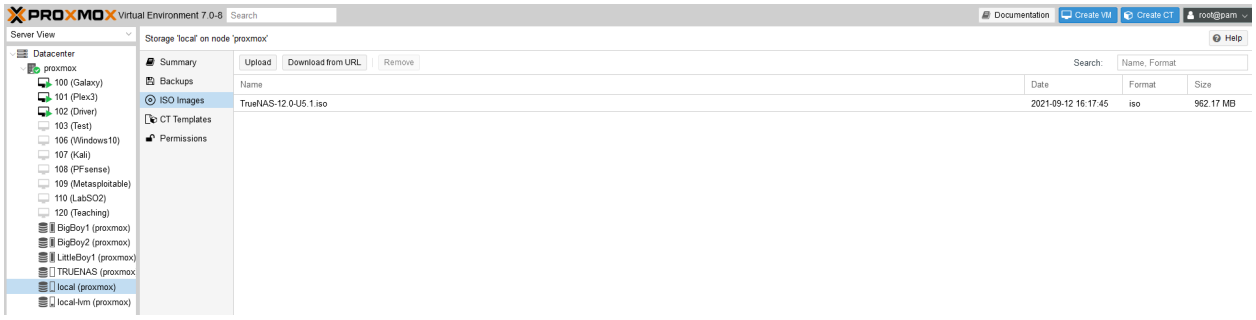
ii. Click the create tab at the top and then select Linux bridge
1. Give the bridge a name
    a. I named mine vmbr1
2. Give it network specifications which will be used in your environment (give it different specifications than either other interface)
    a. I gave mine a 192.168.2.100/24 network
3. Make sure you have auto-start selected

iii. Click Apply Configuration at the top of the screen to apply your changes and create your new network.

c. What you just did was essentially create an out-of-bounds network that you can host new VM's in without having to worry about outside internet connections.

Step 4. Importing Virtual Machines into the Environment

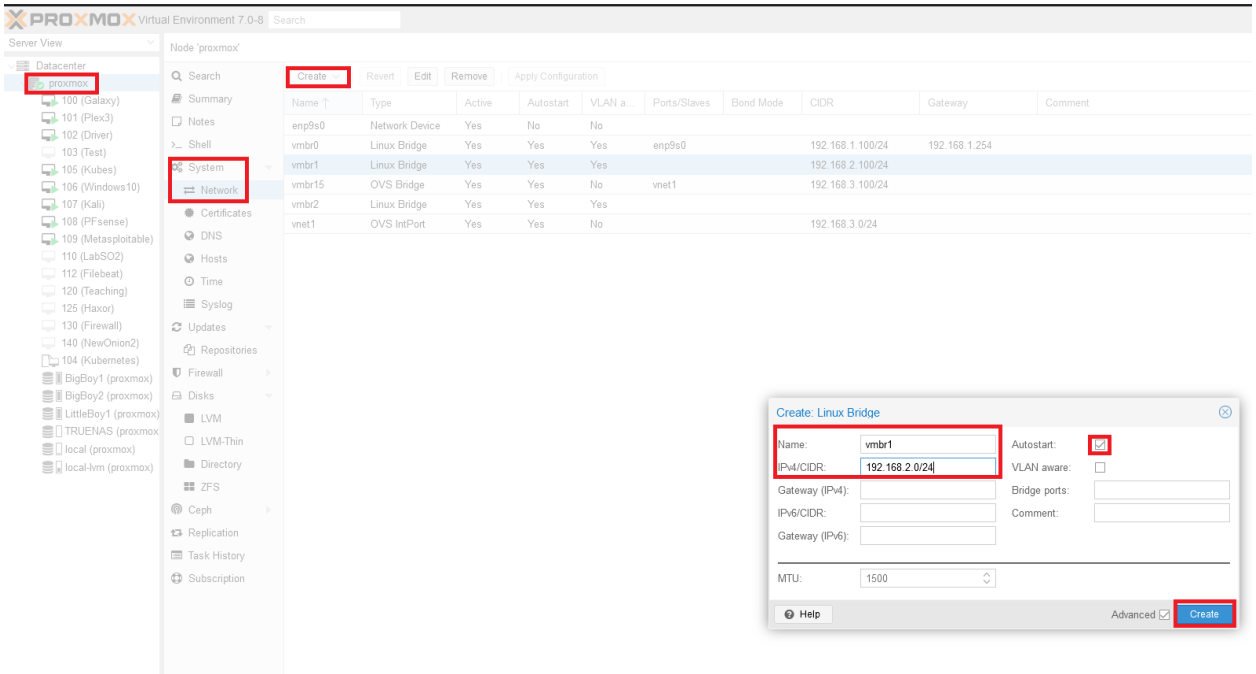
1. Because every server environment is different I will walk through a simple process for uploading VM images onto your local storage (where you installed proxmox on your physical drive)
2. Click on your local storage on the left hand column in the WebGUI
3. Click ISO Images in the middle column and it should bring you to a blank screen
4. From here you can upload any .iso from your local machine and it will store it on your proxmox server, allowing you to create a virtual machine using that iso at any time
    a. I uploaded a TrueNAS iso to later create a NAS in my network for personal use



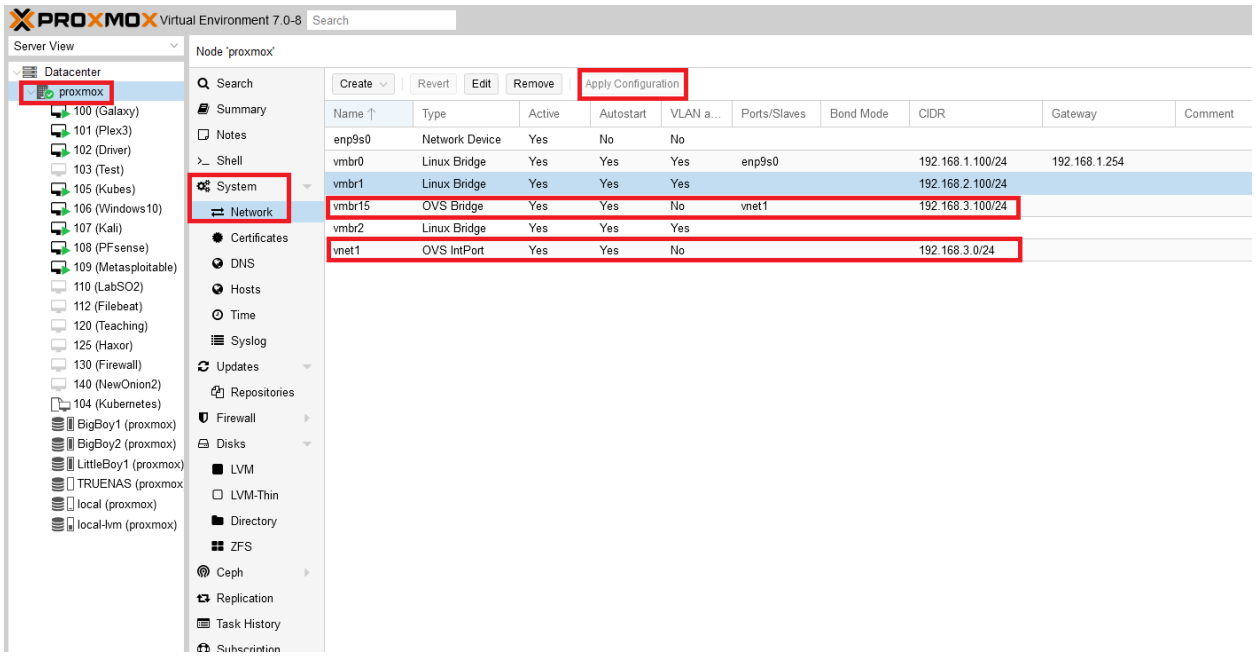
Step 5. Setting up our networking

To really get the full use out of our training environment we will need to properly set up our networking. Thankfully this is a relatively easy process. In my environment I will have 2 internal networks that won't have internet access; vmbr1 and vmbr15. Vmbr1 will be "red space" and vmbr15 will represent our "customer network". It's important that these are separated that way we can manage them through a virtual firewall with both NICs attached. If you're curious, vmbr0 is the bridge that is used to talk with the rest of your network. Only put devices you're comfortable having internet access on vmbr0.

1. In order to accomplish this setup we will have to create 2 virtual linux bridges. The fits bridge we will create is vmbr1. Go to your main node then under system select Network. Then you will click the Create button and select Linux bridge. From here you can name it whatever suits you, make sure it's set to autostart and I like to give add the IPv4 CIDR of my network, in this case my "red space" will be coming from the 192.168.2.100/24 network space.



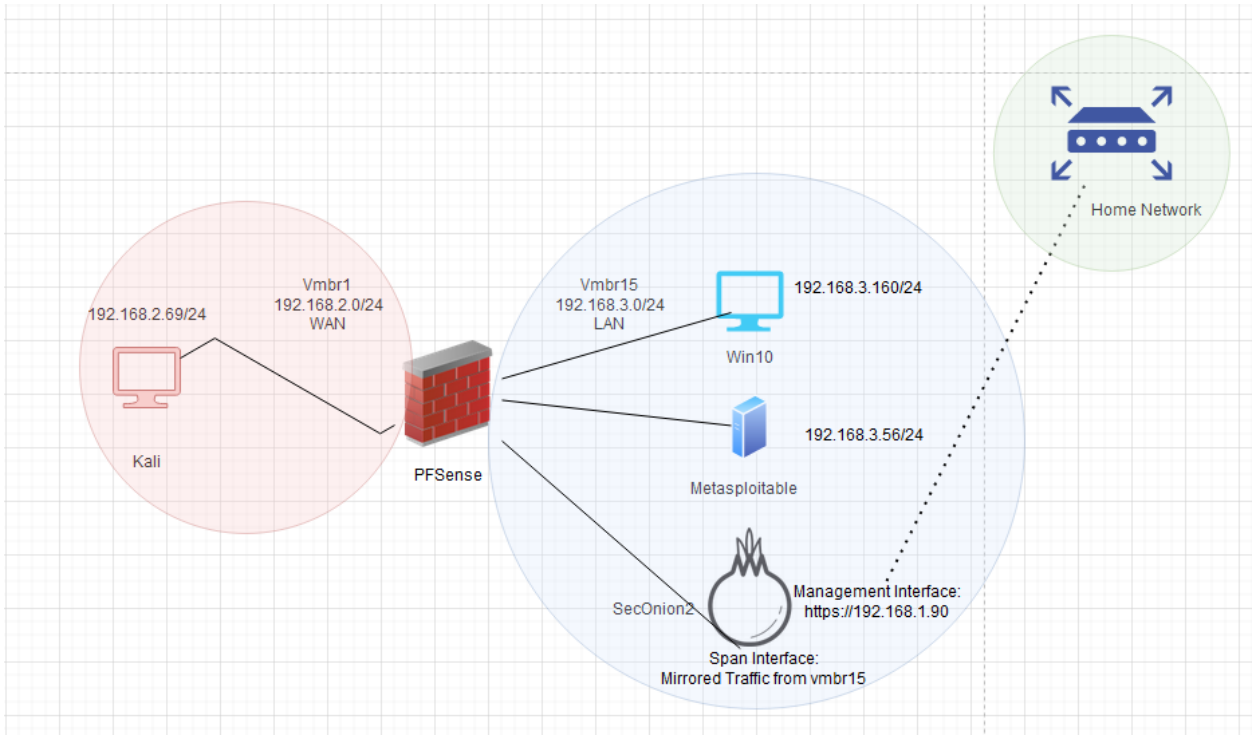
2. Next we will create our "customer network". You will repeat the same steps by going to your main node, then under system selecting Network. From here when you create a network you will instead select OVS Bridge. This will allow us to mirror that traffic in the future for our SecurityOnion2 sensor. I named mine vmbr15, gave it the ip range of 192.168.3.100/24 and made sure I created a bridge port of vnet1. I then edited vnet1 to have the same subnet as my bridge. Once you have made all the necessary changes make sure to click the Apply Configuration button to apply your changes.



3. One last note if you're not planning on using PFsense is to still make at least one of the virtual bridges. This way you won't be exposing your VM's to the internet and can throw exploits without worry. Throughout this guide I don't demonstrate this method but the short answer is to put all your VM's in the same virtual bridge you create.

Step 6. Network Map

Show below is a picture of the network map that I have internally in my network. You can use this network map as the basis for how you may want to setup yours.



Step 7. Extras

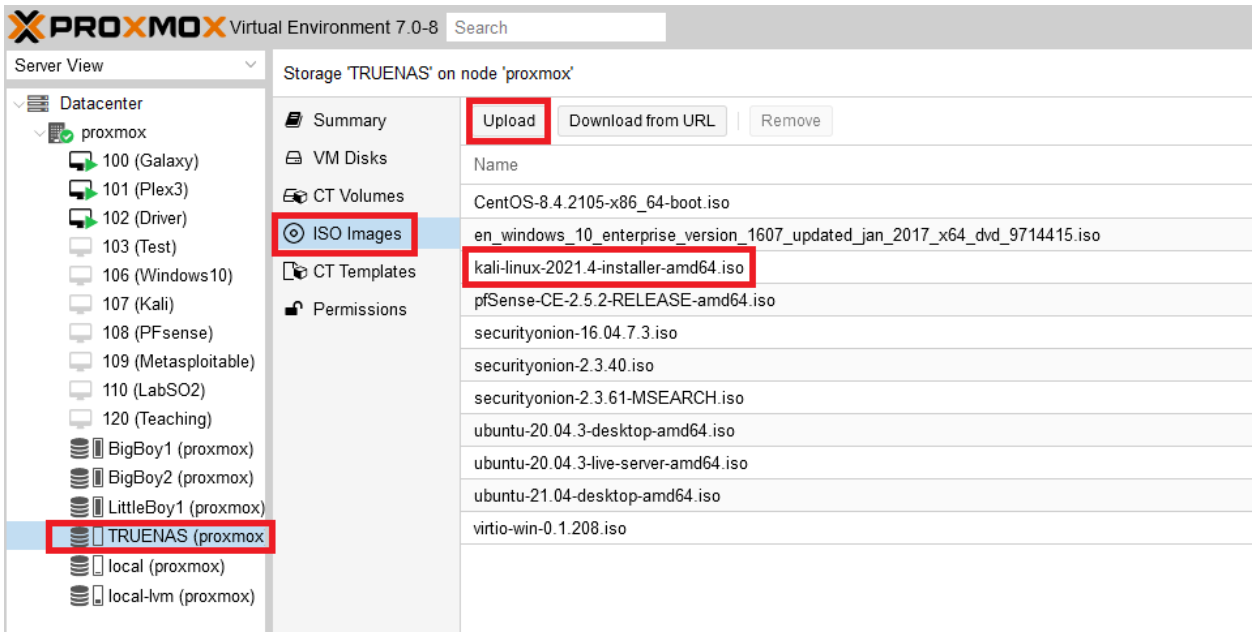
There are a lot of extra configurations setups that you can do. For example in my personal environment I combined my two 14 Terabyte drives and passed them through to TrueNAS and then hosted those drives via a NFS share back to my Proxmox. Now I'm able to store iso files on my NAS and access them from anywhere in my local network. You can also do other stuff like GPU passthrough, VLAN management, you can incorporate an actual firewall into your home network and more.

Setting Up Operating Systems

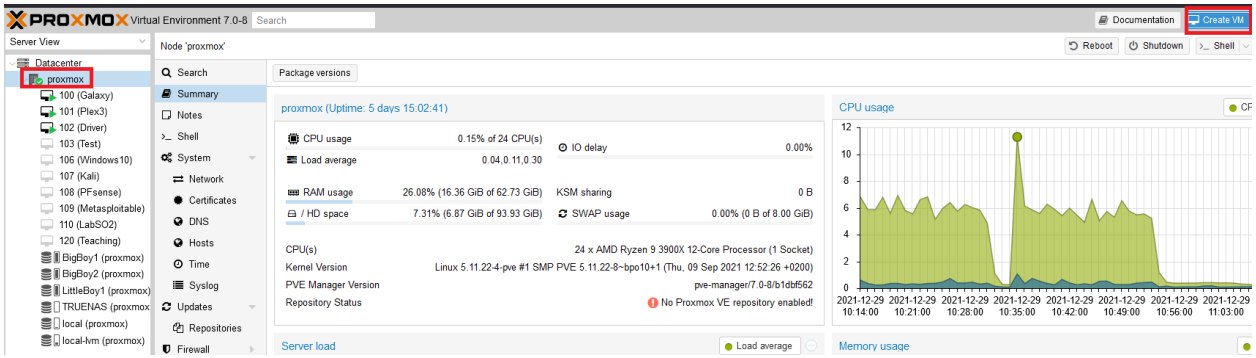
1. Kali

Kali is the easiest of all the Virtual machines to setup and deploy. I will walk you through the process for deploying this VM first, which will be the basis for all the other VM's you will be deploying later.

1. The first step will be to download the latest version of Kali. There are a couple different versions that you can download, but just make sure you select the bare-metal version. The link will be listed here. https://www.kali.org/get-kali/
2. After download, make sure you upload the iso into your Proxmox iso repository. Listed below is how mine is currently configured.



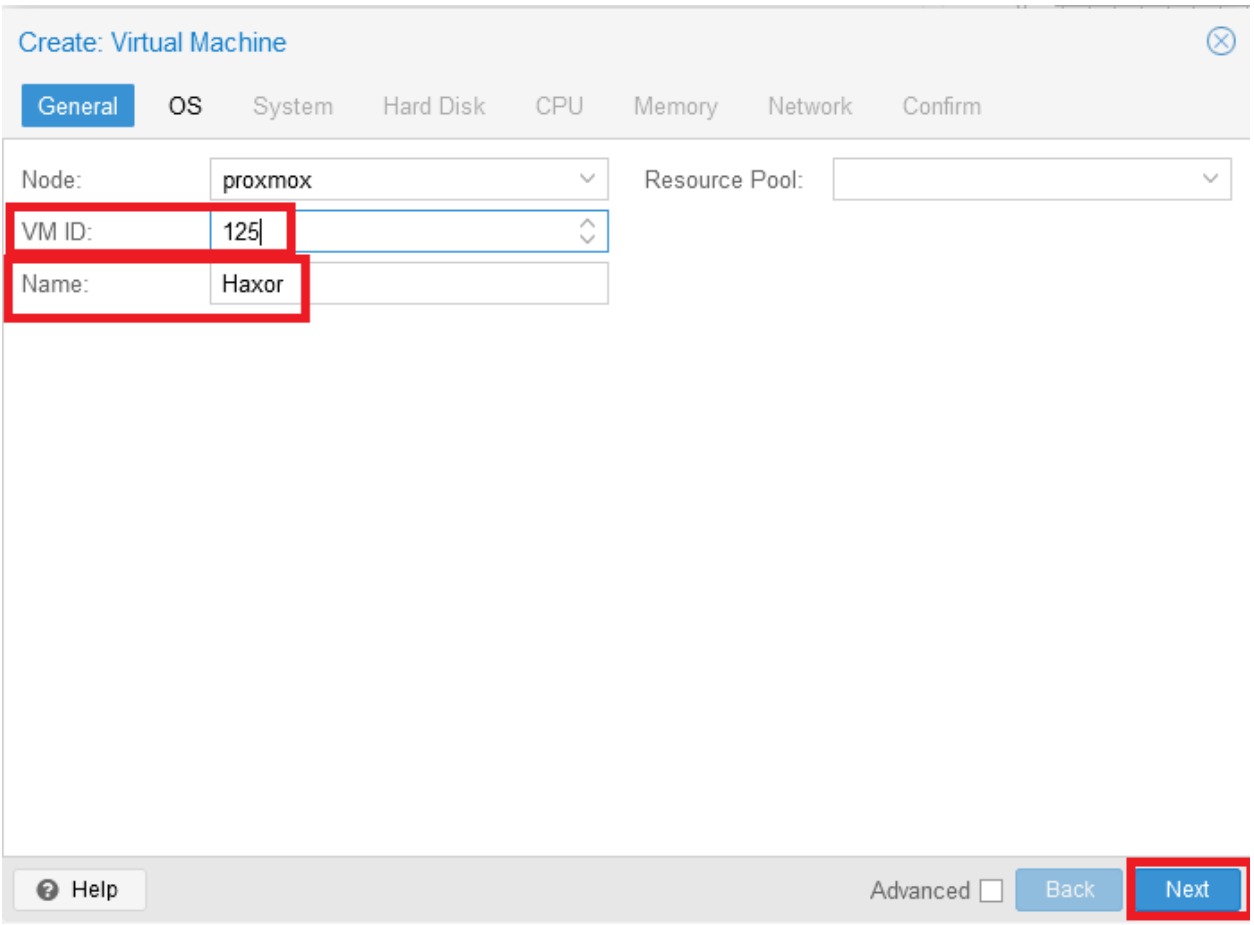
3. You will then want to begin the creation of the VM. You will first need to go to your node and click create VM in the upper right-hand corner as shown below.



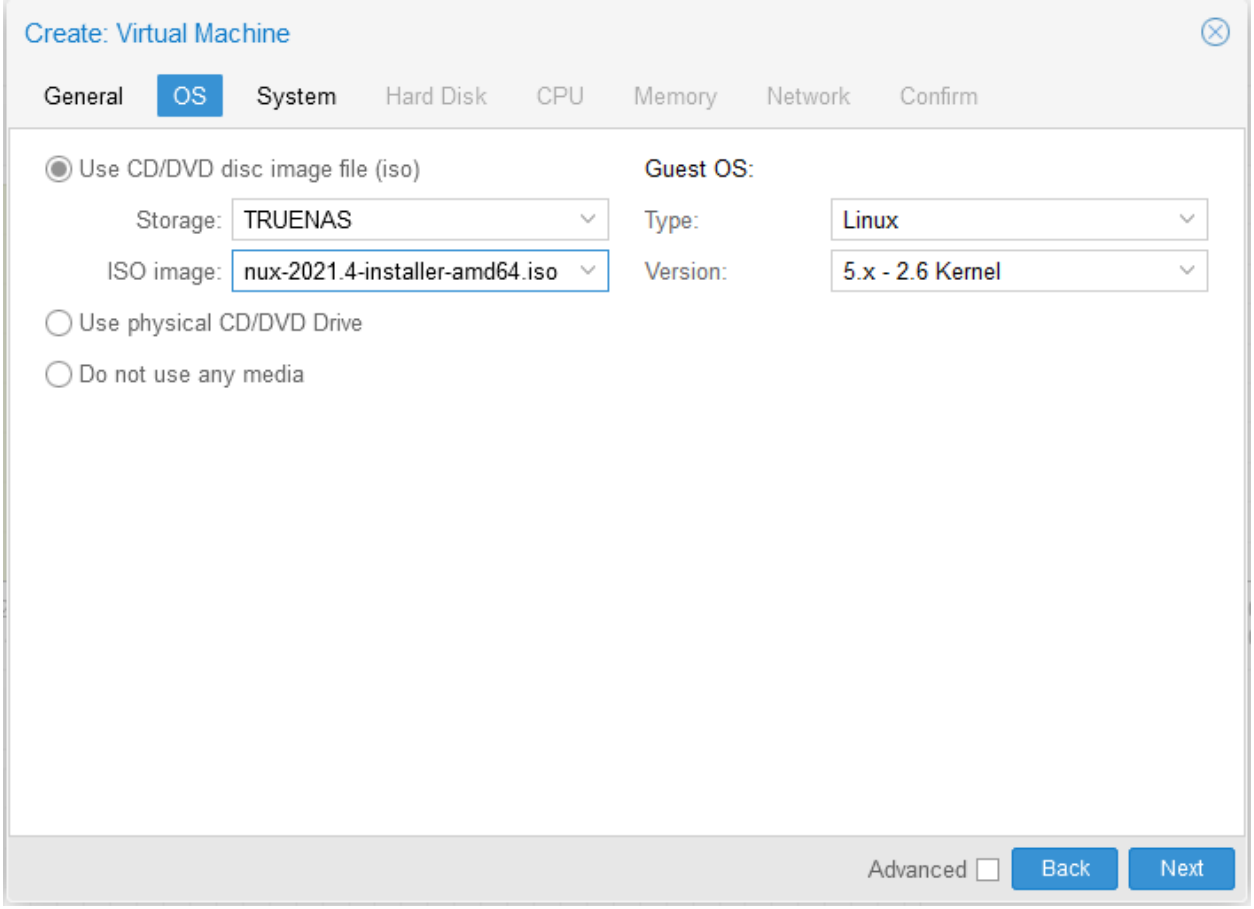
4. When you click the button a window will appear in the center of the screen. This window is where you will make the initial configurations for your new VM. For this first screen, the only important things to note are the VM ID and the Name. The VM ID is what Proxmox

uses to differentiate VM's in your environment, you can leave it at the default which will incrementally increase every time you create a new VM. The next is whatever name you assign it. Once you select those options select Next in the bottom left-hand corner

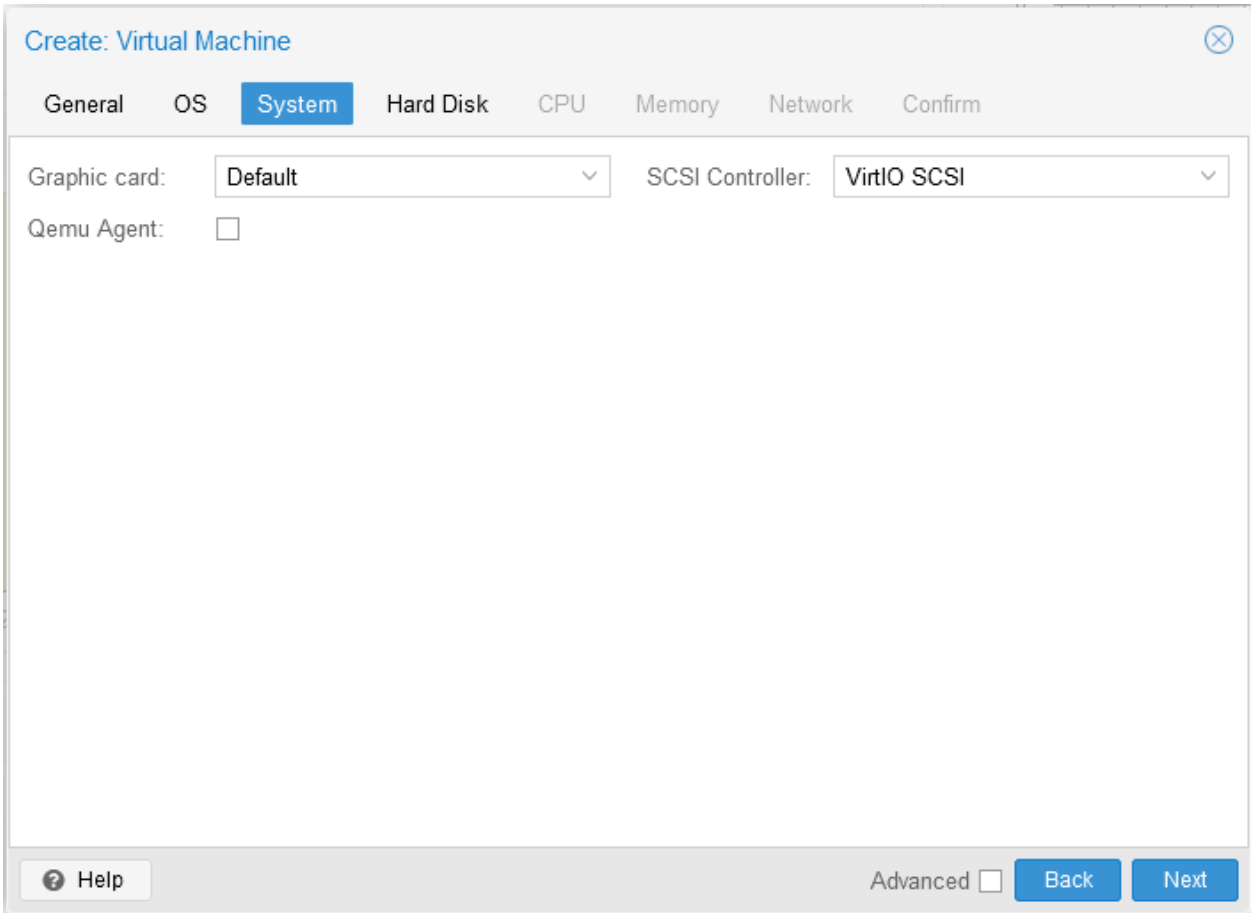
a. Here I gave it a VM ID of 125 and a name of Haxor



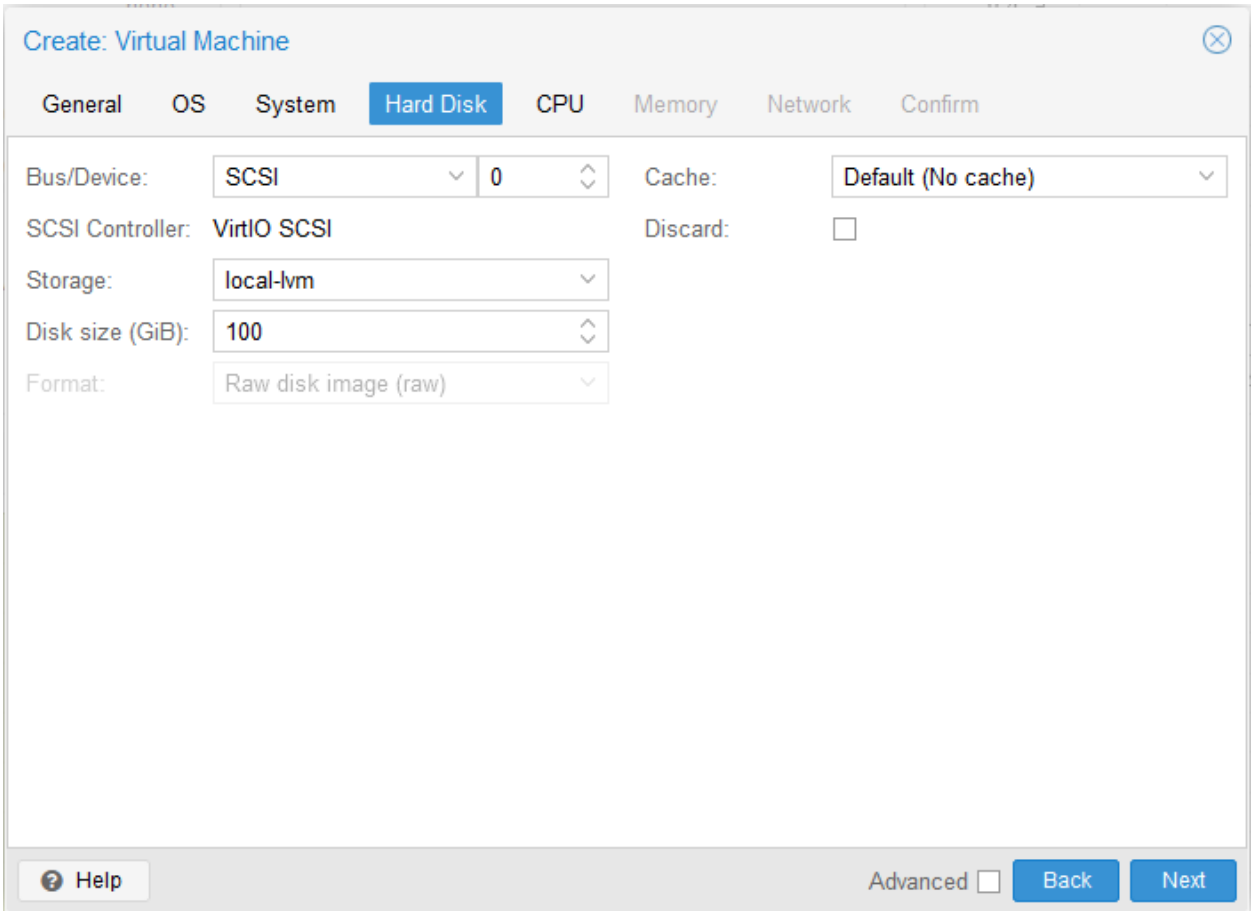
5. From here you will be prompted to find the installation media you will use for this VM. I have the latest iso of Kali installed on my local NAS. After pointing to your local media you should make sure the Guest OS is relevant to the Operating System you plan on using with your VM, for example Kali runs off Debian Linux, if you were going to use a Windows OS you would select a different type.



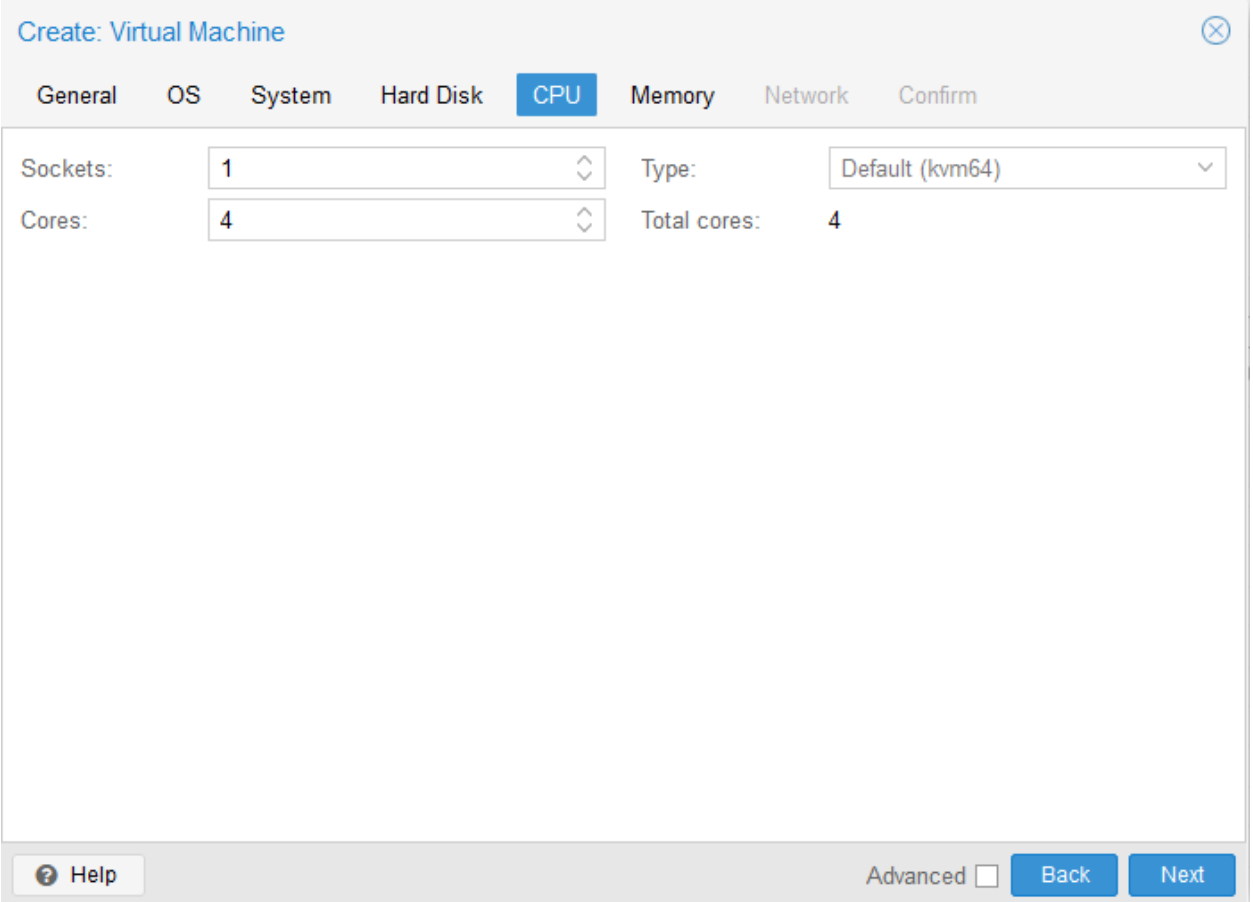
6. You can leave the graphics card as the default and press enter.



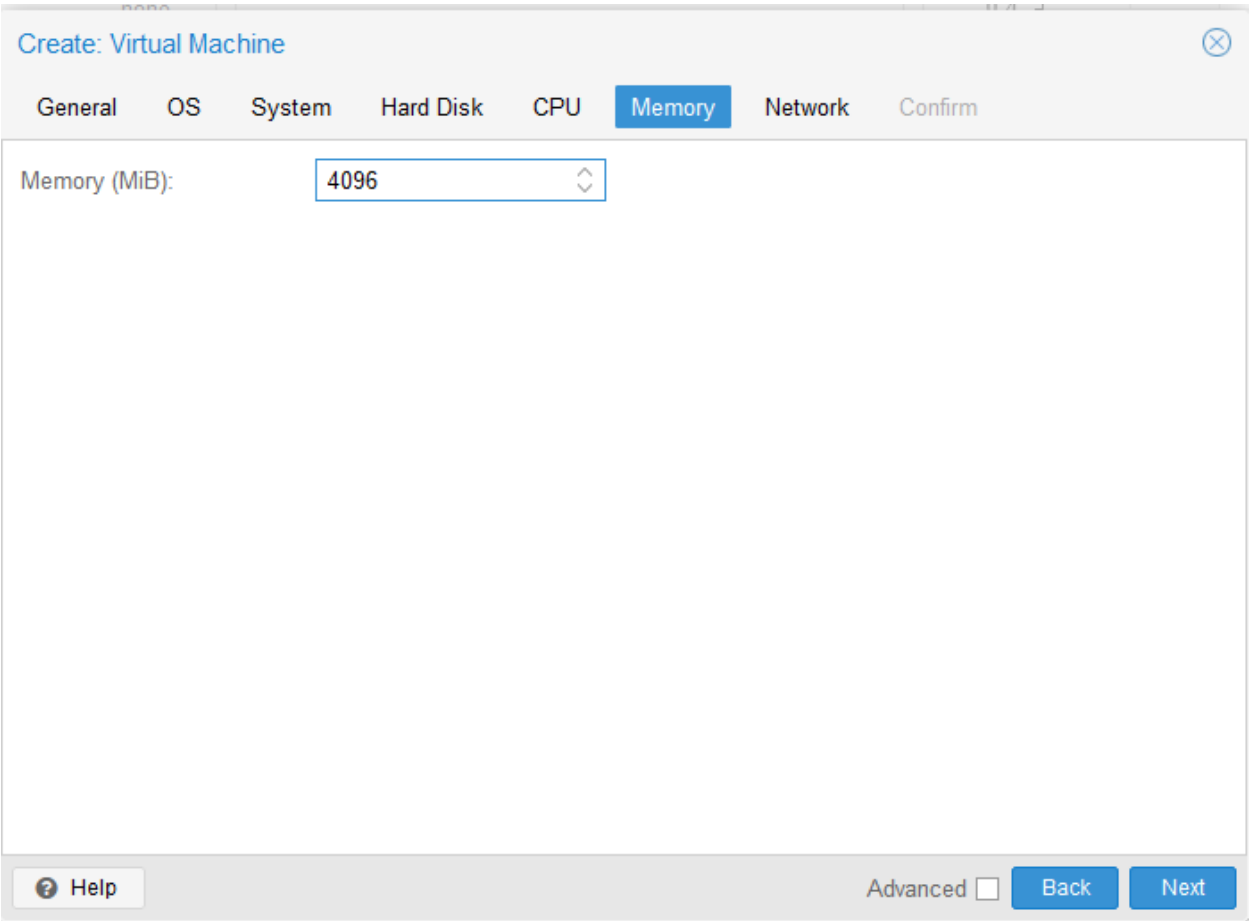
7. The next screen will prompt you for where you want to store the virtual hard disk and how big it will be. I set the size to be 100GiB and store it locally on the same partition that Proxmox is installed on. However where you store it doesn't matter as long as there is enough room.



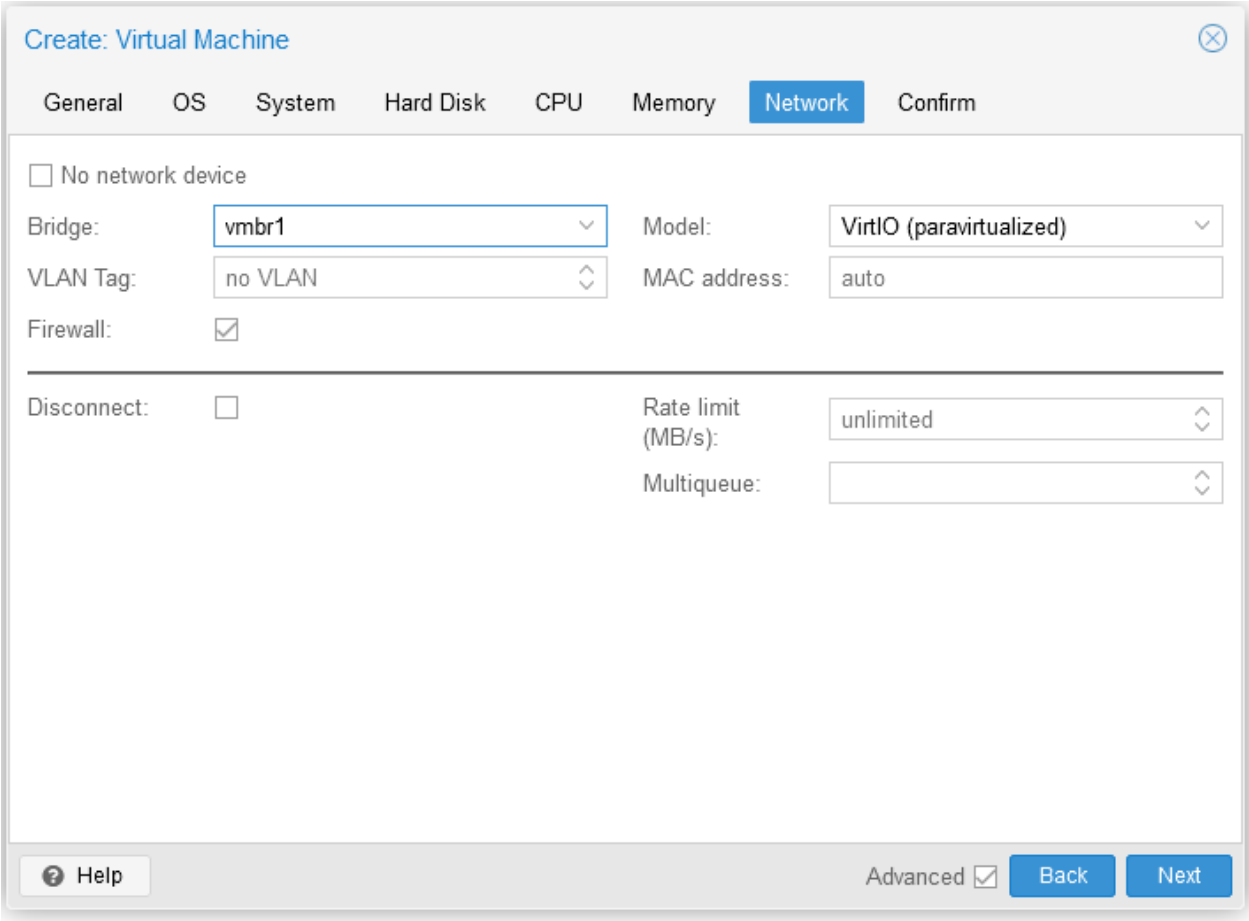
8. The next screen will ask us how many CPUs we want to allocate toward this VM. As mentioned above I recommend 4 CPU cores which will accomplish a majority of the tasks you are looking to accomplish.



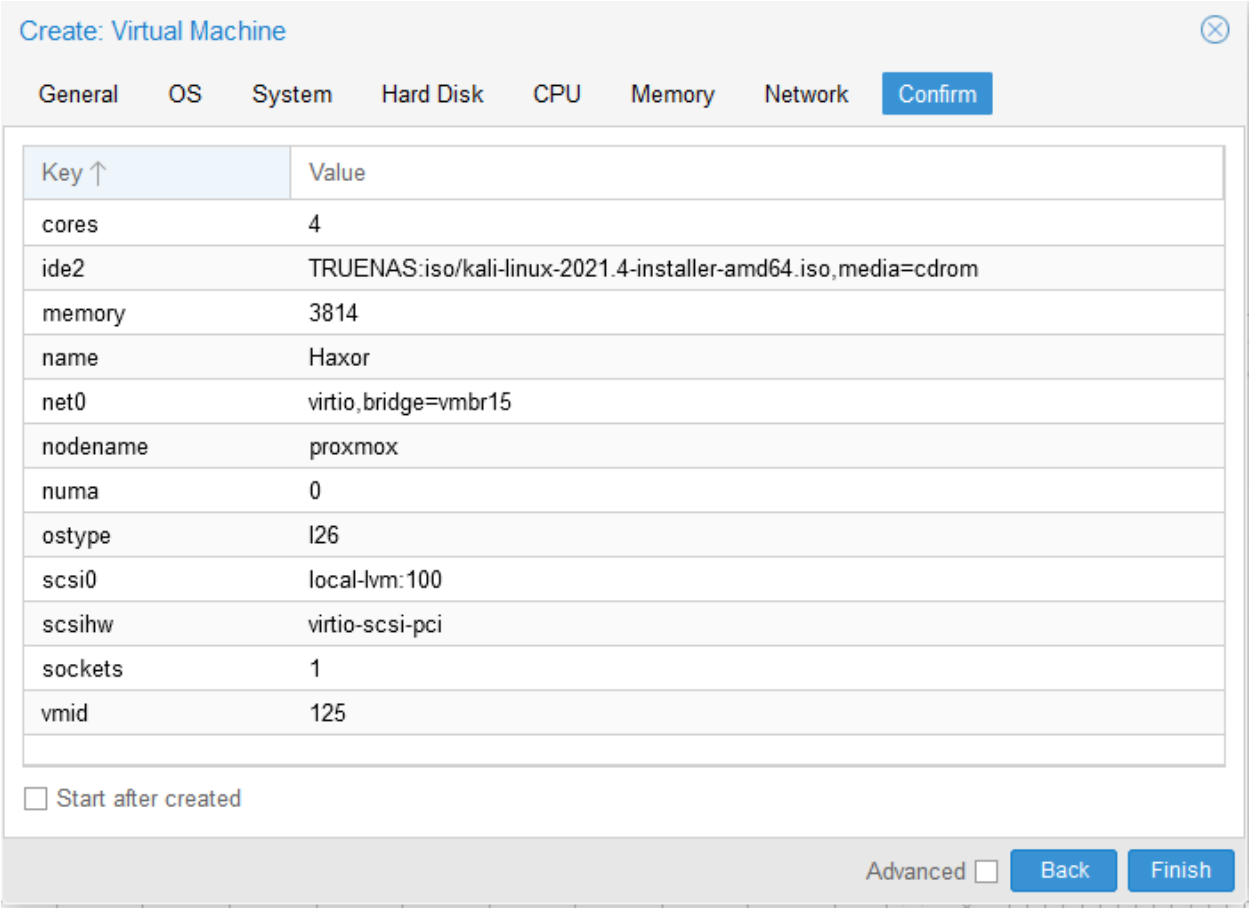
9. You will then be prompted to allocate RAM to this device. You allocate based on MiB so keep in mind you will need to do some conversions in order to properly allocate. Here is a quick website that you can use to do just that https://www.convertunits.com/from/GiB/to/MiB. I gave it 4GB of RAM as mentioned above as the minimum threshold.



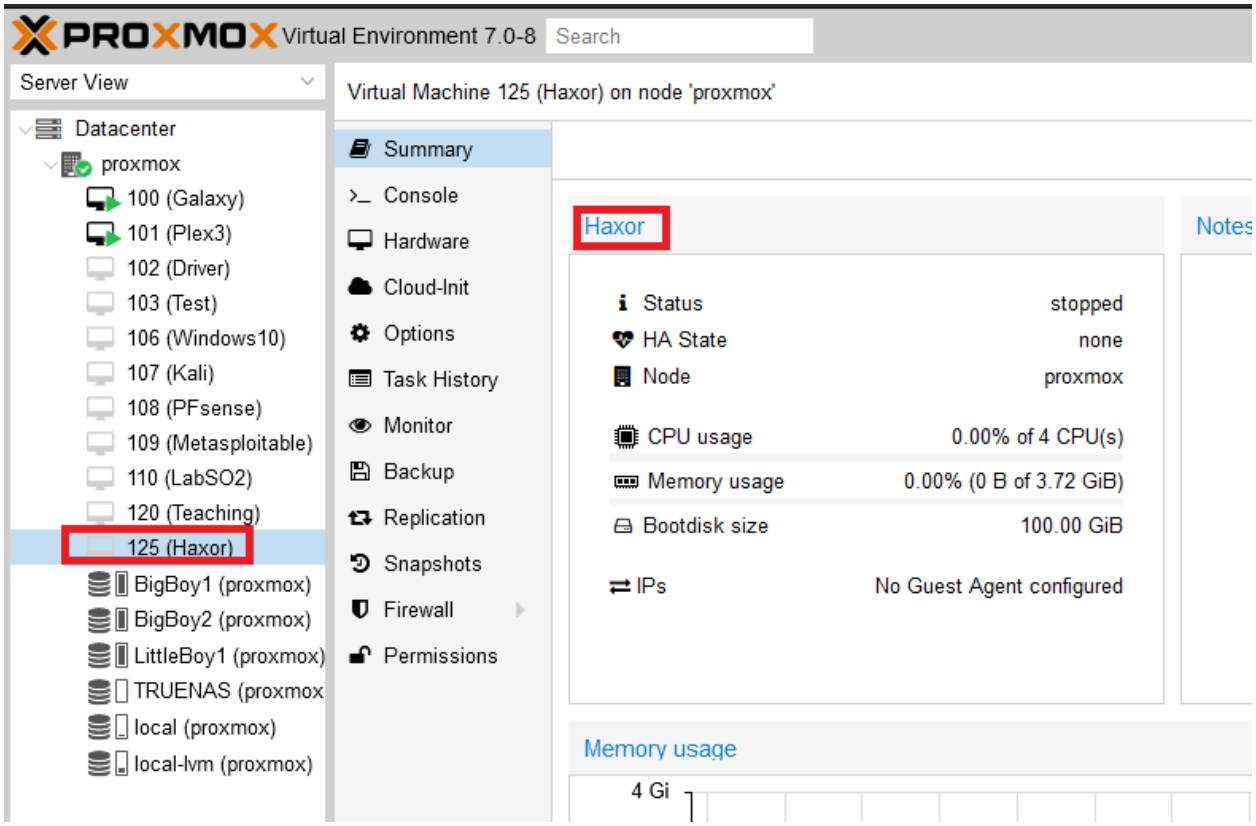
10. The next step will be to assign a NIC to your device. If you remember we created a new bridge network specifically for our home lab environment. Make sure you select that bridge device. Mine was named vmbr1 which is why I selected it.



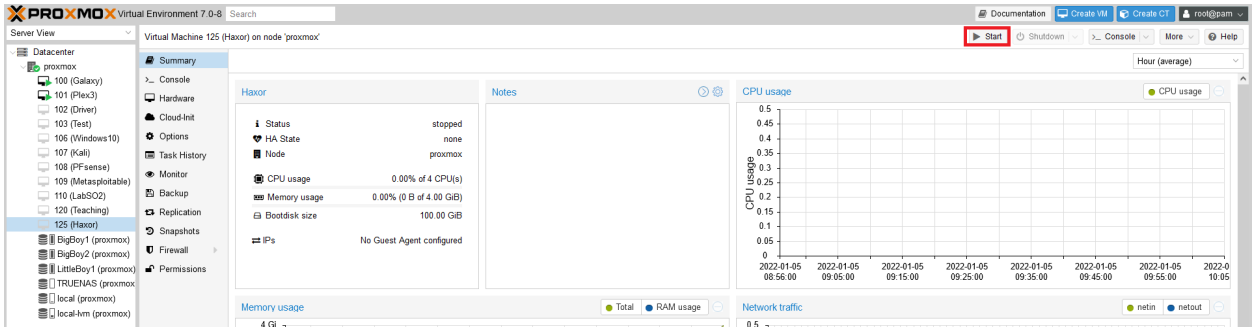
11. You can now review your created vm settings. When you're ready, click Finish to begin the process of VM creation. Mine appear as such for reference.



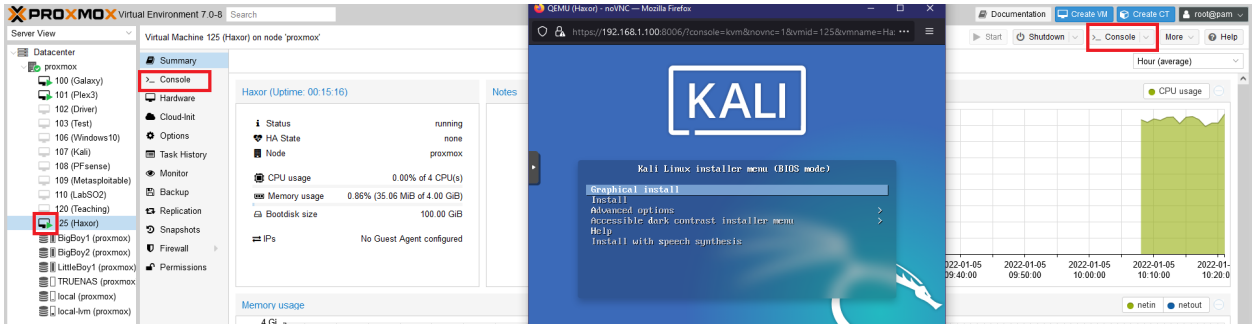
12. Once you hit finish you should notice your VM being allocated to the system. From here you should be able to access the VM from the left hand window using the name and ID you assigned to it.



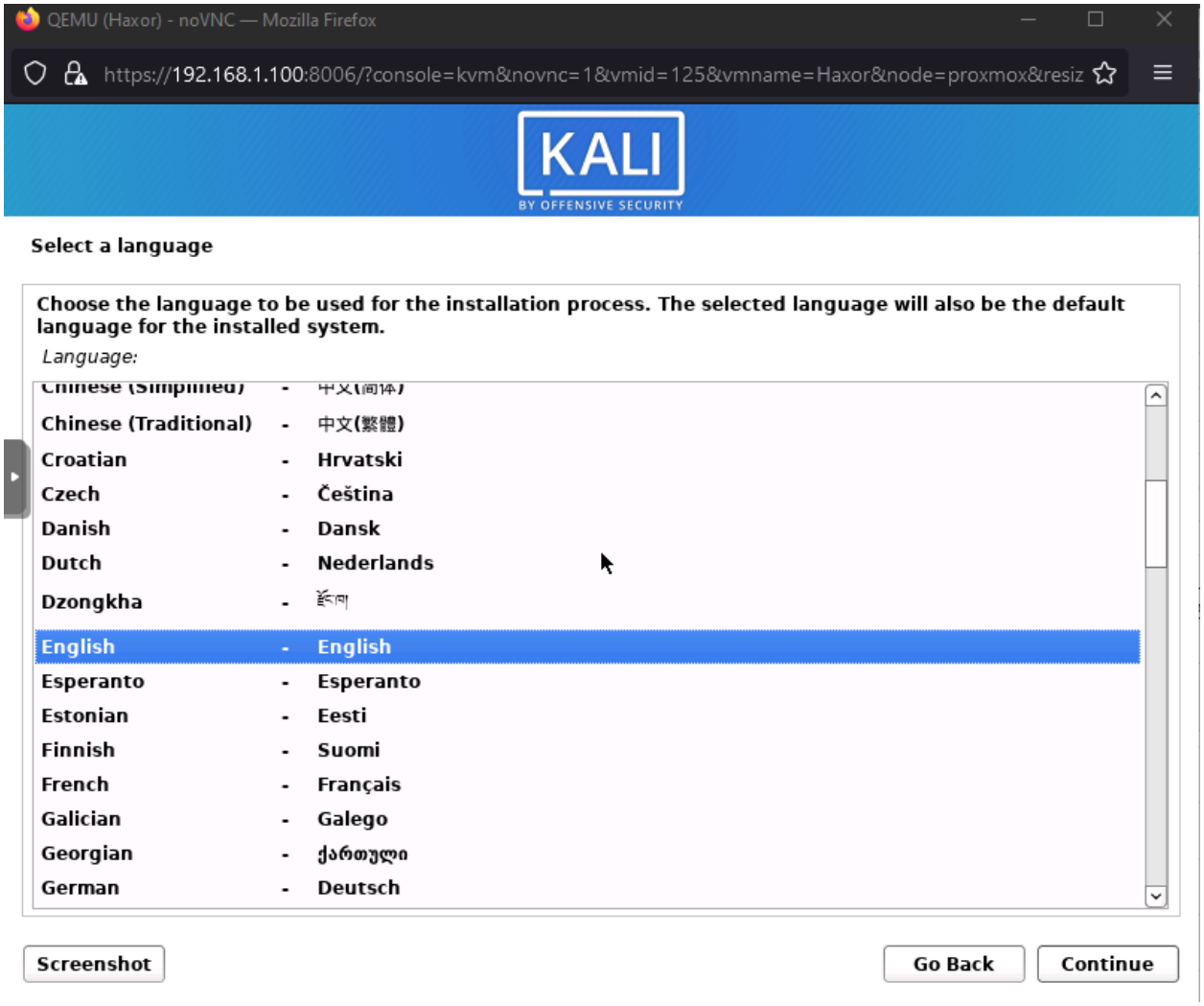
13. Once you click on the VM you will see some general useful information such as the status, how much memory/CPU usage is happening and so forth. You should see no usage because you first have to start the VM. To do that click the Start button in the top right portion of the page.



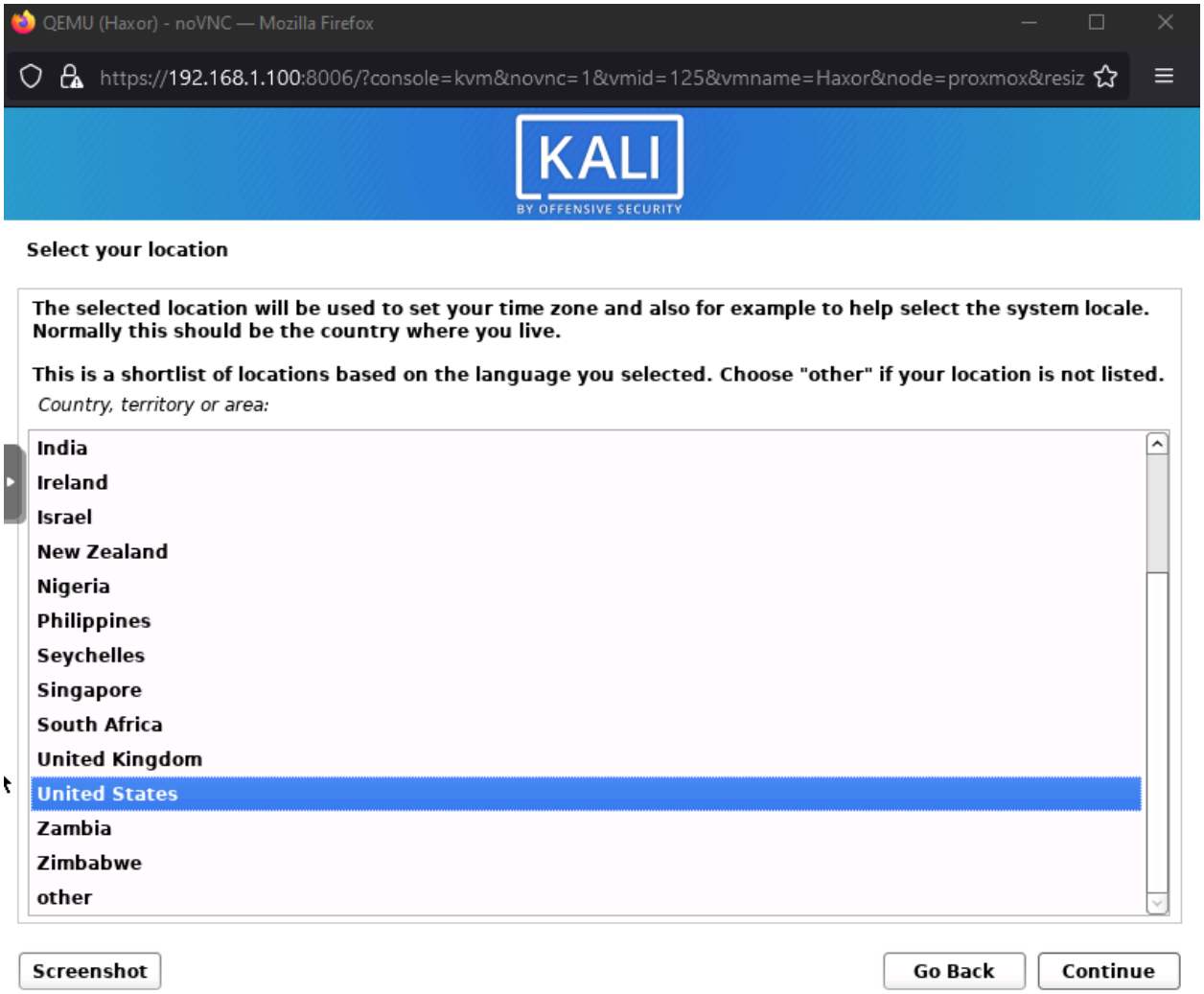
14. Once you start it you should notice a couple of things. You will see over time your Memory and CPU usage increase, and you should also notice a green play button beside your VM. In order to access your VM you will need to console into it. To do that you can either select the console tab on the left hand side of your screen or select Console in the top right hand portion which will open the console into a new window.



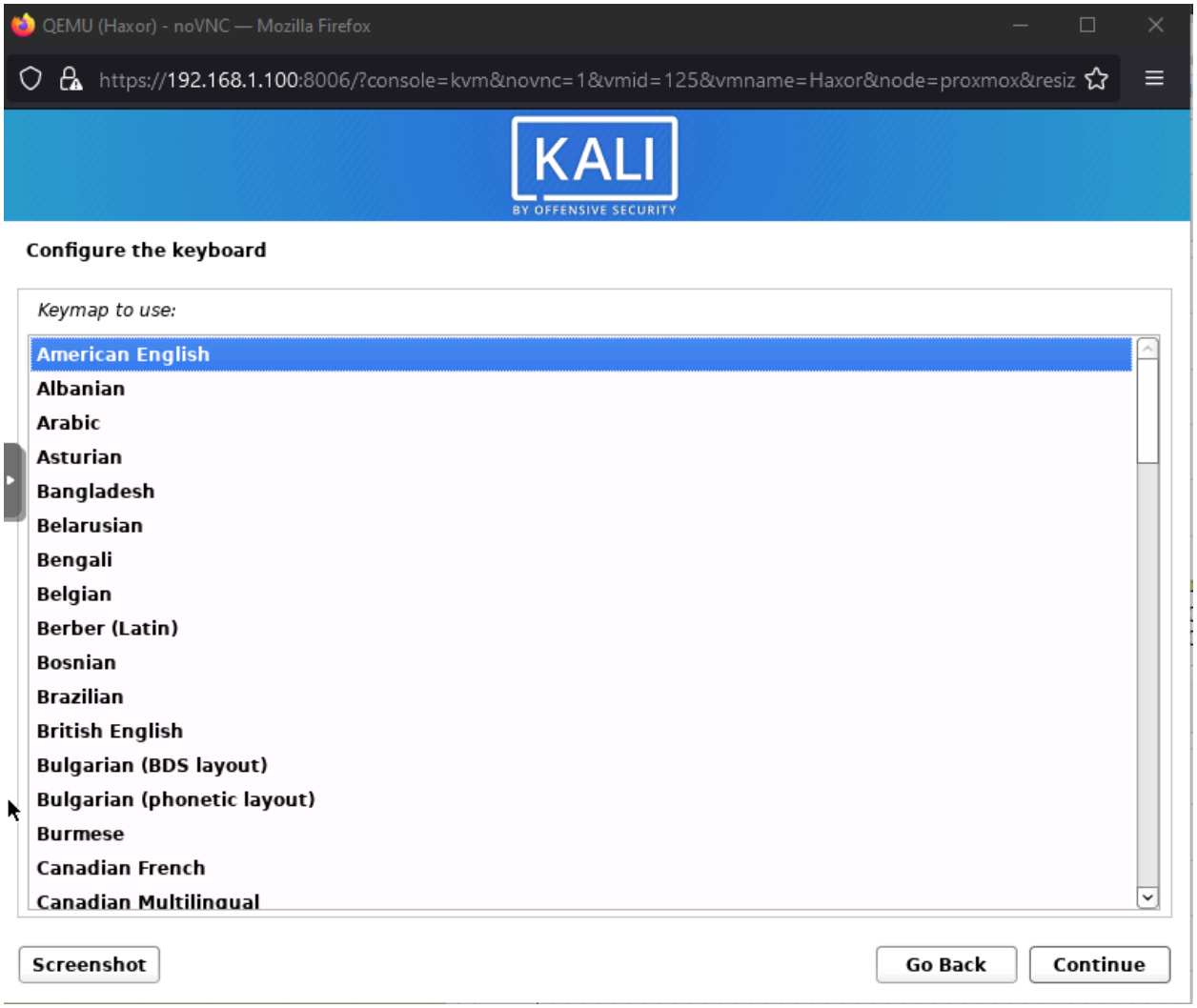
15. In the new window, you will now begin the process of installing Kali Linux. You will want to click in the console window and click enter to begin the Graphical install. From here you will be prompted to select a language, select one of your choosing.



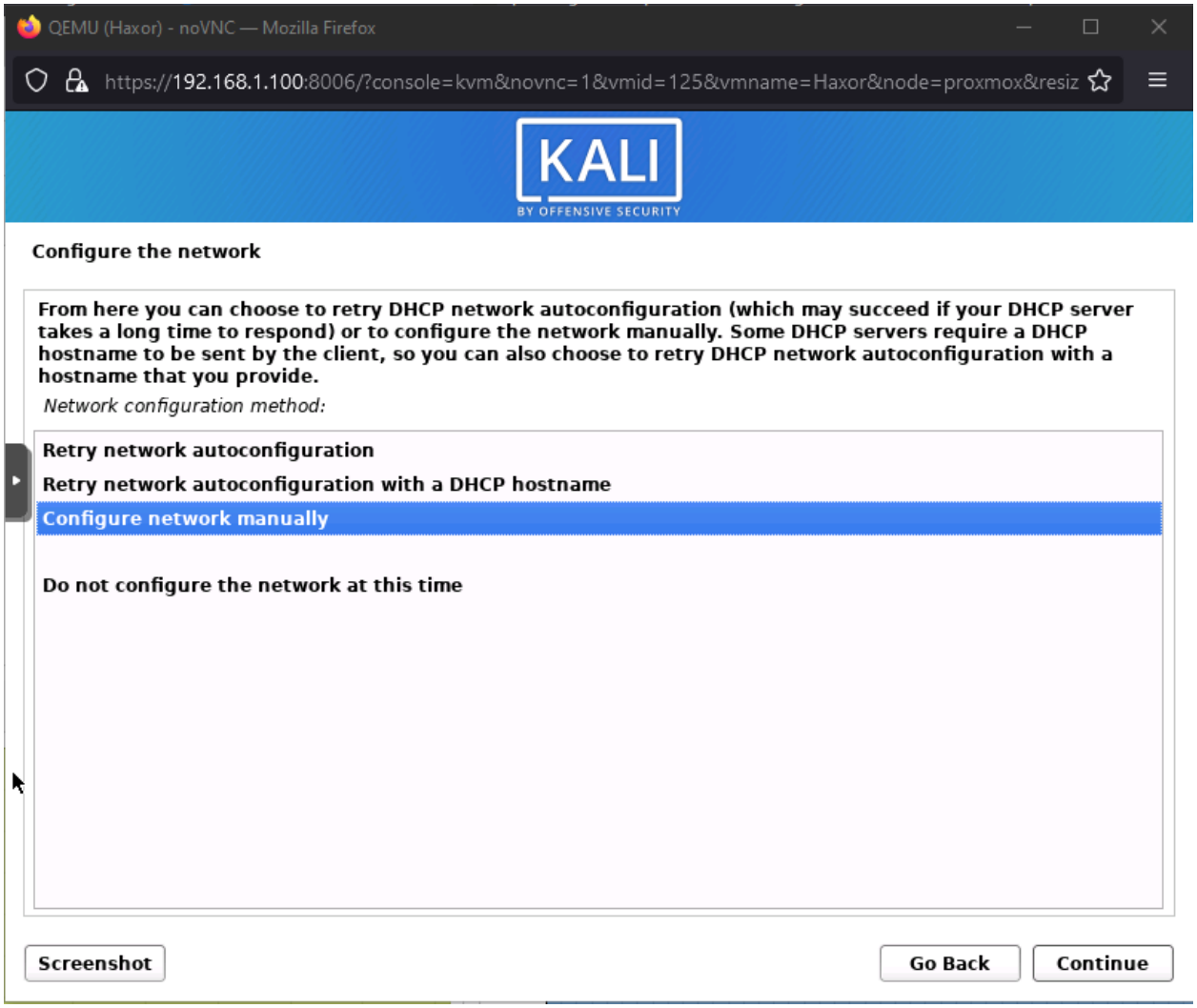
16. You will then be prompted to select your location, again select one of your choosing.



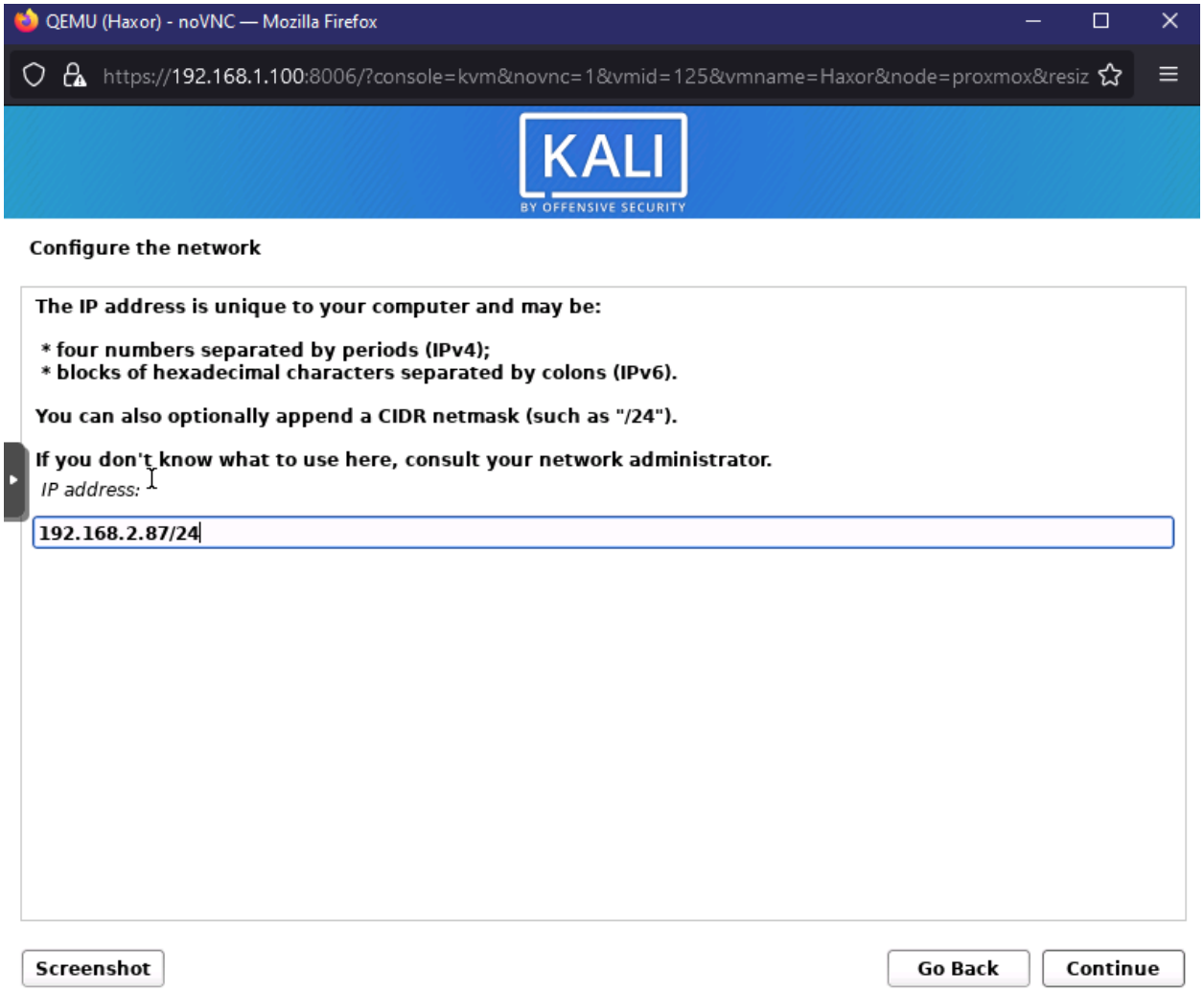
17. You will be prompted to select the keyboard configuration, again select one of your choosing.



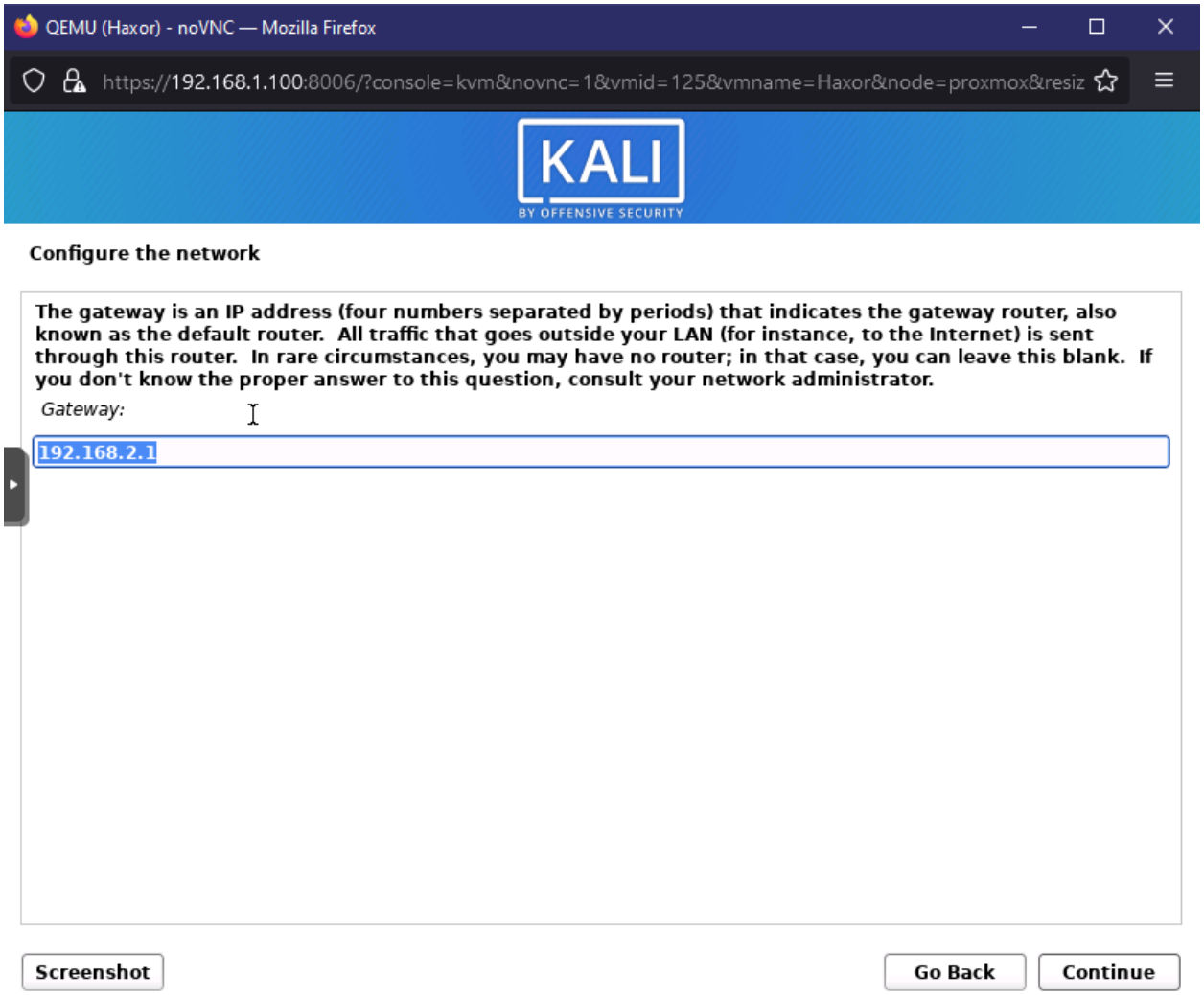
18. After a minute or so you may see a prompt saying DHCP isn't recognized, this is fine because we will be manually configuring the IP address anyway. Click next and on the next screen select Configure network manually and press continue.



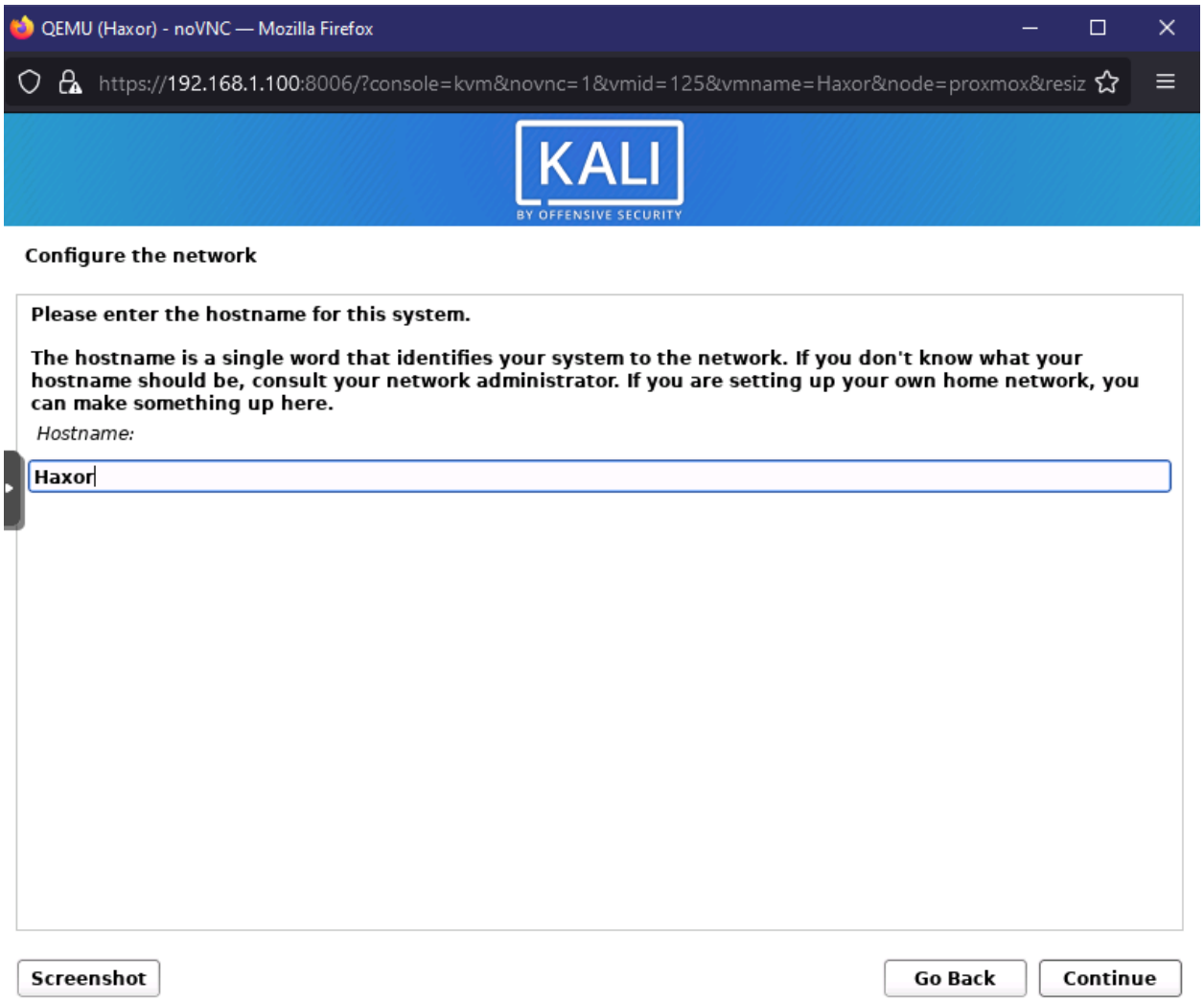
19. You will now be prompted to enter an IP address for this network. If you want to base it off my network map you would assign it an IP address in the 192.168.2.0/24 subnet. I assigned this VM an IP address of 192.168.2.87/24



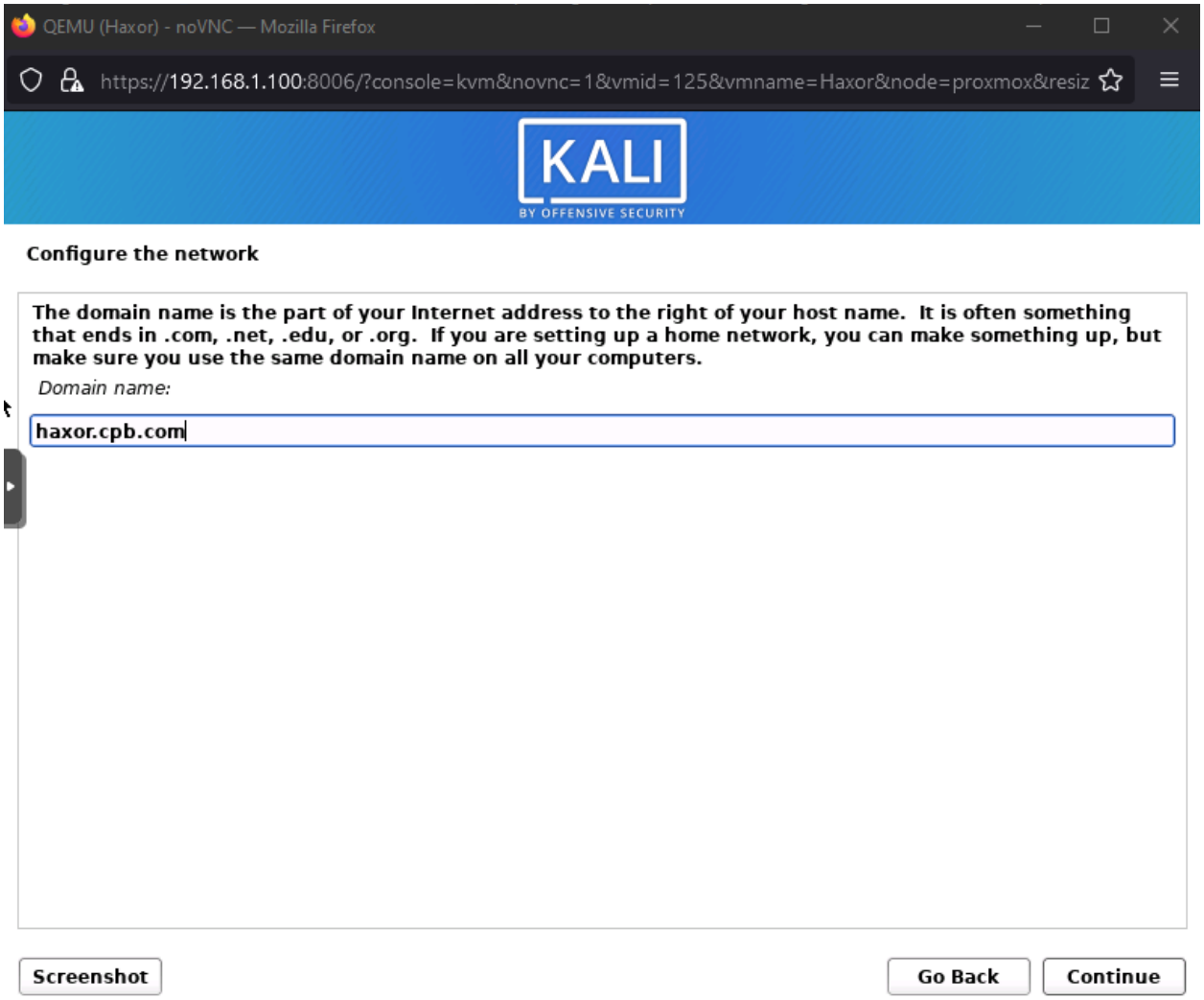
20. You will then be prompted for a default gateway. We will configure the default gateway in the PFSense tutorial coming next, but for now keeping in the spirit of the network map you will want to put an address of 192.168.2.1.



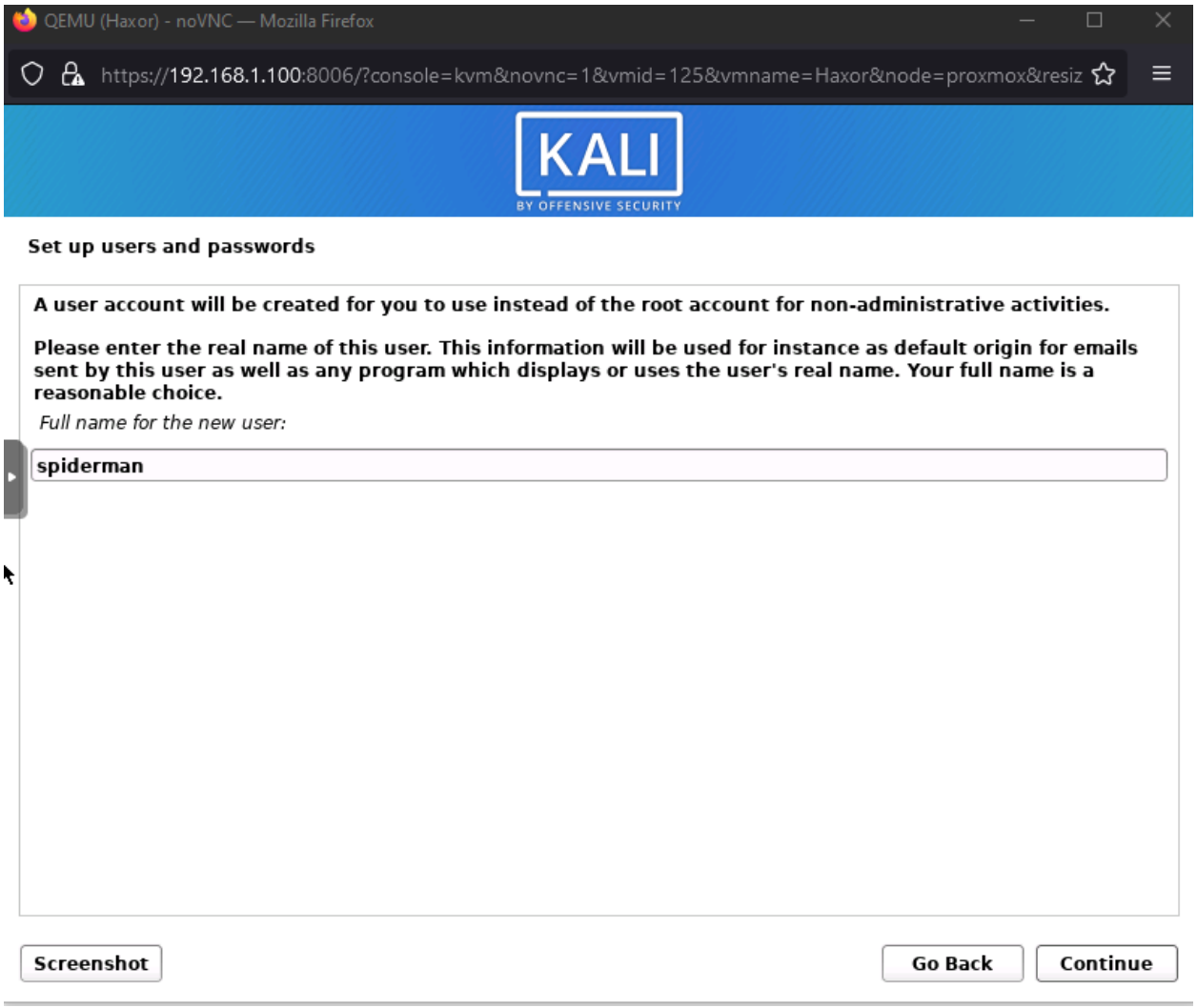
21. The next slide will ask you the same thing, because we aren't using DNS you can leave the default as the default gateway and press continue.
22. You will then be prompted for the hostname. You can name this whatever you want. I named mine Haxor and pressed Continue.



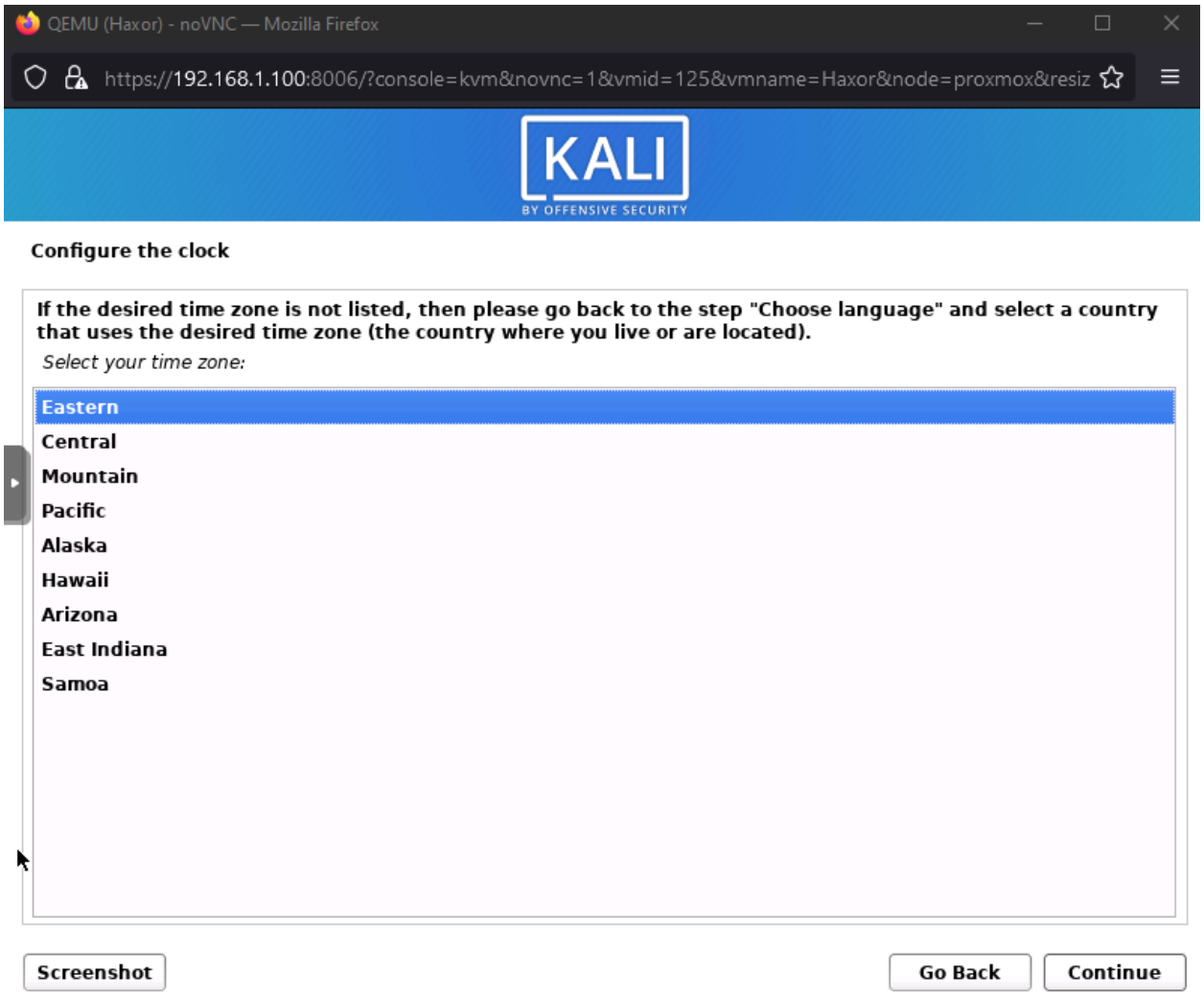
23. You will then be prompted to configure a domain name. Because we are doing a homelab environment this doesn't matter. I put haxor.cpb.com but again, put whatever you like.



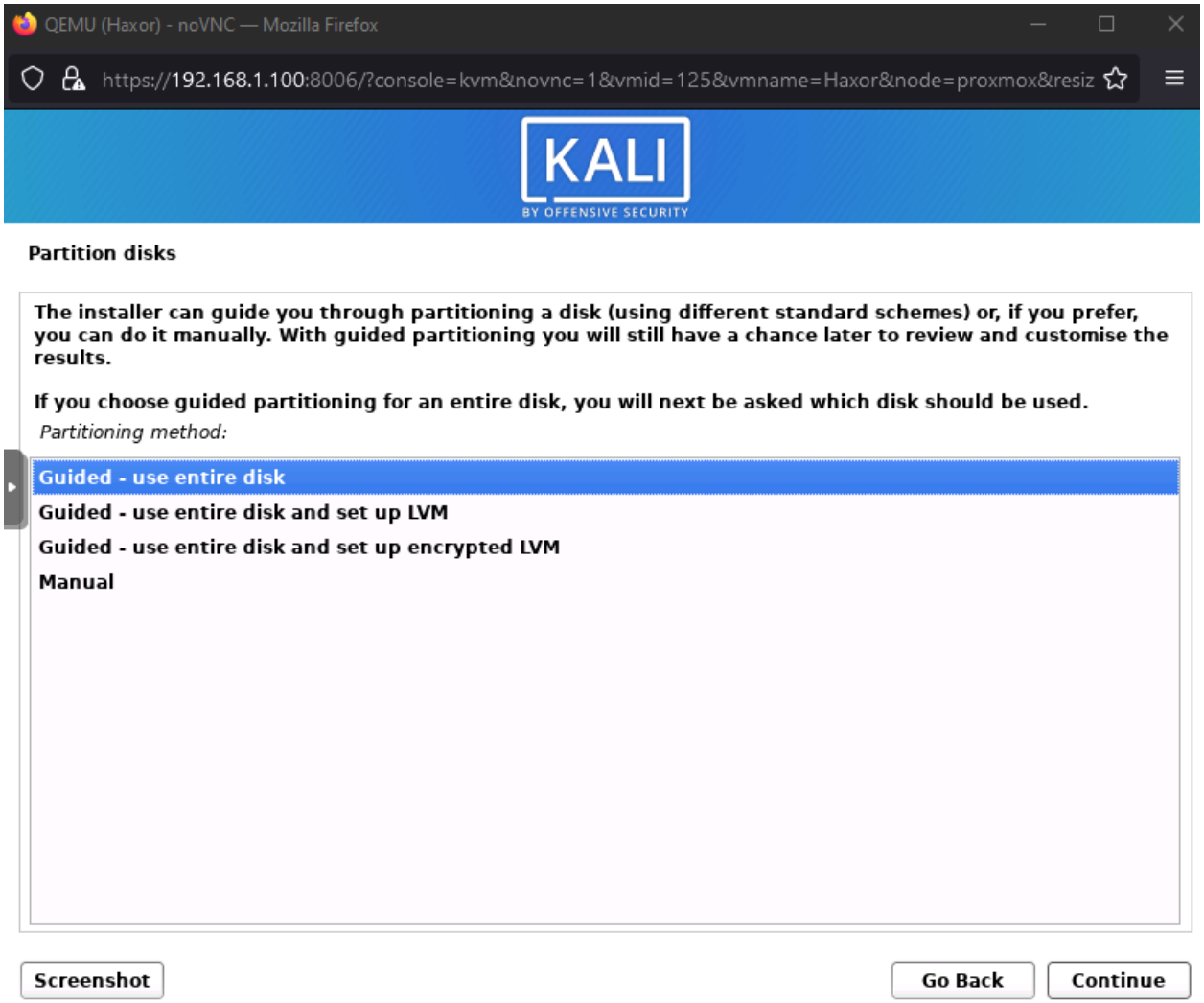
24. You will then be prompted to create a user. Create a username of your choice, just remember to take notes somewhere of your login credentials for the future. I created a user named spiderman.



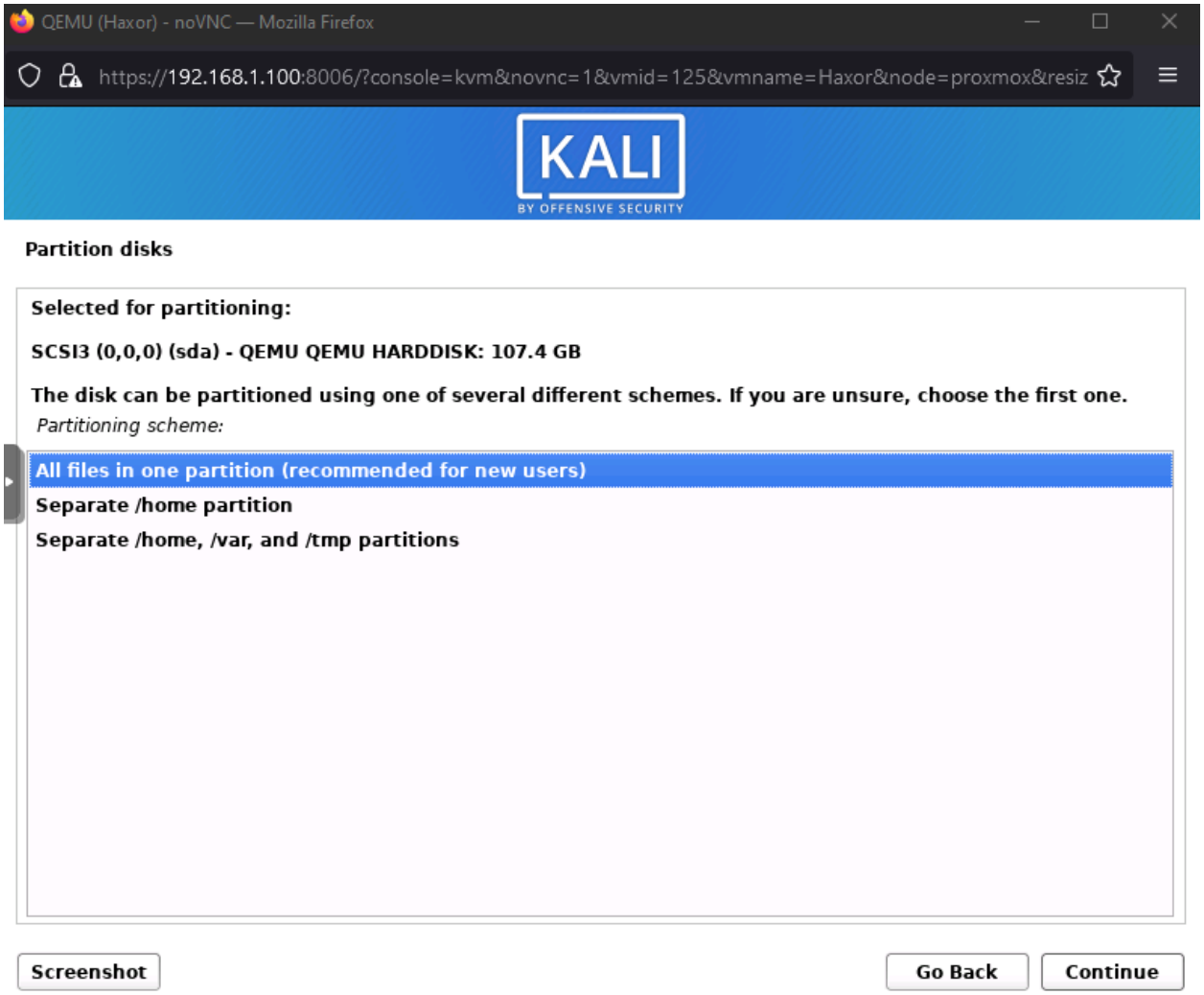
25. I then created a username and password for spiderman; spiderman:toor
26. You will then be prompted to configure a timezone. Enter your desired preference and select continue.



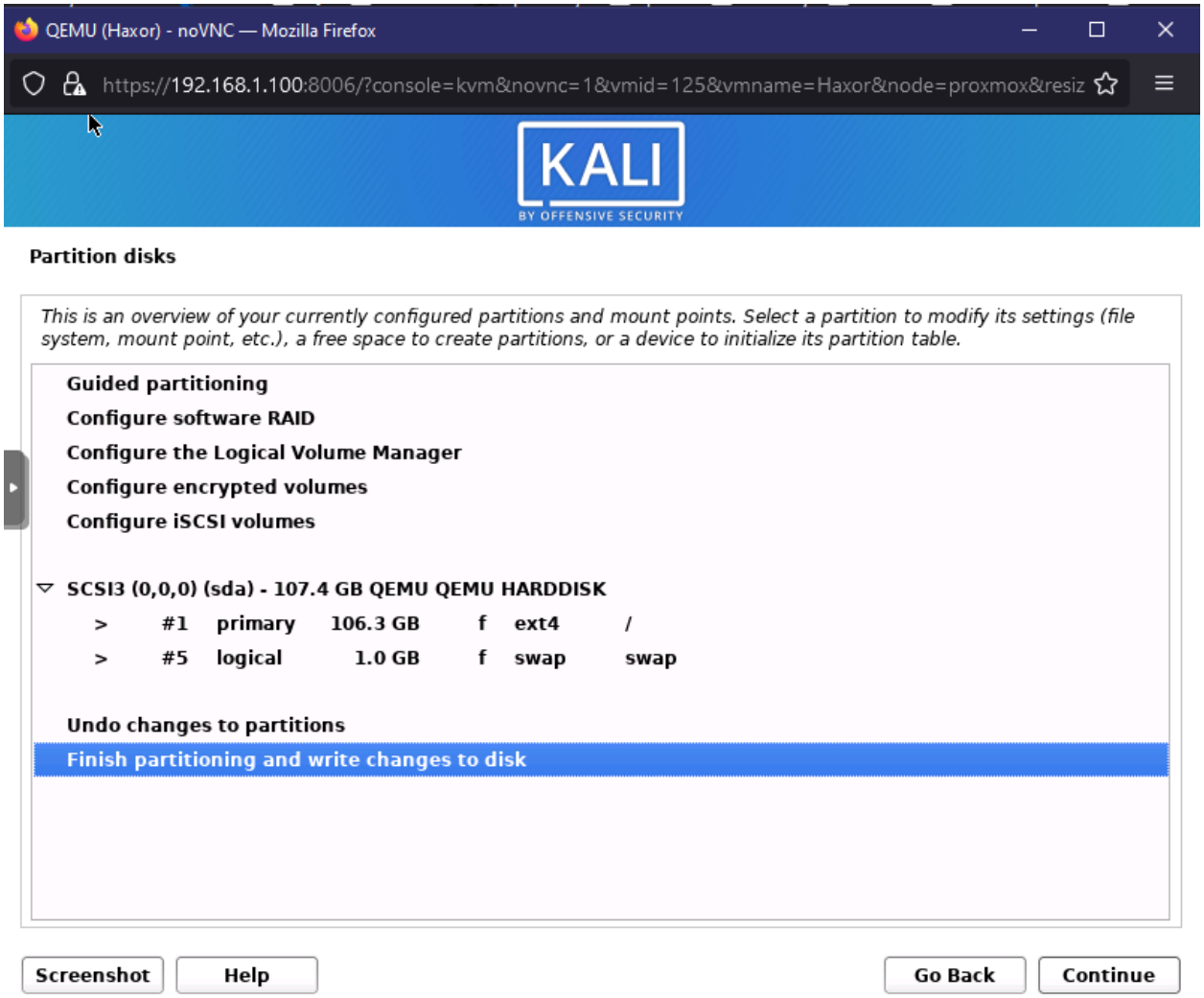
27. You will then be asked how you want to partition the disks. It's best practice to just you the whole virtual disk. Select Guided - use entire disk and press Continue.



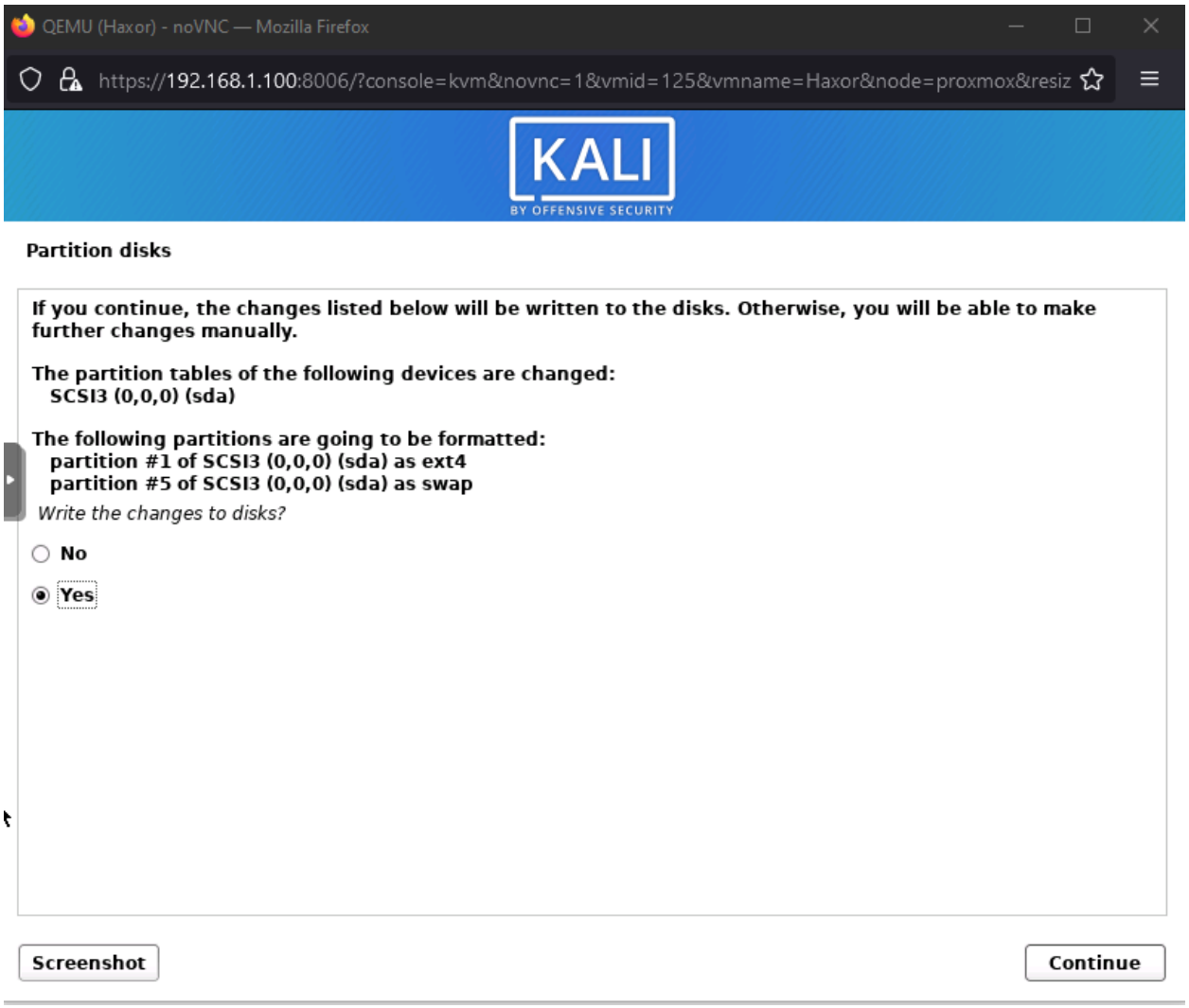
28. Select the only available disk and press continue.
29. It's best if you just have all files on one partition, select All files in one partition and press continue.



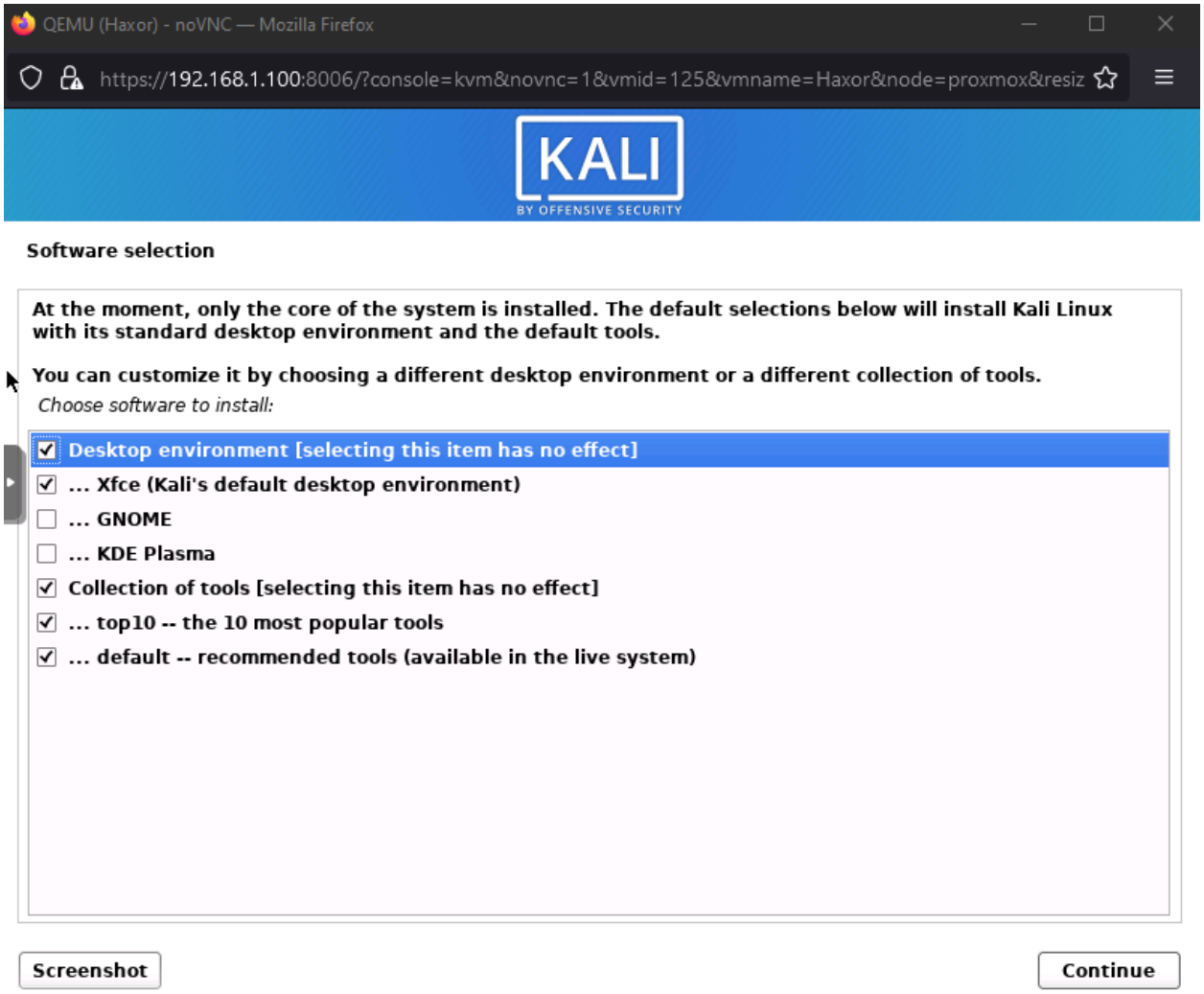
30. Press continue to commit your changes.



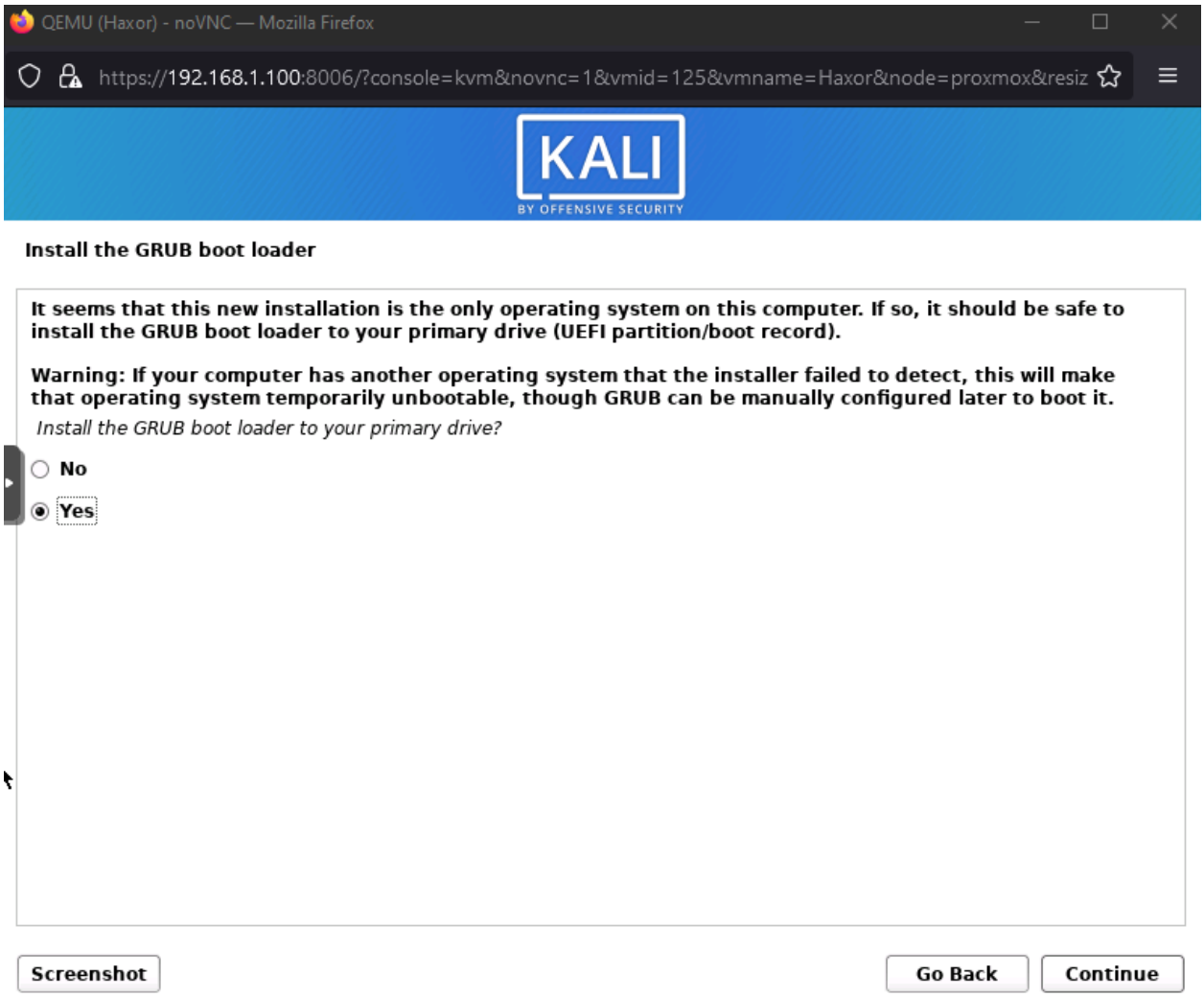
31. Change the prompt to Yes and commit to the changes.



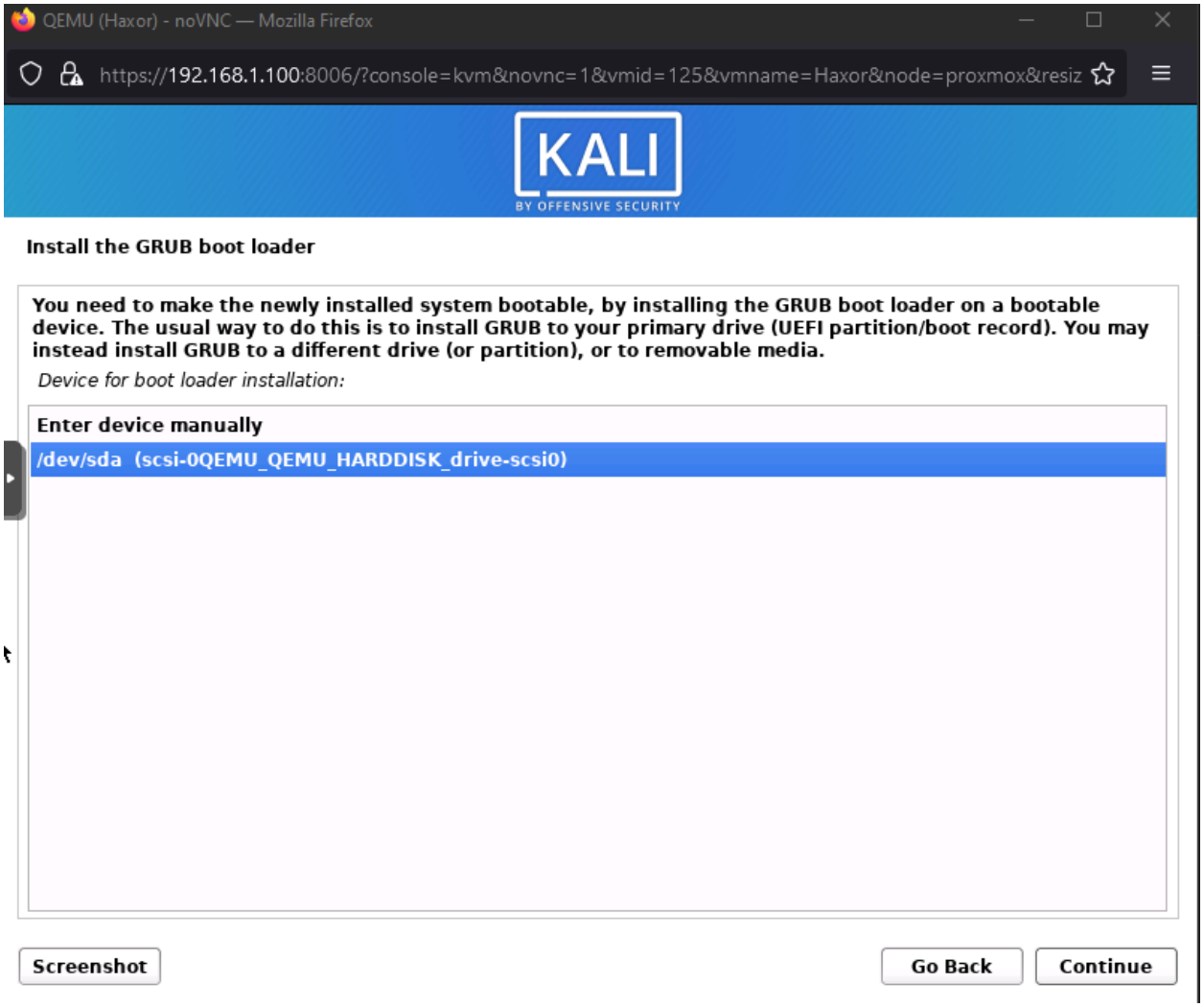
32. You will now wait for the initial install to commence. After a while you will be prompted to select the software you want to install. I would just leave everything at the default and press continue to begin the full install.



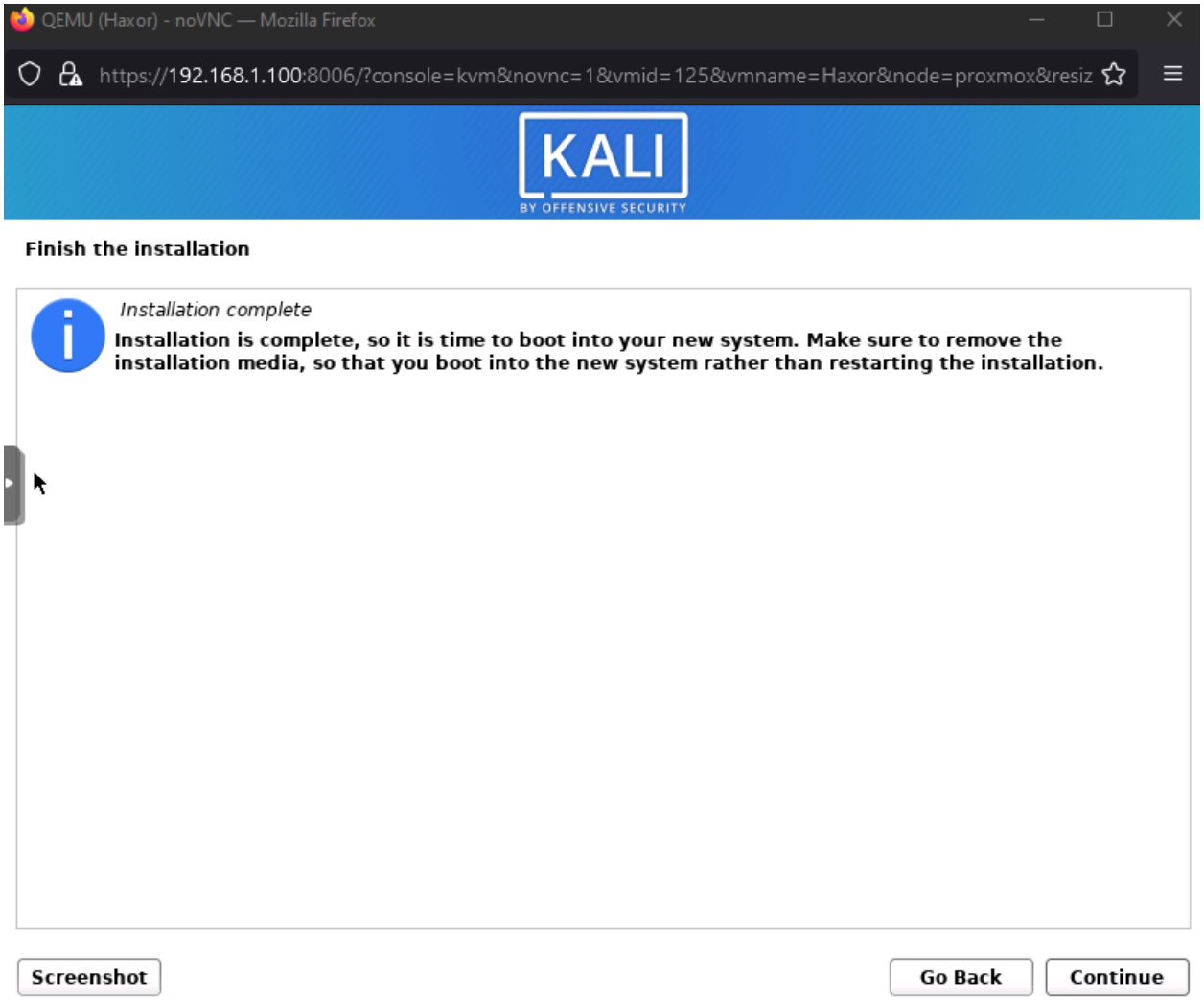
33. After some time, you will be prompted if you want to install the GRUB boot loader. Keep Yes and press Continue.



34. Select the only available drive location /dev/sda and press Continue to point where you want to install the GRUB boot loader.



35. Now you will just have to wait for Kali to fully finish installing. Once you do restart the VM and you should be finished!



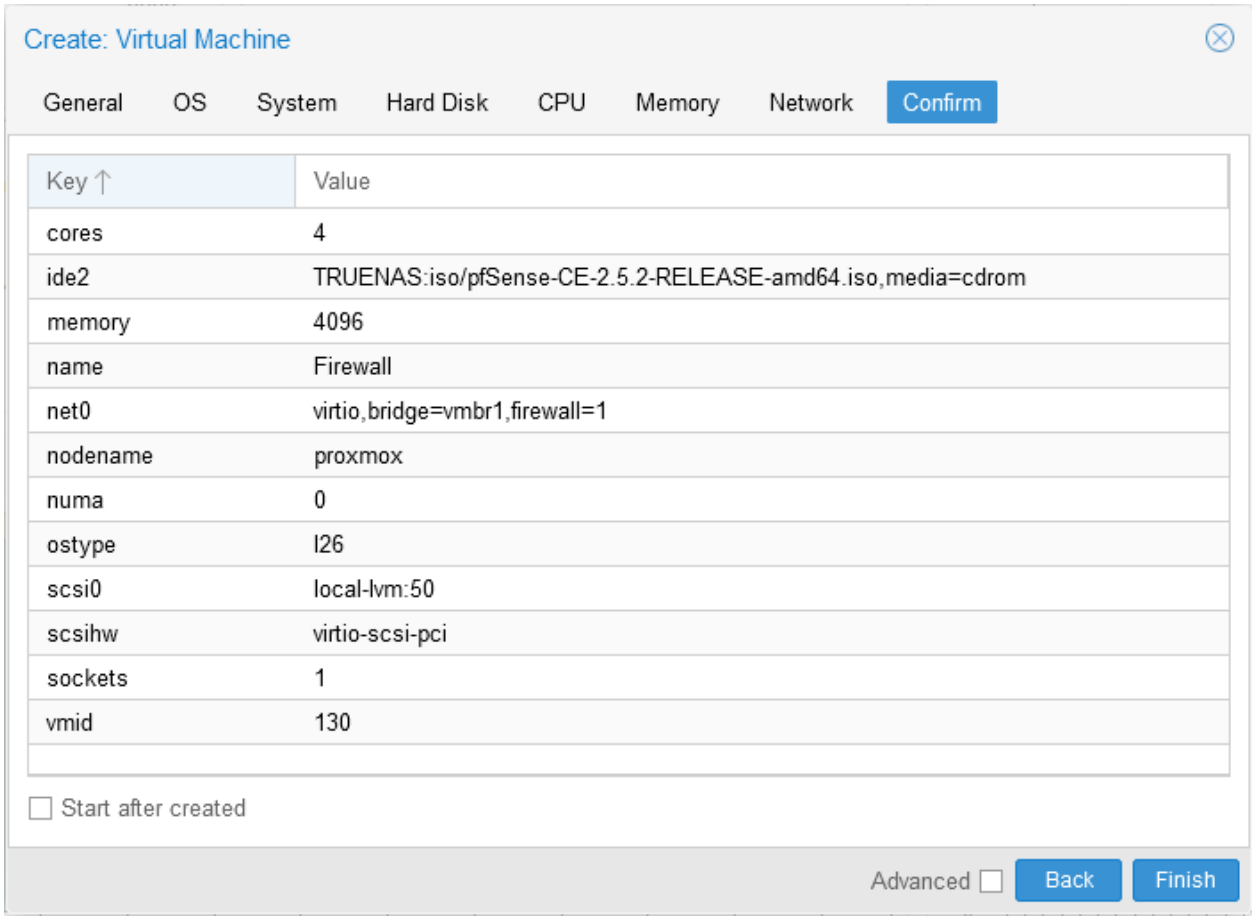
36. Lastly, once your Kali machines boots you will want to login and manually assign an IP address based off your schema. In my instance my IP address is located in “red space” so I gave it an IP address of 192.168.2.69. You can also set your default gateway to 192.168.2.151 if you’re following my guide as that is what we will be setting it to in PFsense.

```
mitchell@Hunter: ~

┌──(mitchell㉿Hunter)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP gr
oup default qlen 1000
    link/ether 1a:e4:92:68:2e:f0 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.69/24 brd 192.168.2.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::18e4:92ff:fe68:2ef0/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(mitchell㉿Hunter)-[~]
└─$ sudo ifconfig eth0 192.168.2.69 netmask 255.255.255.0
```

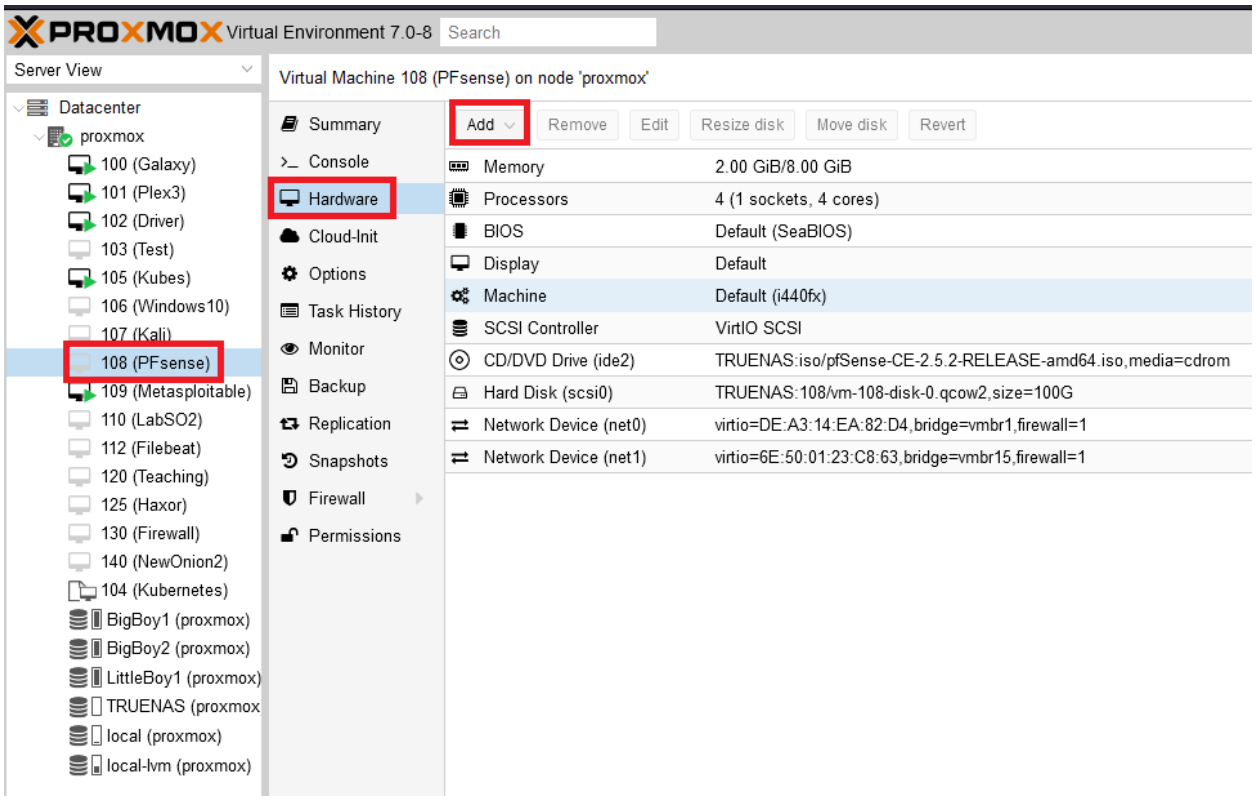
2. PFSense

Next we will install and configure PFSense, our internal firewall. This is a relatively straight-forward process with the only exception being some after install configurations. I won't provide screenshots for installing the VM through proxmox, just know that the process is the same as in Kali.

1. The first step will be to download and upload the latest version of PFSense into your proxmox environment. The link for the download is listed below.
    a. https://www.pfsense.org/download/
2. After you have completed the upload, create a new VM with the following parameters
    a. General
        i. VM ID: First Available
        ii. Name: Your Preference
    b. OS
        i. Point to your PFSense .iso
    c. System
        i. Leave default
    d. Hard Disk
        i. 50GB
    e. CPU
        i. Cores: 4
    f. Memory
        i. Memory (MiB) 4096
    g. Network
        i. Bridge: vmbr1 (the network bridge with the 192.168.2.100/24 network connection)
    h. Should look similar to below

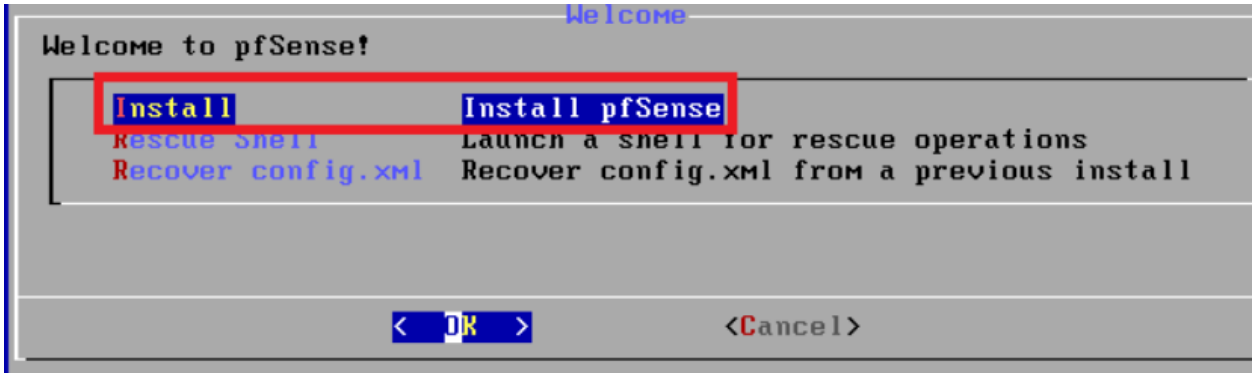


3. Once your VM finishes allocating we will then add another network interface. What this allows us to do is have 2 different Proxmox networks talk to each other through this PFSense firewall.
    a. The first thing step is making sure your VM is off and click on your VM and navigate to the Hardware tab.

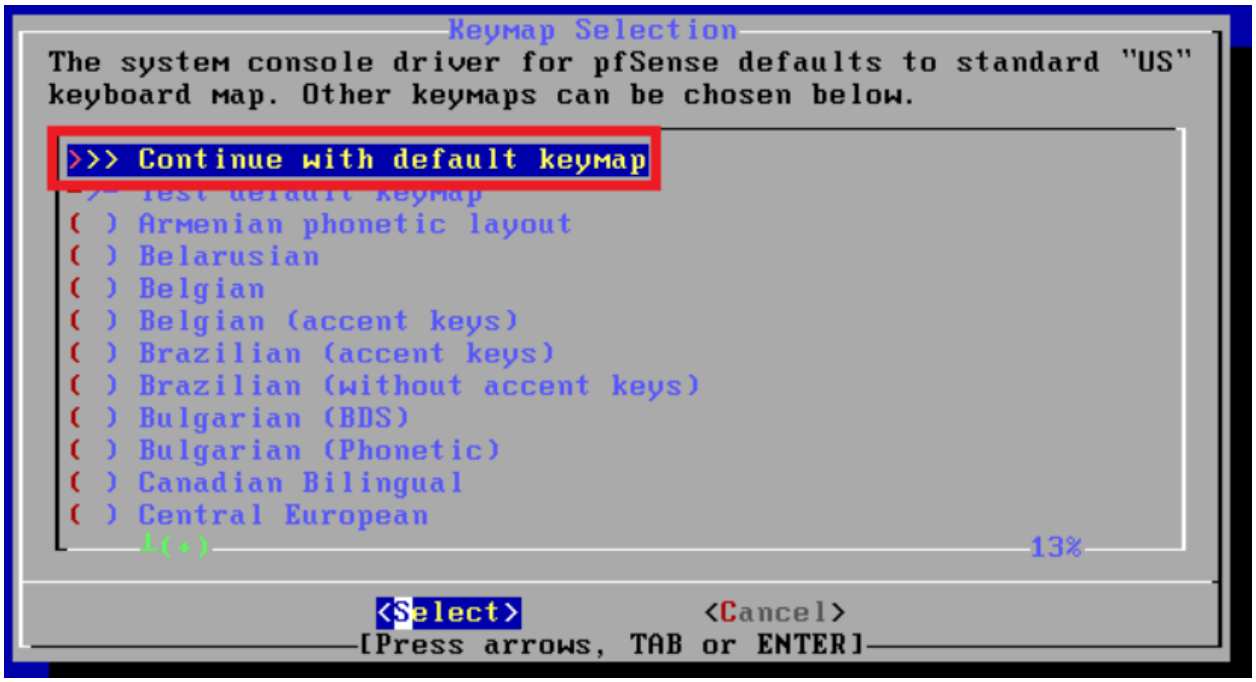


b. Next you will want to add another interface, as you can see from the previous screenshot this was already configured. To add another network interface simply click the Add button on the top and select which network bridge you want to add to the VM. In this instance I've added **vmbr15**. We created these 2 interfaces at the start of the guide. By doing this we are going to allow communication between these 2 networks through our firewall. In the future, you could practice writing and creating firewall rules using this setup.

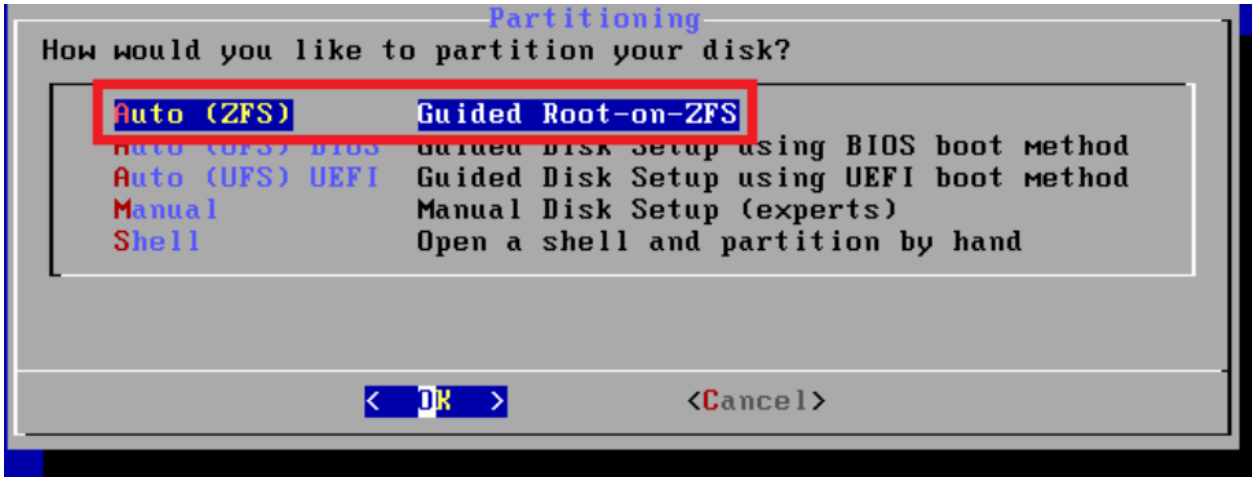
4. From here you will want to power up your PFSense router and accept the EULA when it pops up.
   a. You will then be prompted to begin install, press enter to start install.



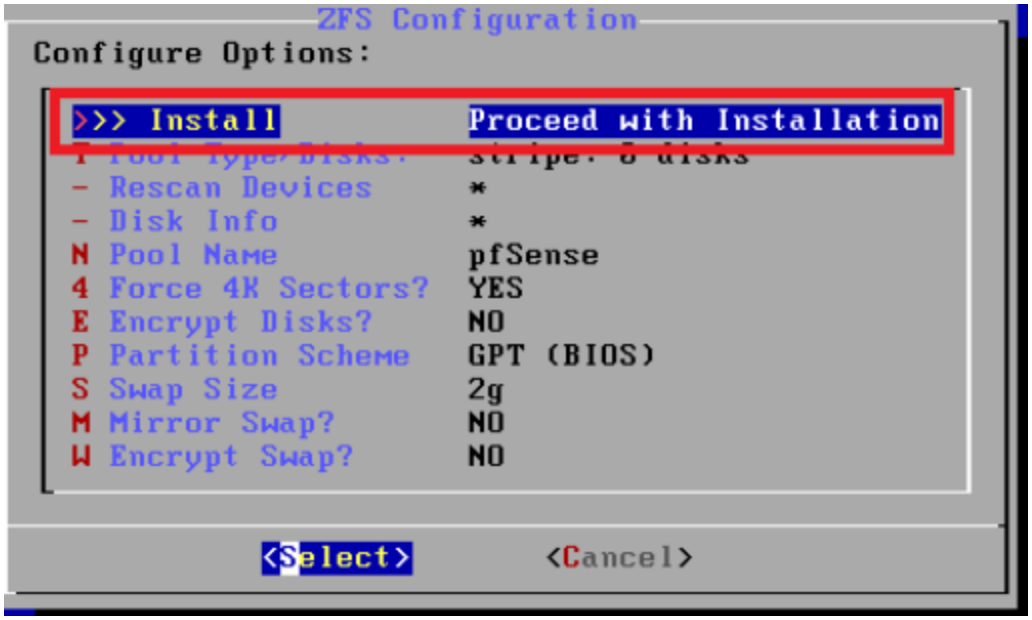
   b. You will then be asked what type of keyboard layout would you like. You can leave the default and press enter.



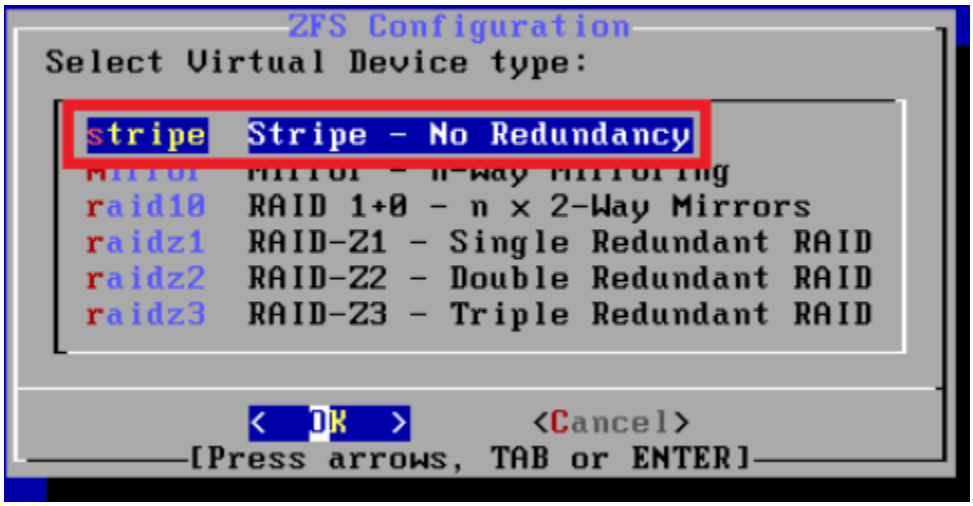
c. Press enter to use ZFS for disk partitioning.



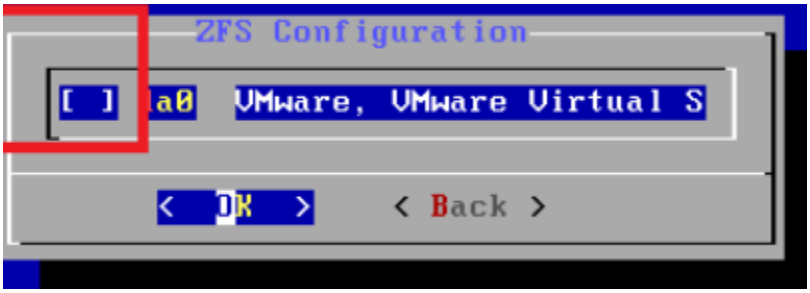
d. Then press enter to begin the install process.



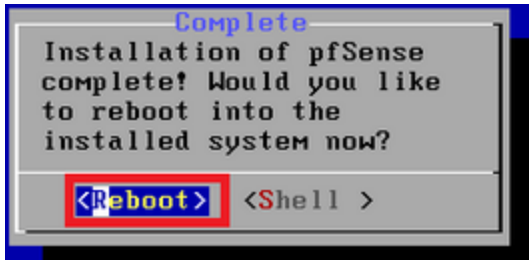
e. You will then be asked if you want to raid your router. Because this is a very lightweight deployment that won't be necessary so you can press enter.



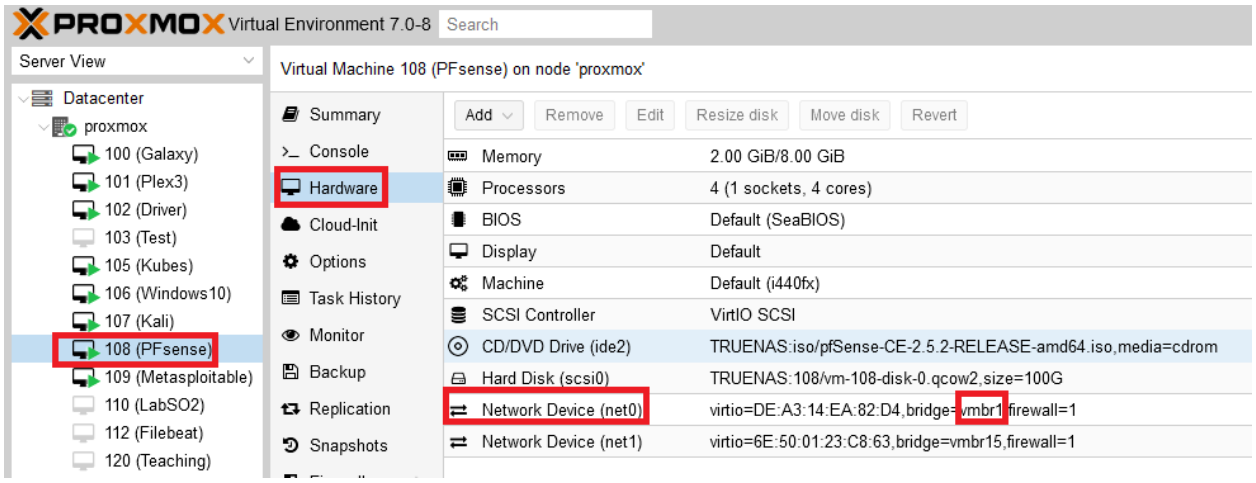
f. Press spacebar to select the only virtual disk available to install the OS on.



g. Confirm that you want to delete the contents of the disk, then when prompted select no to opening a shell and then lastly reboot the machine to finish the initial installation.



5. Now that the OS has been installed, we're going to configure our networking so that PFSense can act as our internal router for our lab environment. This means that our lab will be able to have 2 different networks with the ability to talk to one another. See the network map previously listed for an example of this.
    a. Once you boot into PFSense you will be presented with a couple of different options. What we need to do is configure both interfaces to match our network map. We will first start by configuring our WAN interface which will be representing our "red space". In PFsense your WAN should correlate with vtnet0 which corresponds with our proxmox vnet0 which should be vmbr1 if you're following my guide.



    b. Once you have confirmed these settings press 2 to begin the configuration.Then press 1 to configure the WAN interface. Press N to manually configure the default gateway then put in the IPv4 address you want your WAN gateway to be, in the lab example it will be 192.168.2.151/

```
Enter an option: 2

Available interfaces:

1 - WAN (vtnet0 - static)
2 - LAN (vtnet1 - static)

Enter the number of the interface you wish to configure: 1

Configure IPv4 address WAN interface via DHCP? (y/n) n

Enter the new WAN IPv4 address.  Press <ENTER> for none:
> 192.168.2.151
```

c. You will then be asked to put a subnet mask, again if you're following my guide put 24. You will then be asked for your upstream gateway but just press enter to leave it blank. Type n and press no to not put an IPv6 DHCP address and then press enter to not configure IPv6.

```
Subnet masks are entered as bit counts (as in CIDR notation) in pfSense.
e.g. 255.255.255.0 = 24
     255.255.0.0   = 16
     255.0.0.0     = 8

Enter the new WAN IPv4 subnet bit count (1 to 31):
> 24

For a WAN, enter the new WAN IPv4 upstream gateway address.
For a LAN, press <ENTER> for none:
>

Configure IPv6 address WAN interface via DHCP6? (y/n) n

Enter the new WAN IPv6 address.  Press <ENTER> for none:
>
```

d. You will then be asked if you want to revert to http over https, type n and press enter. Then you will be notified of the changes and get kicked back to the main shell. From here you will repeat the same steps except this time only changing the default gateway IP for your LAN network to 192.168.3.1/24. When finished your home screen should look like this.

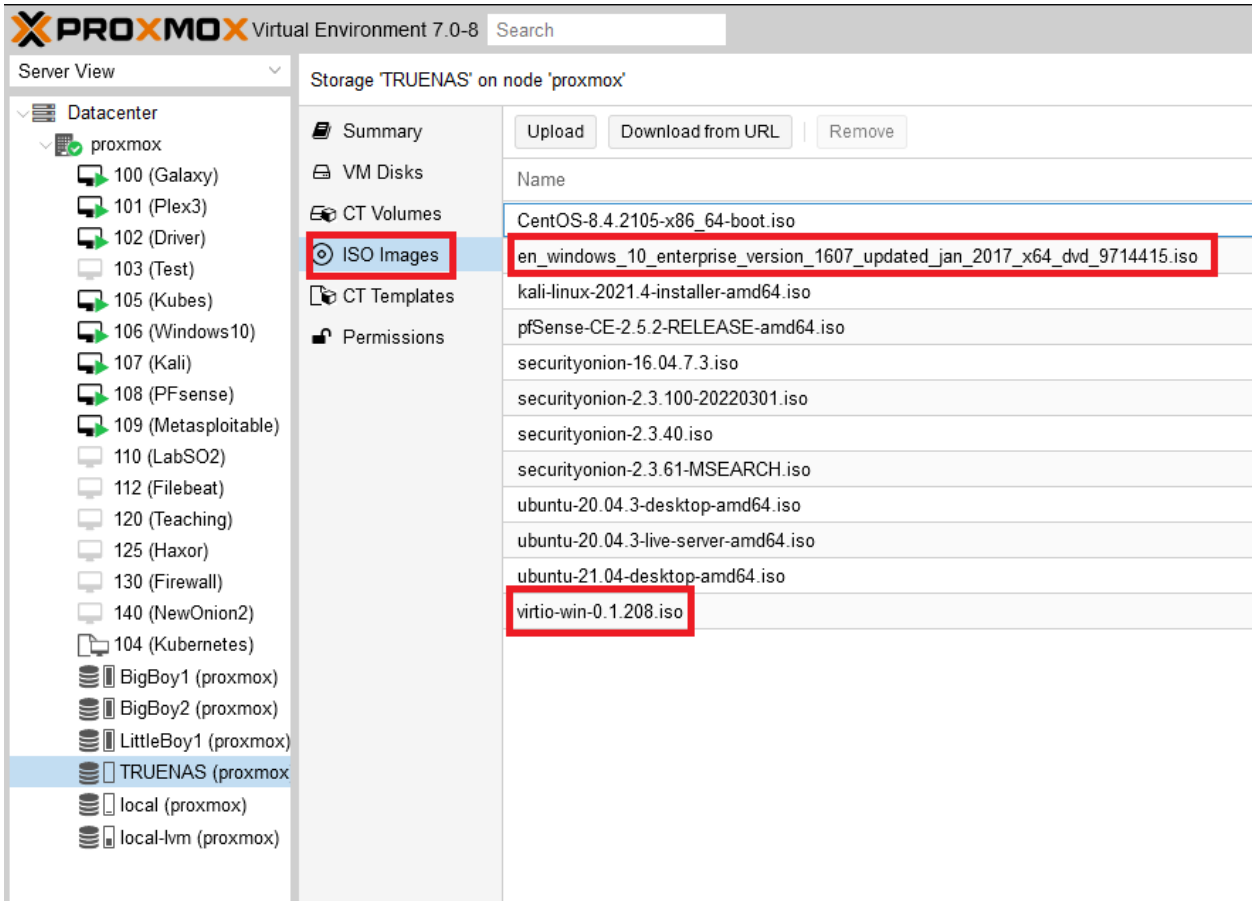
```
*** Welcome to pfSense 2.5.2-RELEASE (amd64) on pfSense ***

 WAN (wan)       -> vtnet0     -> v4: 192.168.2.151/24
 LAN (lan)       -> vtnet1     -> v4: 192.168.3.1/24

 0) Logout (SSH only)                  9) pfTop
 1) Assign Interfaces                 10) Filter Logs
 2) Set interface(s) IP address       11) Restart webConfigurator
 3) Reset webConfigurator password    12) PHP shell + pfSense tools
 4) Reset to factory defaults         13) Update from console
 5) Reboot system                     14) Disable Secure Shell (sshd)
 6) Halt system                       15) Restore recent configuration
 7) Ping host                         16) Restart PHP-FPM
 8) Shell

Enter an option:
```

6. Congrats! That's all there is to configuring PFSense. Once we configure our Windows machine we will be able to access it from the web console at https://192.168.3.1.

3. Win10

Installing and configuring Windows10 is a relatively easy and straightforward process. Keep in mind, you can get a free copy of Windows from the microsoft store based on a version of Windows 10 you already have. I purposely have a slightly out-of-date version of Windows 10 but that's just my preference for pentesting. As far as I'm aware these same steps wouldn't work for Windows 11 but feel free to try and experiment yourself!

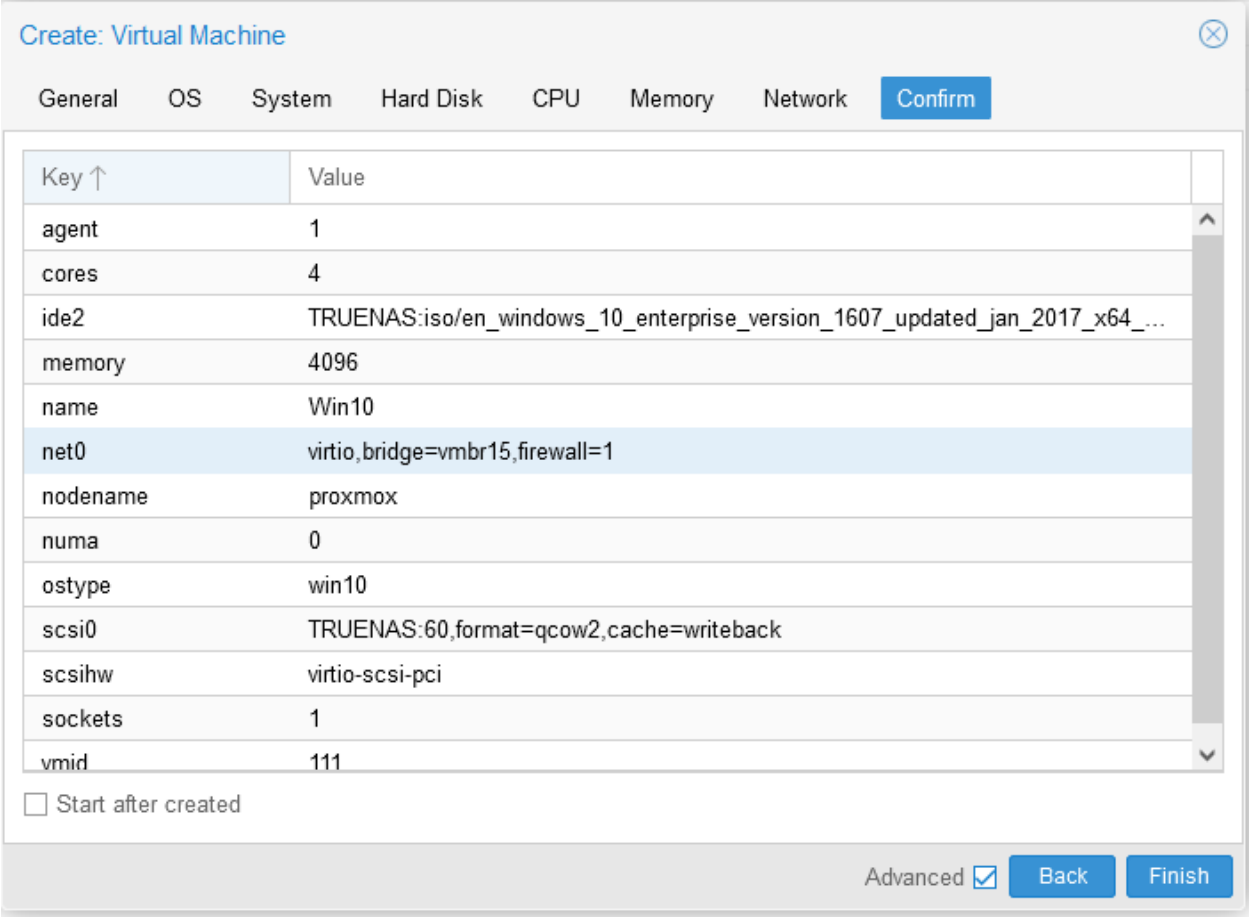
1. The first step will be to upload the iso to your proxmox environment. There are plenty of legitimate ways to get a copy of Windows10. I would recommend cloning your version of Windows 10 using the Windows installation media tool listed here. https://www.microsoft.com/en-us/software-download/windows10
2. Next we will be downloading and uploading Windows VirtIO drivers to our proxmox. This will allow us to install additional drivers for our network card and hard disks for better performance in the future. Download the drivers from the link provided (I select "Latest virtio-win ISO") and upload them just like you would an ISO. https://github.com/virtio-win/virtio-win-pkg-scripts/blob/master/README.md



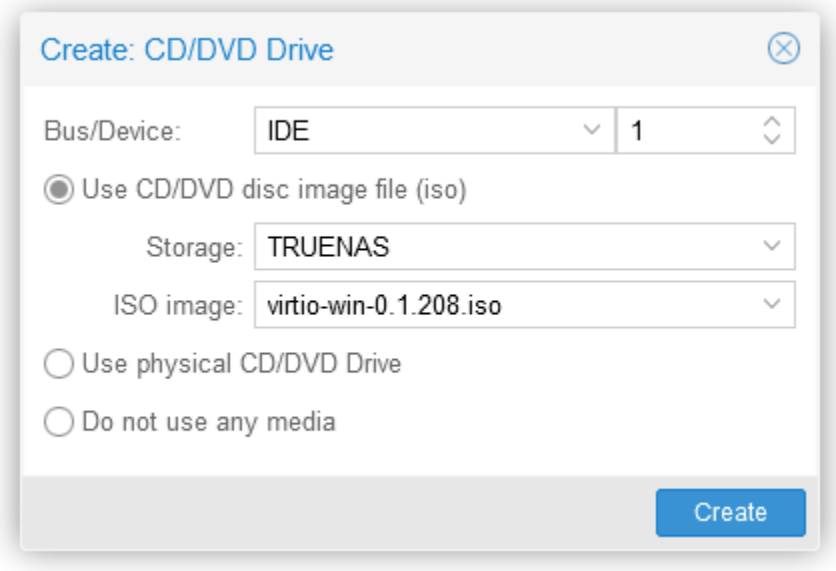
3. After you upload your ISOs into proxmox, we will start configuring our VM. Like other VM's first you will start by right clicking your node and clicking "Create VM". Here are the settings you will want to use for your Windows VM.
    a. General
        i. VM ID: First Available

ii. Name: Your Preference

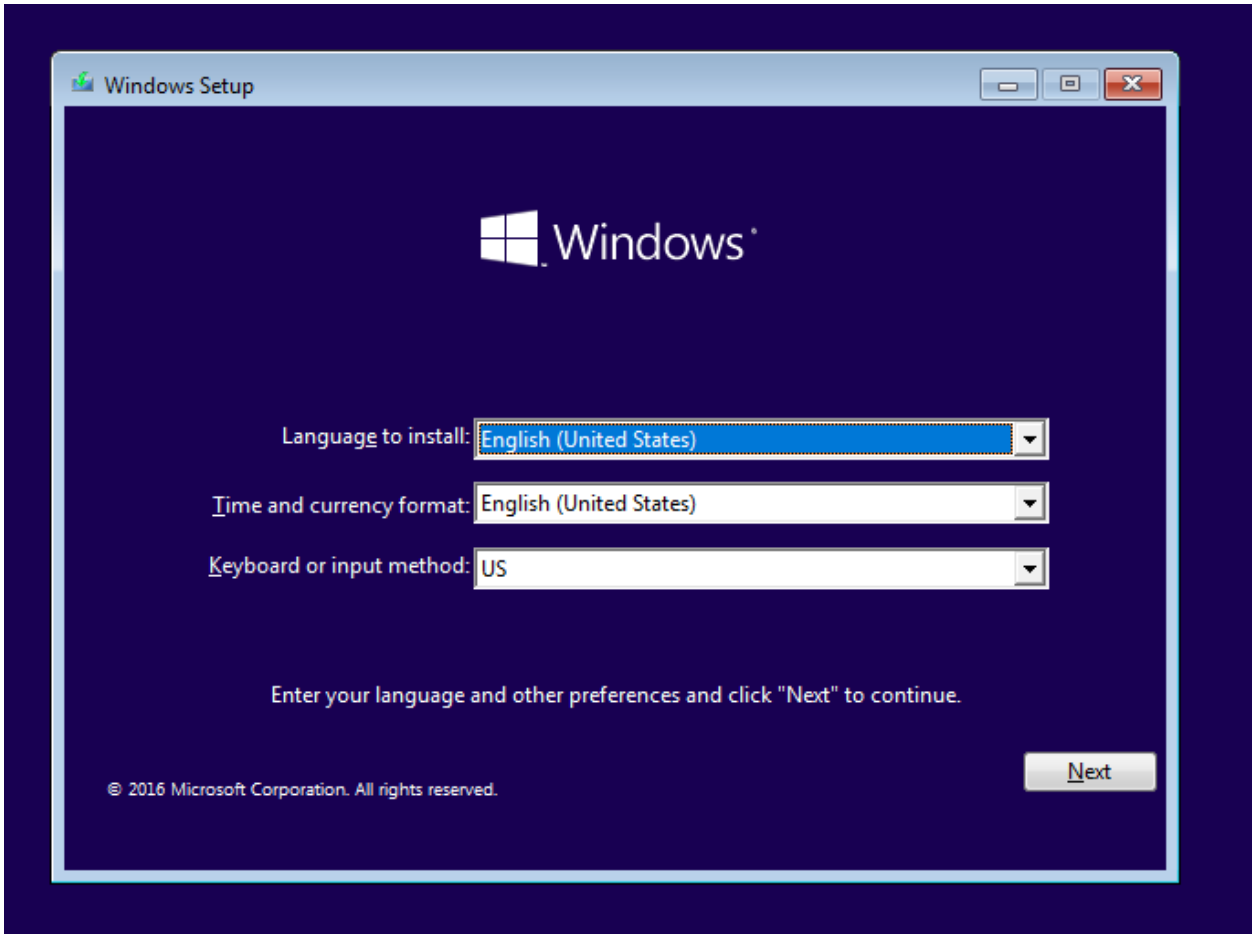
b. OS
   i. Point to your Windows10 .iso
   ii. Select Guest OS type as Microsoft Windows
c. System
   i. Leave default
   ii. Select Qemu agent
d. Hard Disk
   i. Bus/Device: SCSI
   ii. 60GB
   iii. Cache: Write Back
e. CPU
   i. Cores: 4
f. Memory
   i. Memory (MiB) 4096
g. Network
   i. Bridge: vmbr15 (the network bridge with the 192.168.2.100/24 network connection)
   ii. Model: VirtIO
h. Should look similar to below



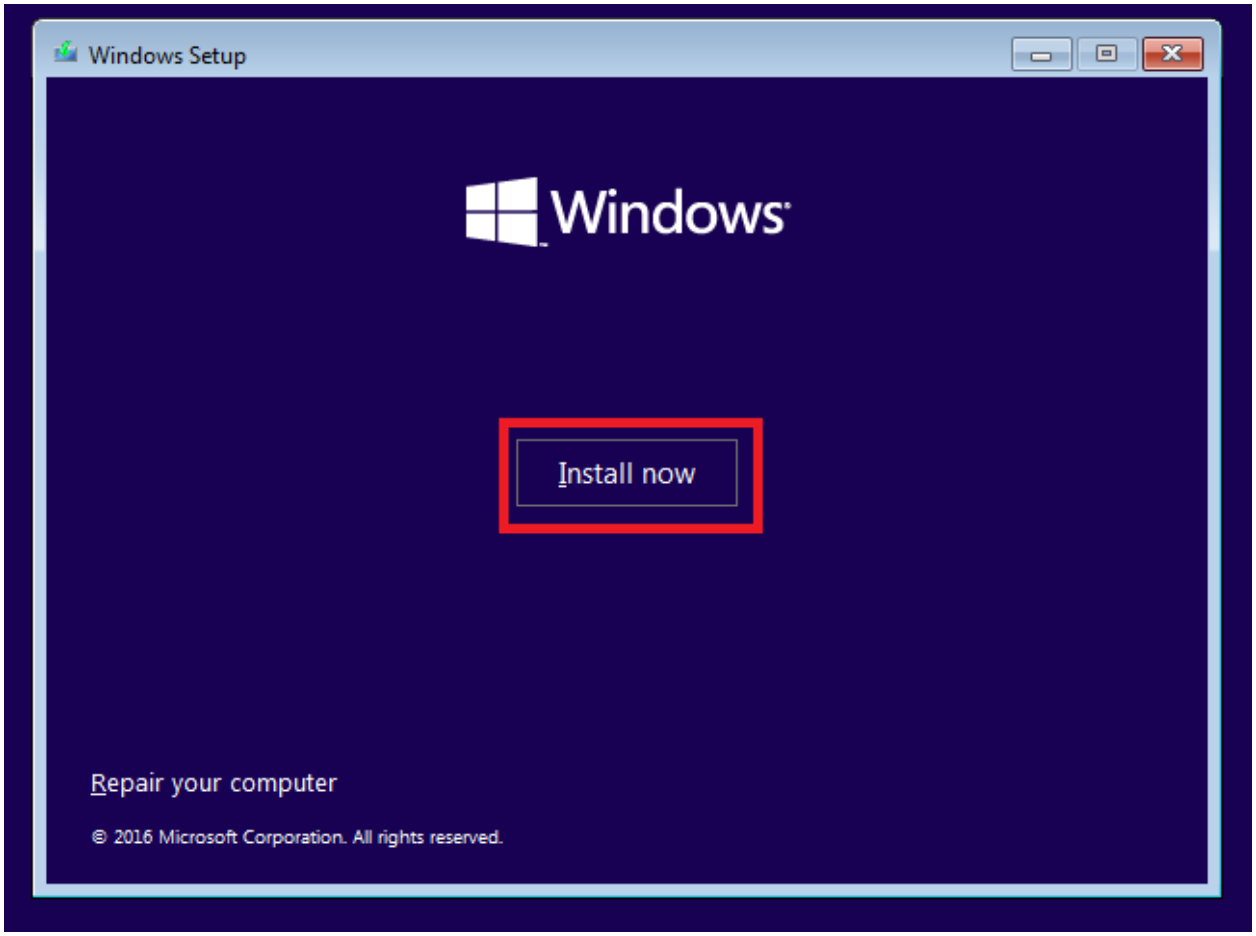
4. Lastly before we boot the VM we will need to add a CD drive with the VirtIO drivers so that we can install them on our system.
    a. Click your newly created VM and then Hardware. Then click Add at the top and select CD/DVD drive. In here you will want to select these settings:
        i. Bus/Device: IDE 1
        ii. Storage: Where your VirtIO ISO is
        iii. ISO image: Your VirtIO ISO



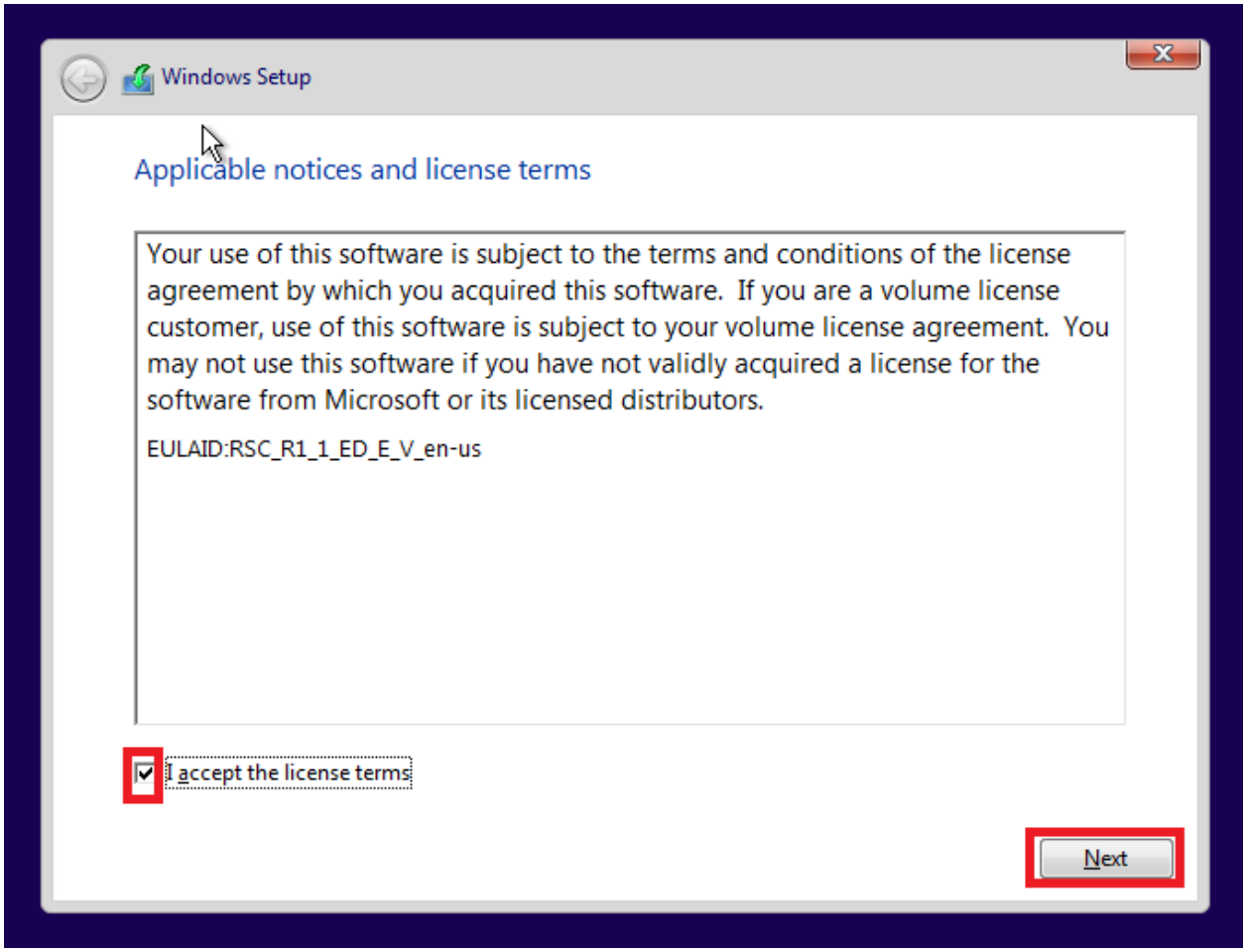
5. Start your VM and now we will begin the installation of Windows 10. This is a pretty straightforward process that we'll walk through and get a properly functioning Windows 10 VM.
    a. Once you power on your VM, you will want to wait until you land on a welcome page prompting you to begin install. Leave the defaults and press Next.



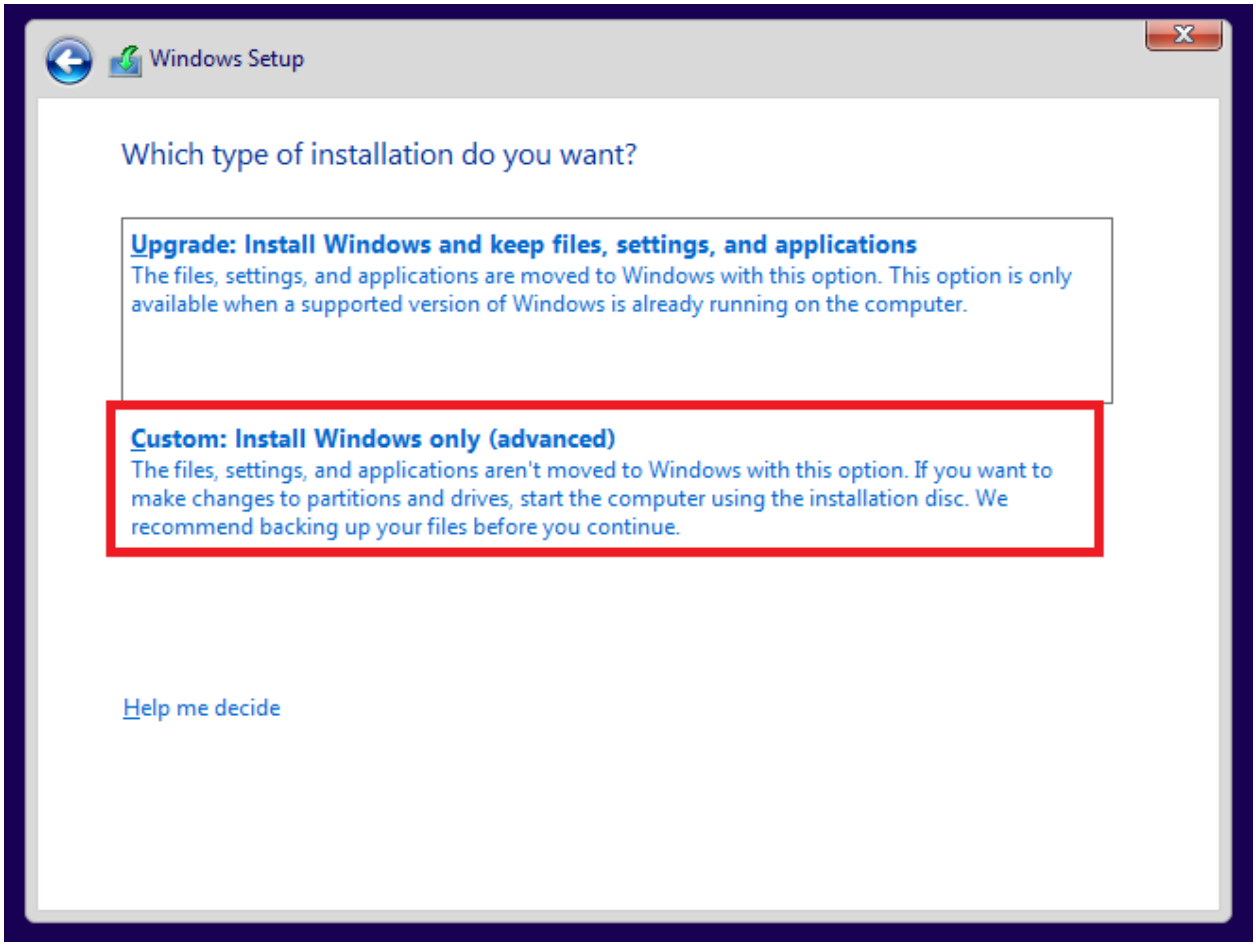
b. Next you will click Install Now to start the installation.



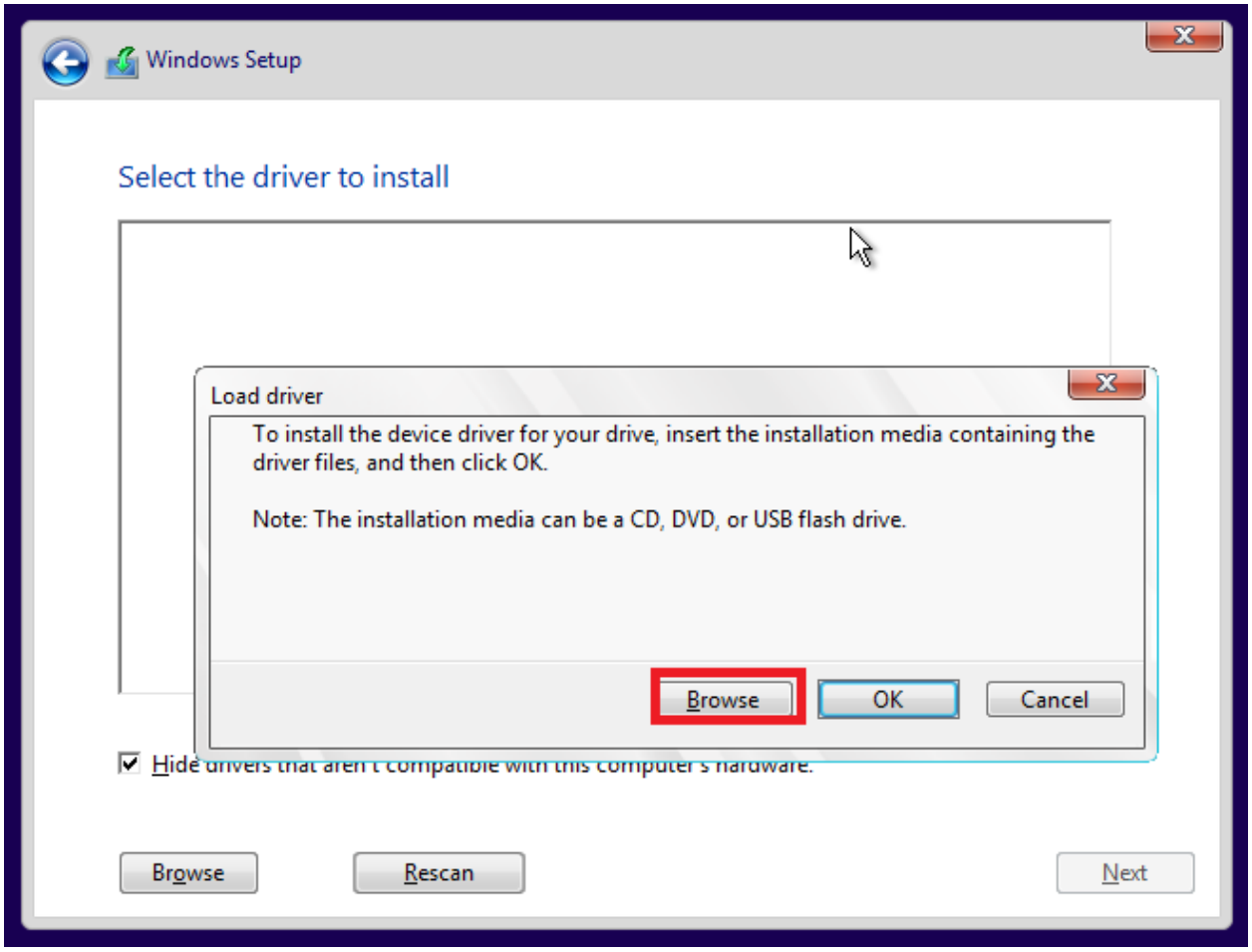
c. After a brief minute, a EULA will appear. Click the checkbox to agree to the terms and press Next to continue.



d. Next you will be prompted what type of installation you want to do. Select Custom.

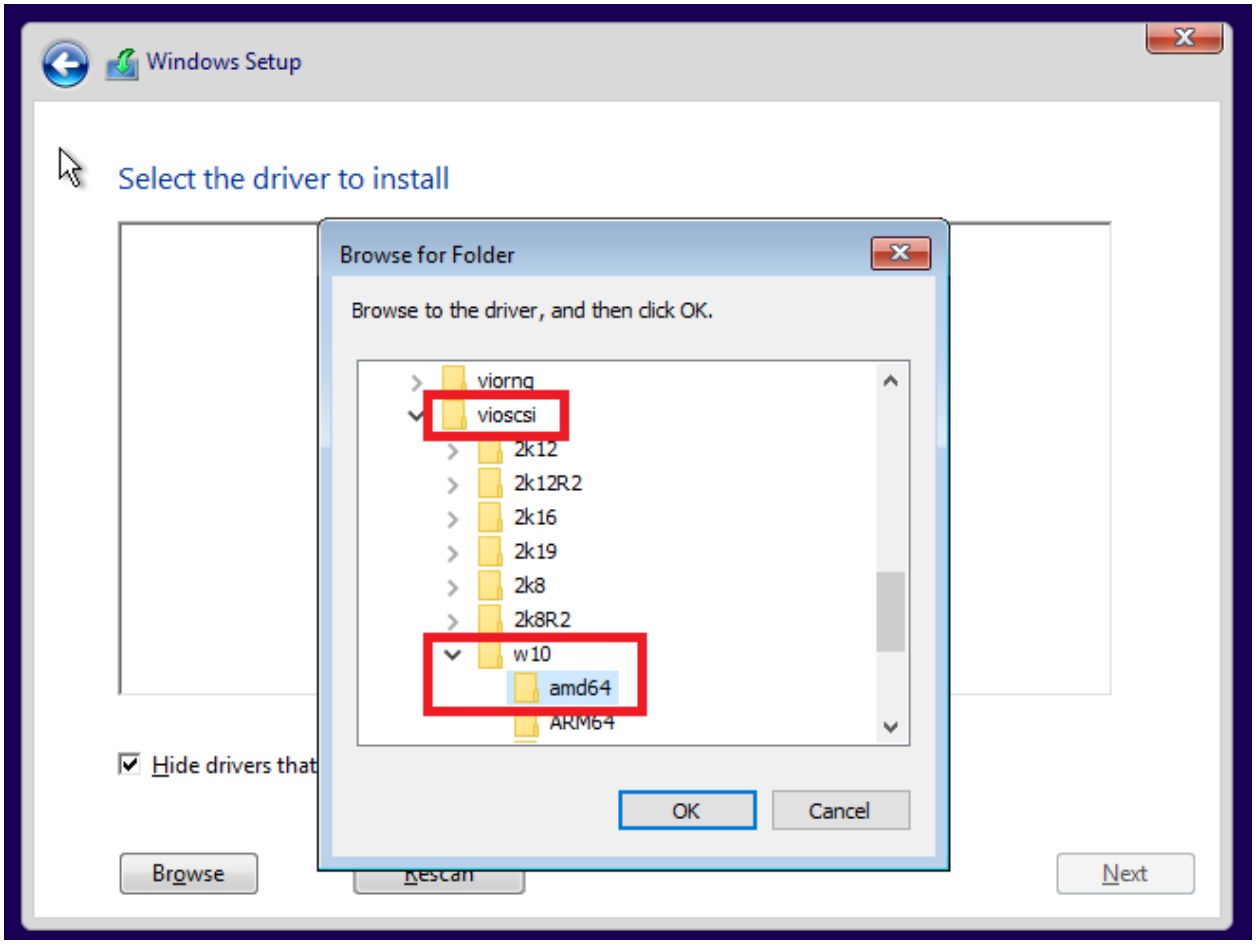


e. Next you will be prompted to install the OS on a drive. The problem is, we don't have a disk to select. This is where the VirtIO drivers come into play. Click Load Driver and then click Browse and navigate to the VirtIO driver.

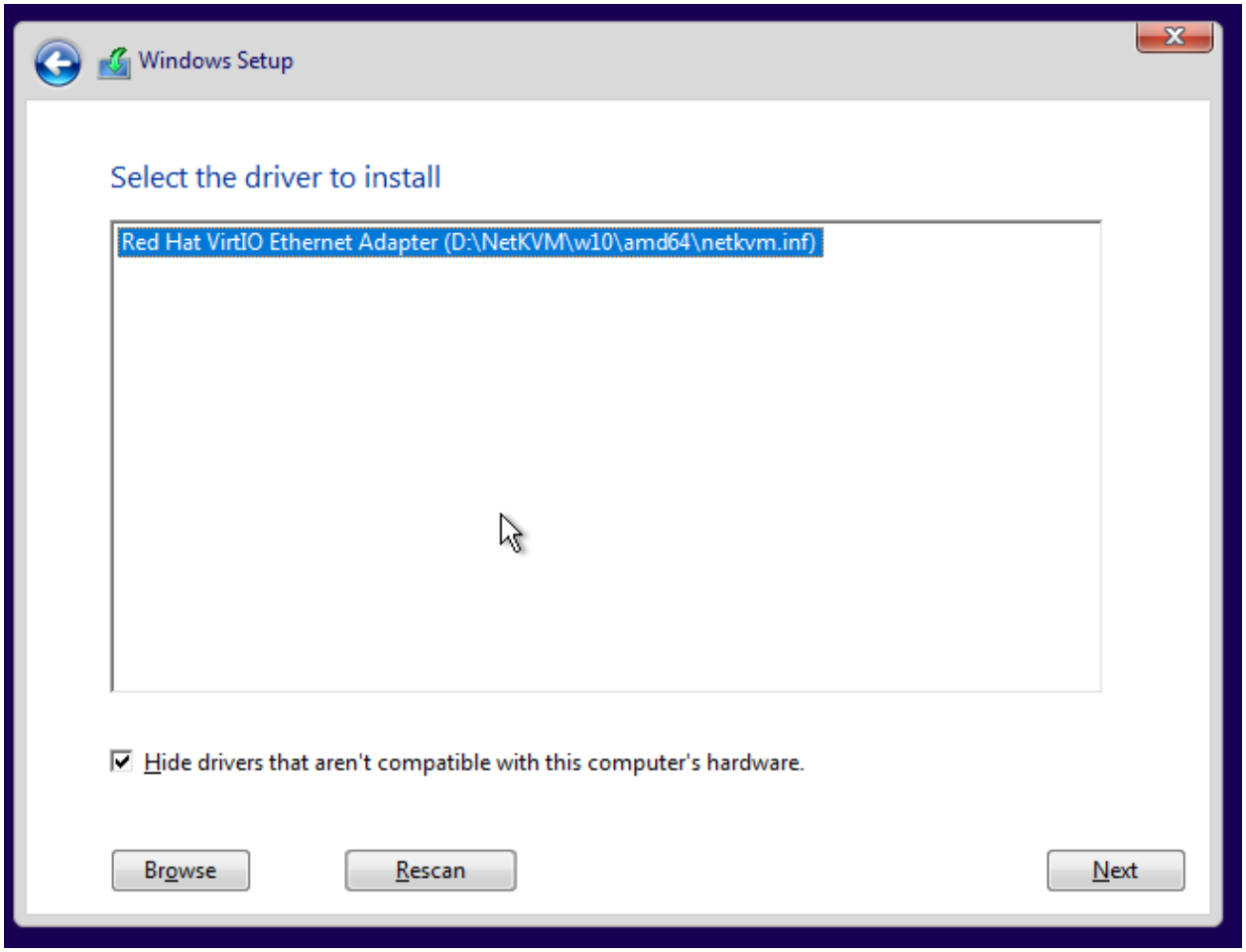


f. You should then see a CD Drive (mine is the D: drive) where our VirtIO drivers are. Expand that drive and go to the following file. D:\vioscsi\w 10\amd64 and press OK. Once the driver is loaded press next to install that driver.

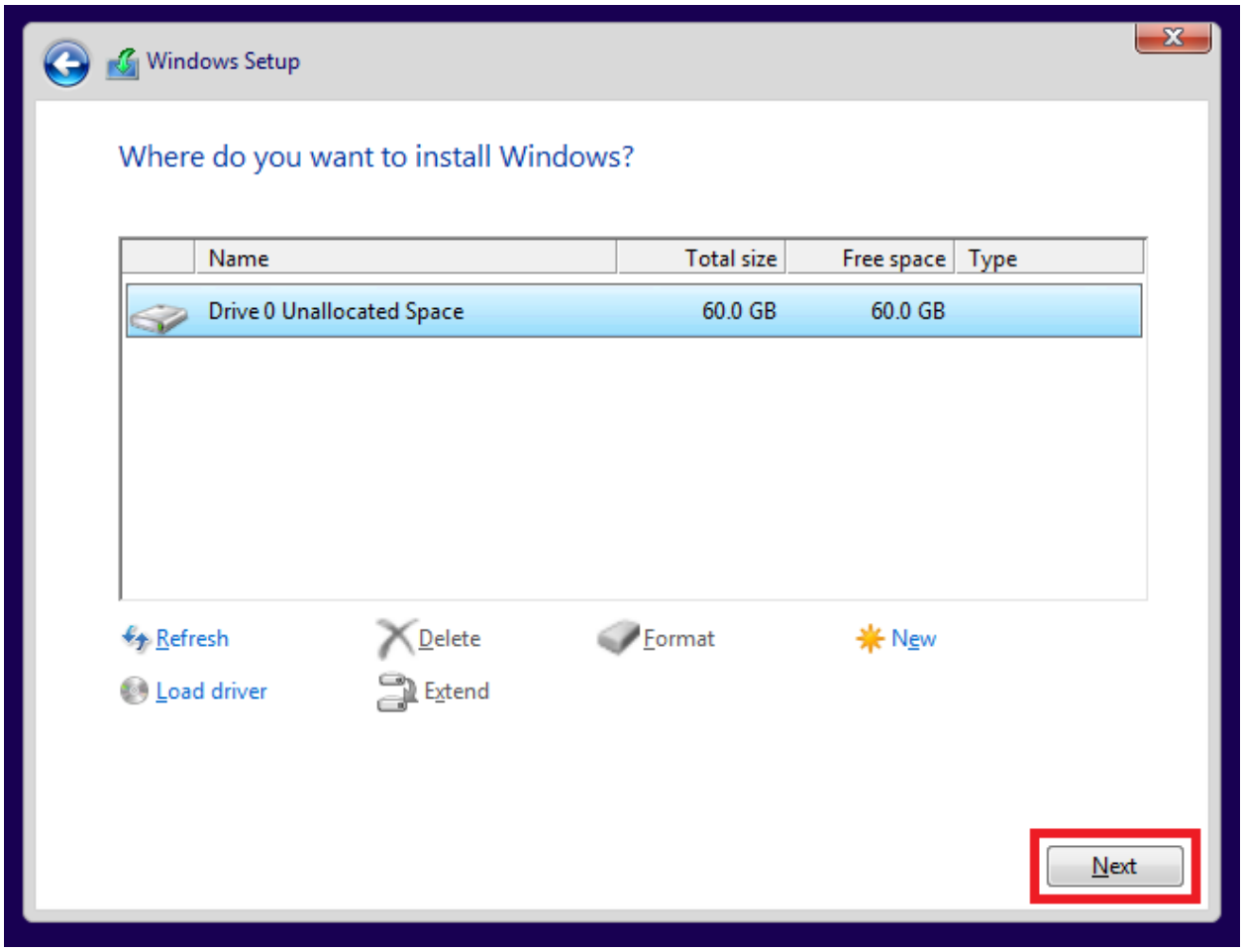
   i. **Note** If you're using an Intel CPU select x86 instead



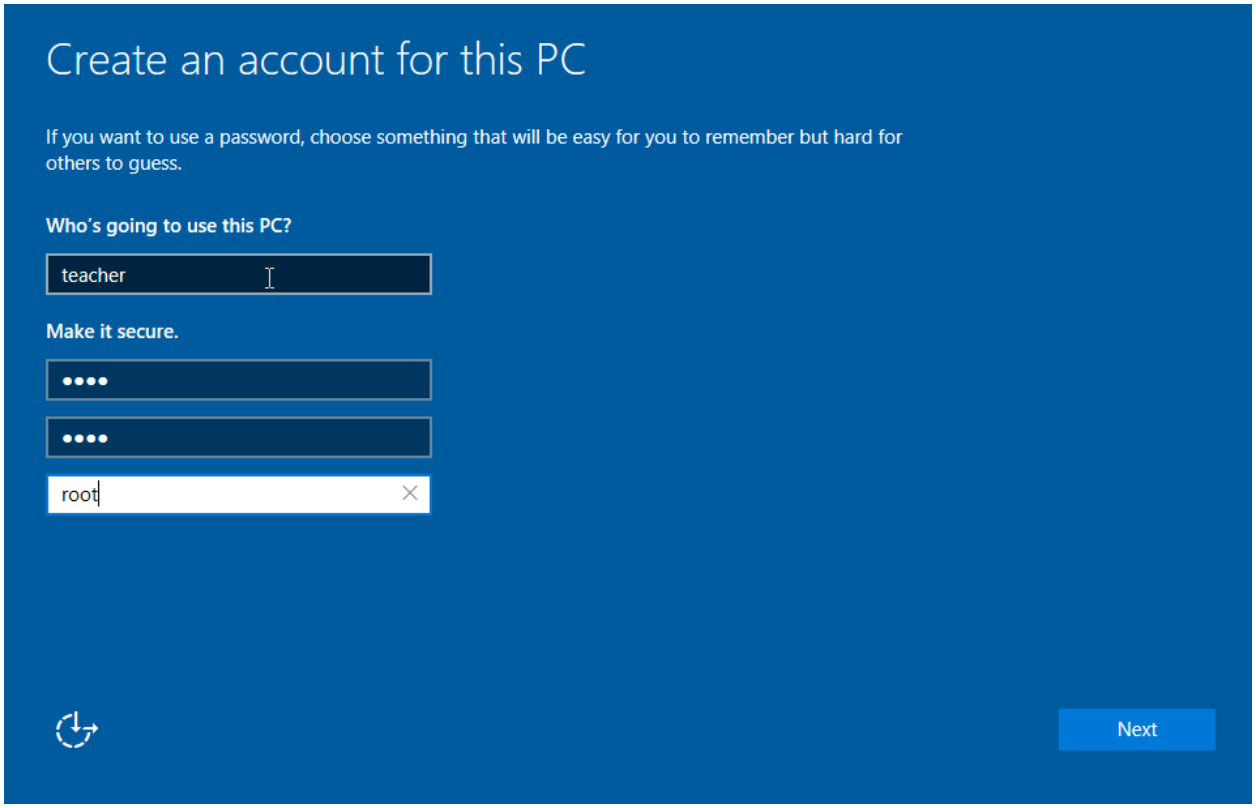
g. Once that driver install is complete you should now see your 60GB drive. We should now take the time to install our network driver for the best experience. Repeat the same steps as before and make sure you install the following driver:
   i. D:\netkvm\w 10\amd64 (This handles our network adapter)



h. Now that we have our drivers installed, we can click next to begin the actual install. This process usually takes around 15 minutes.

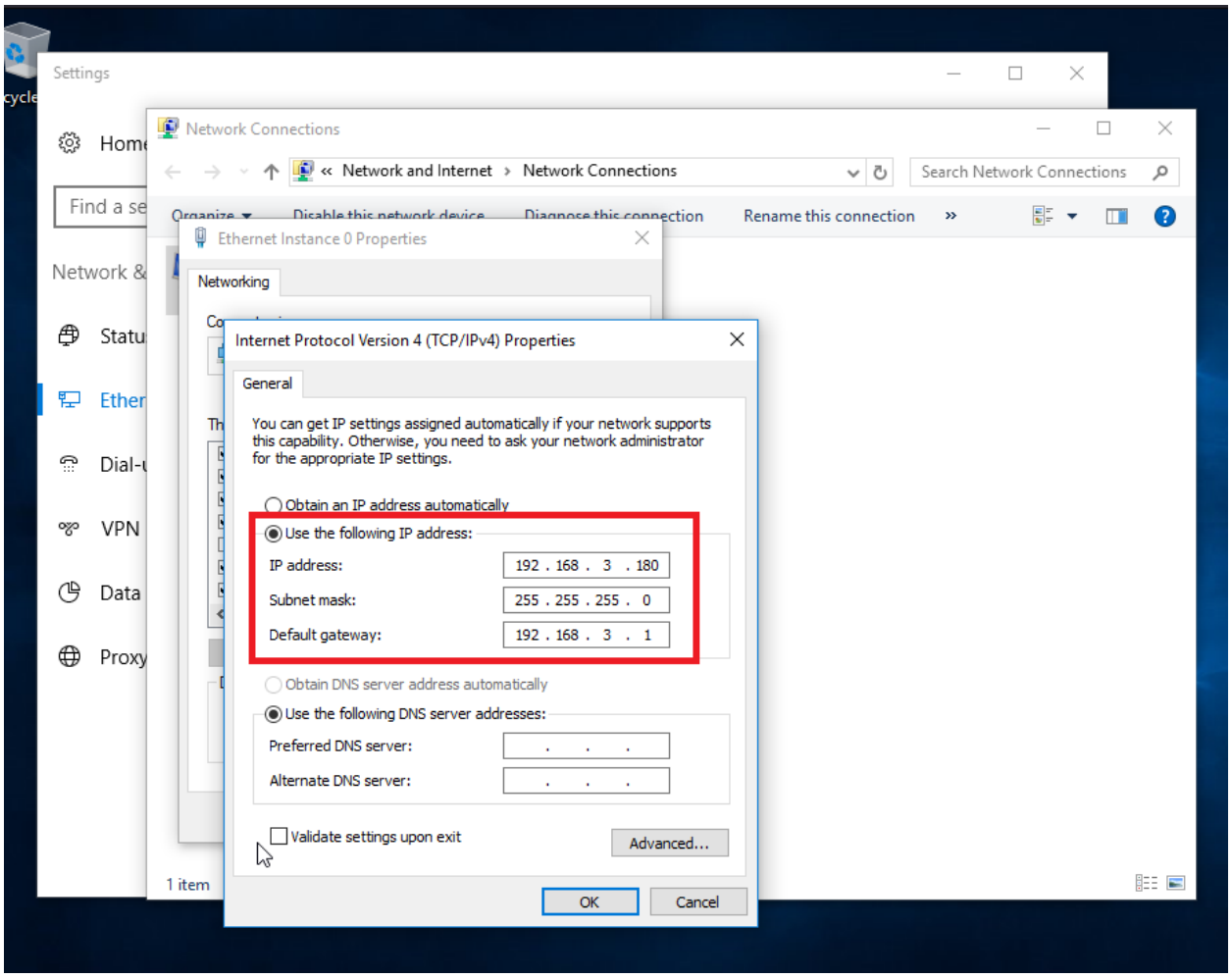


i. Once it finishes your VM will automatically restart and you will be able to start configuring your Windows VM to your preferences. I like to click express settings to have the install go as fast as possible and the Windows monitoring settings don't matter as it will not be connected to the internet. Additionally make sure you give yourself a username and password that's easy for you to remember.

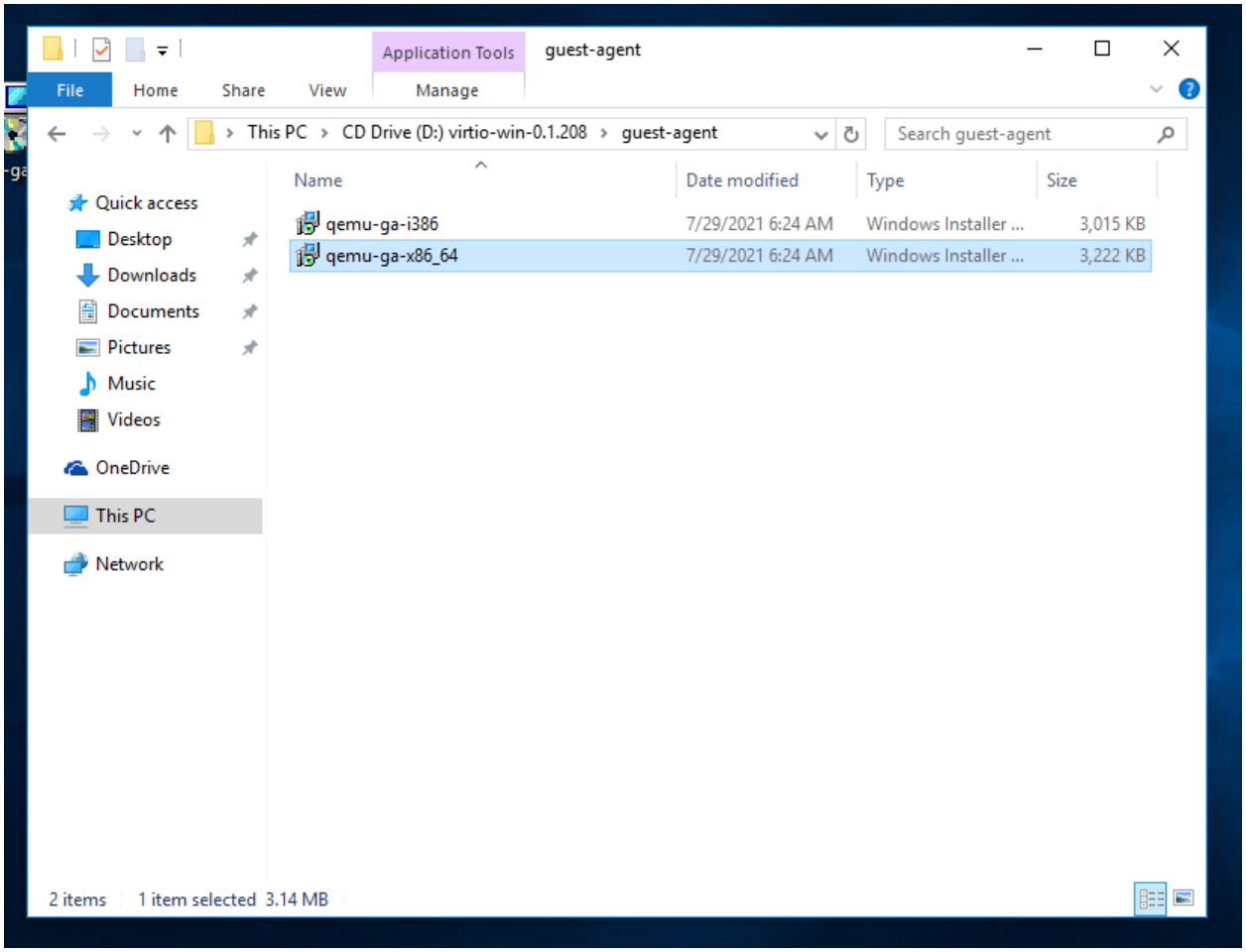


j. You may be prompted to setup Cortana or other Microsoft services here. Again it's your choice but I choose not to. You will then get to a point where it will install again. Just wait for Windows to do it's thing and then you will be brought to the main Windows page.

k. From here the last thing I recommend is going into your network adapter settings and manually configuring an IP address for your range. I'm setting mine based off my network diagram 192.168.3.180.



I. That last thing we will want to do is install the qemu-guest agent for best performance. To do so open up file explorer and navigate to to your D: drive and select guest-agent. From here launch the x86 guest agent.



m. Then you will want to restart your VM which will apply the guest agent configuration.
n. Congrats! As far as this tutorial you're done configuring your windows machine. If you have your PFsense firewall running you should now be able to ping and access the web interface. Some additional steps you may want to take depending on your use case could be disabling/tweaking the Windows firewall, creating more user accounts etc.

4. Metasploitable

Metasploitable is the hardest of the OS's to install and configure due to lack of support of .vmdk files inside proxmox. To get around this we will need to transfer the file to proxmox, convert the file into a qcow2 image and then launch the VM. Here are the instructions to do so.

1. The first step would be to download the latest version of metasploitable if you haven't already done so. The link can be found here, just fill out your information and get the download. https://sourceforge.net/projects/metasploitable/
2. After your download has completed, unzip and extract the contents of the files to your desktop. It should look similar to this

Name	Date modified	Type	Size
Metasploitable.nvram	12/16/2021 9:36 PM	VMware Virtual M...	9 KB
Metasploitable.vmdk	12/16/2021 9:36 PM	VMDK File	1,880,512 KB
Metasploitable.vmsd	12/16/2021 9:36 PM	VMware snapshot ...	0 KB
Metasploitable.vmx	12/16/2021 9:36 PM	VMware virtual m...	3 KB
Metasploitable.vmxf	12/16/2021 9:36 PM	VMware Team Me...	1 KB

3. You will want to transfer the files to the /tmp directory on your proxmox server. Here I used WinSCP from my host machine to login to my proxmox server and transfer the files.

/tmp/

Name	S...	Changed	Rights	Owner
..		9/12/2021 4:24:07 PM	rwxr-xr-x	root
systemd-private-d0e9171c...		12/15/2021 10:26:47 PM	rwx------	root
systemd-private-d0e9171c...		12/15/2021 10:26:46 PM	rwx------	root
Metasploitable.nvram	9...	12/14/2021 1:08:35 PM	rw-r--r--	root
Metasploitable.vmdk	1...	12/14/2021 1:08:48 PM	rw-r--r--	root
Metasploitable.vmsd	0...	12/14/2021 1:08:48 PM	rw-r--r--	root
Metasploitable.vmx	3...	12/14/2021 1:08:48 PM	rw-r--r--	root
Metasploitable.vmxf	1...	12/14/2021 1:08:48 PM	rw-r--r--	root
sploit.qcow2	1...	12/16/2021 10:34:58 PM	rw-r--r--	root

4. From here you will want to either SSH or putty your way into your proxmox server to start converting the file into a qcow2. **If you're wondering why we have to do this, it's because Proxmox is built on a redhat platform which uses qcow instead of a traditional .vmdk (VMware) or .ova (virtualbox) platform.**

```
10.55.0.100 - PuTTY
root@proxmox:/tmp# pwd
/tmp
root@proxmox:/tmp# ls
Metasploitable.nvram  Metasploitable.vmxf
Metasploitable.vmdk   sploit.qcow2
Metasploitable.vmsd   systemd-private-d0e9171cdcad40c9b554e2dbda8936b4-chrony.service-PEulcj
Metasploitable.vmx    systemd-private-d0e9171cdcad40c9b554e2dbda8936b4-systemd-logind.service-BUkBui
root@proxmox:/tmp#
```

5. From here you will want to go to the proxmox web interface and create a new VM by clicking “create VM” in the top left. From here select the following options on each screen.
    a. Name the VM something appropriate and take note of the VM ID, in this case I’ll be using 120.
    b. In the OS tab, click “do not use any media” and make sure Linux is the selected guest OS.
    c. Under system, you can leave everything default and click next.
    d. Under Hard Disk make sure you select the appropriate Storage location for your environment (where you want to store the data) in my case that will be local-lvm and I’ll leave the default disk size at 32GiB.
    e. For CPU increase the Cores count to 2.
    f. For Memory you can leave the default at 2048 (2GB) or reduce it to 1024 (1GB) if you’re worried about resources.
    g. For network you will want to select vmbr15 if you’re following along with my guide.
    h. Hit finish to create the VM.
6. From here you will want to go back into your proxmox. What we will be doing is converting the metasploitable vmdk file into a qcow2 and then changing the configuration of our VM inside proxmox.
    a. First you will create a new directory, and you will base it off your vm id. Type mkdir /var/lib/vz/images/120

```
root@proxmox:/var/lib/vz/images# mkdir /var/lib/vz/images/120
```

    b. Go to your /tmp directory (or wherever the metasploitable.vmdk file is) and type the following command to make it a qcow2. Type qemu-img convert -f vmdk Metasploitable.vmdk -O qcow2 Meta.qcow2

```
root@proxmox:/tmp# qemu-img convert -f vmdk Metasploitable.vmdk -O qcow2 Meta.qcow2
```

    c. Now move your new qcow2 vm into your newly created directory. Type mv Meta.qcow2 /var/lib/vz/images/120

```
root@proxmox:/tmp# mv Meta.qcow2 /var/lib/vz/images/120
```

    d. Once the file is in there you will want to edit your new metasploitable VM config file. Use nano to edit the attributes to point your first boot disk (ide0 or SCSI) to your local config file. In my example, Scsi0 is my first bootable drive and I pointed it to my local file where I stored the new qcow2 image.

```
boot: order=scsi0;ide2;net0
cores: 2
ide2: none,media=cdrom
memory: 2048
name: metatest
net0: virtio=E2:3B:36:3B:73:A9,bridge=vmbr15,firewall=1
numa: 0
ostype: l26
scsi0: file=local:120/Meta.qcow2,size=32G
scsihw: virtio-scsi-pci
smbios1: uuid=7837e668-fa6c-4d4b-85b0-949b3e62c43e
sockets: 1
vmgenid: 4f159f4c-d5c9-4f69-9ae3-ee33be7b8be0
```

e. Lastly you will need to change the properties of local files so that they can boot qcow 2 images off them. To do this you will edit the /etc/pve/storage.cfg file and add images under “local” like so.

```
dir: local
        path /var/lib/vz
        content backup,iso,vztmpl,images
```

.

7. All you need to do now is power on the VM and it should begin the installation of Metasploitable2. Once it finishes it will reboot, you will then login and give the box an IP address. The login credentials will always be msfadmin:msfadmin.

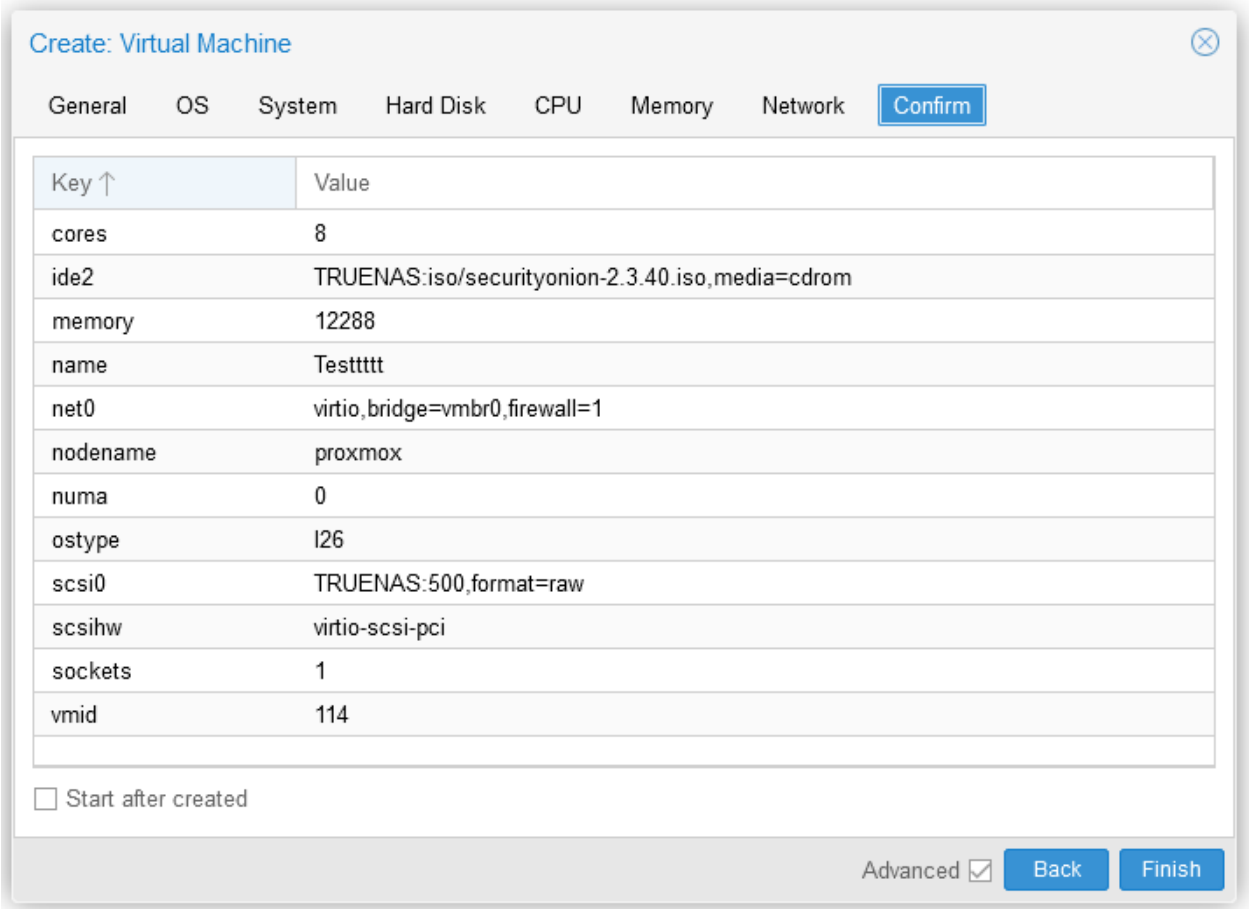
```
msfadmin@metasploitable:/etc/network$ sudo ifconfig eth0 192.168.3.56 netmask 25
5.255.255.0
```

8. Awesome! You now have an easily exploitable VM that you can test exploits with!

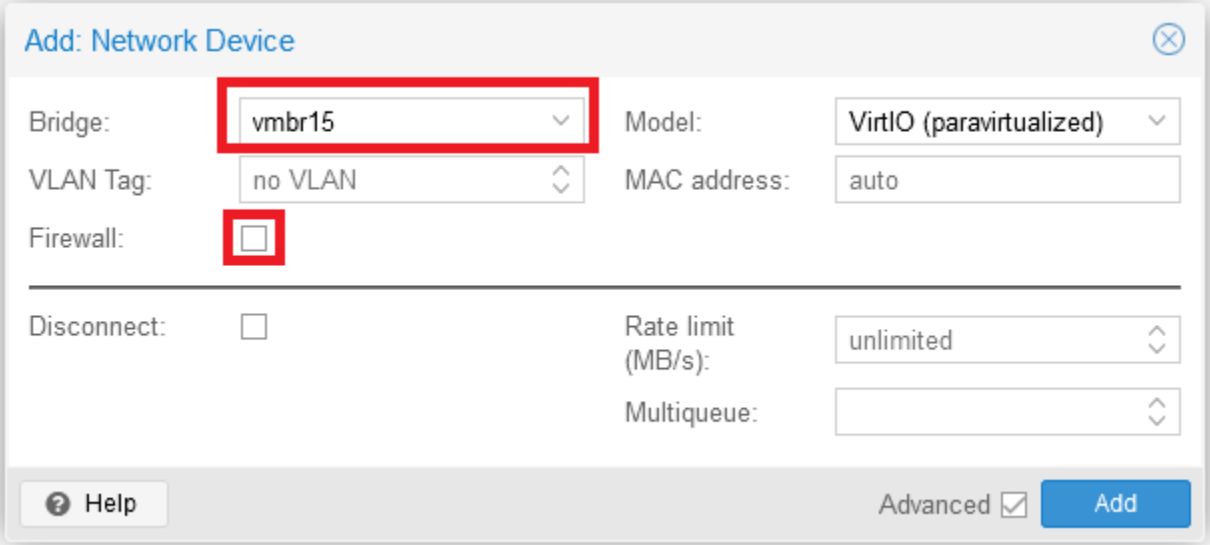
5. SecOnionv2

SecurityOnion2 is in my opinion the most important tool in this homelab and the reason I went out and built this lab and wrote this guide. This portion will walk you through the steps of setting up and configuring a standalone SecurityOnion2 deployment as well as creating and allowing ourselves access to the SOC. Lastly we will configure mirroring of our vmbr15 network and pass that traffic into our SecurityOnion2 deployment so that we will have live network traffic. This portion isn't intended to explain the ins-and-outs of SecurityOnion2 and as such I won't be going over every choice I make. If you're interested in learning more please reach out to me about some of my other trainings that specifically go into detail on the entire setup process and my reasoning for choices I make.

1. The first thing you will want to do is go out and grab a SecurityOnion2 ISO from the internet and import it into your proxmox environment. For this guide I will be using SecOnionv2.3.40 so there may be slight differences in the installation steps. You can download SecurityOnion2 ISOs from this link. https://github.com/Security-Onion-Solutions/securityonion/blob/master/VERIFY_ISO.md
2. Once you have the ISO uploaded you will then create a new VM and use the following settings.
    a. General
        i. VM ID: First Available
        ii. Name: Your Preference
    b. OS
        i. Point to your SecOnion2 ISO
    c. System
        i. Leave default
    d. Hard Disk
        i. 500GB
    e. CPU
        i. Cores: 8
    f. Memory
        i. Memory (MiB) 12288
    g. Network
        i. Bridge: vmbr0 (you want to be able to access your SecurityOnion from outside your network)
    h. Should look similar to below



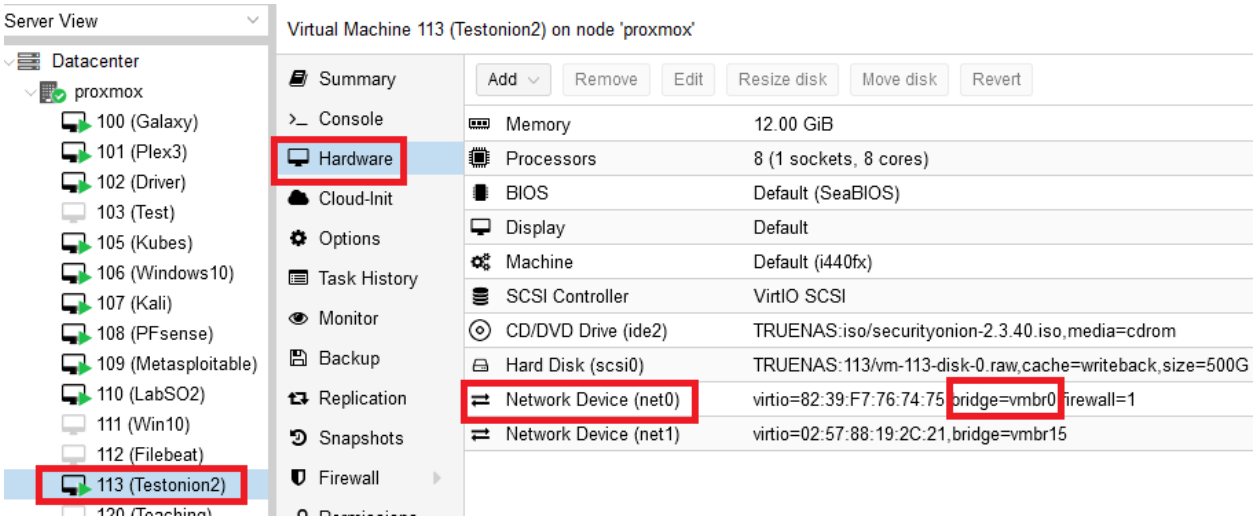
3. Before powering on your VM, you will want to go to the VM's hardware settings and add another network interface. In this case you will be adding vmbr15 if you're following my guide. Make sure when adding the network interface that you disable the Firewall.



4. Now we're ready to power on and install the SecurityOnion2 OS onto our virtual disk. To do this you will just follow the steps listed below.

a. First wait for the install to begin in basic graphics mode.
b. Then you will be prompted to type yes to continue.
c. Then you will be prompted to enter an administrative username. This will be your username you will use to login to the console in the future.
d. Then you will be prompted to enter and confirm a password. Make sure to not use special characters or spaces, there is currently a bug in the installation that will lock you out of logging in if you use those in your password.
e. Next you will wait for SecurityOnion2 to automatically start the install. From here it should take around 10 minutes for the OS to be installed. You will have to press enter after the install is finished to reboot the VM. Now you have installed the underlying OS that SecurityOnion2 resides on. Next we will go into configuring our deployment of SecurityOnion2.

5. Once the VM reboots and you login you will be greeted by a screen prompting you to begin setup. Before we begin, just a couple of quick notes. We are going to be configuring a standalone deployment. That means that this instance of SecurityOnion2 will handle both the data ingestion as well as storing and retrieving that data for you. In addition we will configure this SecurityOnion2 instance to be accessible on your home net but won't call out to the internet. That means that if you have another computer you want to access the SOC from (web GUI) you will be able to do so. Once we configure this standalone we will move into the last phase which is ingesting the data.
    a. First press enter to confirm we would like to begin setup.
    b. Press enter to confirm we want to run the standard installation.
    c. Using your arrow keys, press down and then press spacebar to select STANDALONE deployment and press enter.
    d. Using your arrow keys, press down and then press spacebar to select AIRGAP so that our instance doesn't have access to the internet.
        i. This only prevents SecurityOnion2 from pulling updates over the wire. You can do a standard installation if you would prefer that.
    e. Type AGREE and press enter to agree to the Elastic Stack EULA.
    f. Enter a hostname for your deployment. I recommend something easy like homeonion. Press enter to confirm your hostname.
    g. You will then be prompted to select your management NIC. Your management NIC corresponds with the network you want to access Kibana, SOC etc. In my case that's my personal home network (vmbr0). If I look in my hardware settings inside Proxmox, vmbr0 corresponds with net0 aka eth0. Double check your settings and press spacebar over your selection and enter to continue.



h. You will then be prompted if you want to assign a static or DHCP address. For future reference, you will always be assigning an IP address based off a static address. Press enter to confirm STATIC.
    i. We always choose a static address because if our managers IP address were to ever change, it would break any connection to any other node in a distributed system. While this specific deployment isn't a distributed one, it's good habit to always be choosing static.

i. Type in an IP address you want to give to your SecurityOnion2 VM. This needs to be in the same subnet of your home IP range. In my case, my home address is in the 192.168.1.0/24 address space so I will be assigning a random address in that range. Type in your IP address with CIDR notation and press enter to confirm. (I did 192.168.1.99/24)

j. Type in your home networks gateway address. If you don't know it open up a command line interface on machine in your network, type ipconfig and note the Default Gateway. Mine is 192.168.1.254 so I will input that and press enter.

k. You will then be prompted to input your DNS server. Because this is an air gapped system this doesn't matter so you can press enter to leave the default.

l. You will then be prompted to enter your DNS search domain. Like the previous page this doesn't matter, HOWEVER; I have had problems leaving this in the default configuration. So instead I type in cpb.com and press enter.

m. You will then be prompted to press OK to allow SecurityOnion2 to initialize the networking. Press enter to continue.

n. After it's finished, you will then be asked what interface you want to use to monitor network traffic. Only 1 NIC should be available to select. Again this NIC corresponds with our "customer network" and will require further configuration which we will go over later. Press spacebar to select the interface and press enter to confirm.

o. Next you will be prompted to define your $HOME_NET variable. This is important as it helps define rules in Snort. Use the same IP schema that you will use for your "customer network". In my case my "customer network" has an IP range of 192.168.3.0/24 so that is what I'll be typing and pressing enter to confirm.

p. You will then be prompted how advanced would you like to make your configuration. For this instance a BASIC installation is suitable. Press enter to confirm BASIC.
q. You will be prompted which tool you would like to generate metadata. I prefer using Zeek and will press enter to confirm that choice.
r. You will be prompted to select an IDS ruleset. Unless you have a paid token the only valid choice is ETOPEN. Press enter to confirm ETOPEN.
s. Press enter to acknowledge the statement about services.
t. This next page will prompt you to select any services you wish to enable for this deployment. Like the previous textbox indicated, the more you enable the more resource intensive your deployment will be. I recommend enabling the following services:
    i. GRAFANA
    ii. OSQUERY
    iii. PLAYBOOK
    iv. STRELKA
u. Press enter to confirm the default docker IP range.
v. Enter an email address to use for logging into the SOC and Kibana. This doesn't necessarily have to be an actual email and thus I use mitchell@cpb.com
w. Type and confirm a password for this account and remember the aforementioned bug when it comes to security complexity.
x. Press enter to confirm you want to access the SOC using an IP address.
y. Press enter and type and confirm a password for the soremote user.
    i. We won't be using this in a standalone deployment but it is important for distributed environments.
z. Again, a BASIC deployment is acceptable for this deployment type. Press enter to confirm BASIC deployment setup.
aa. Press enter to leave the default zeek processor at 1.
bb. Press enter to leave the default suricata processor at 1.
cc. Press enter to confirm a NODEBASIC config install.
dd. Press enter so that we can open up the firewall for proper access.
ee. You will want to put the IP address of the machine that you will primarily be accessing SecurityOnion2 over port 443. Alternatively you can use your whole home network range so every computer can access it. I chose the latter and typed 192.168.1.0/24.
ff. Press enter to confirm the STANDALONE deployment type.
gg. You will then need to wait for SecurityOnion2 to apply your configuration changes. This can roughly take ~30 minutes to up to an hour +. Be patient as usually it will throw you an error message if something does fail during install.
    i. If you do experience an error during install that's unfortunately common. I would personally recommend deleting the VM from proxmox and starting back from the first step and configuring it the exact same way. If you're still experiencing failures try to change some settings or use a newer version of SecurityOnion2.

6. After SecurityOnion2 finishes installing and you're able to successfully login to the SOC you will now want to configure a virtual SPAN port so that network traffic that's generated inside your "customer network" is sent to your SecurityOnion2 VM. This will all be done inside a Putty/SSH connection to your Proxmox. You will also want to leave your SecurityOnion2 VM running.
    a. Putty/SSH into your Proxmox OS (will be the same IP as your web GUI).
    b. You will then need to download some packages for configuration.
        i. Type: apt install openvswitch-switch ethtool

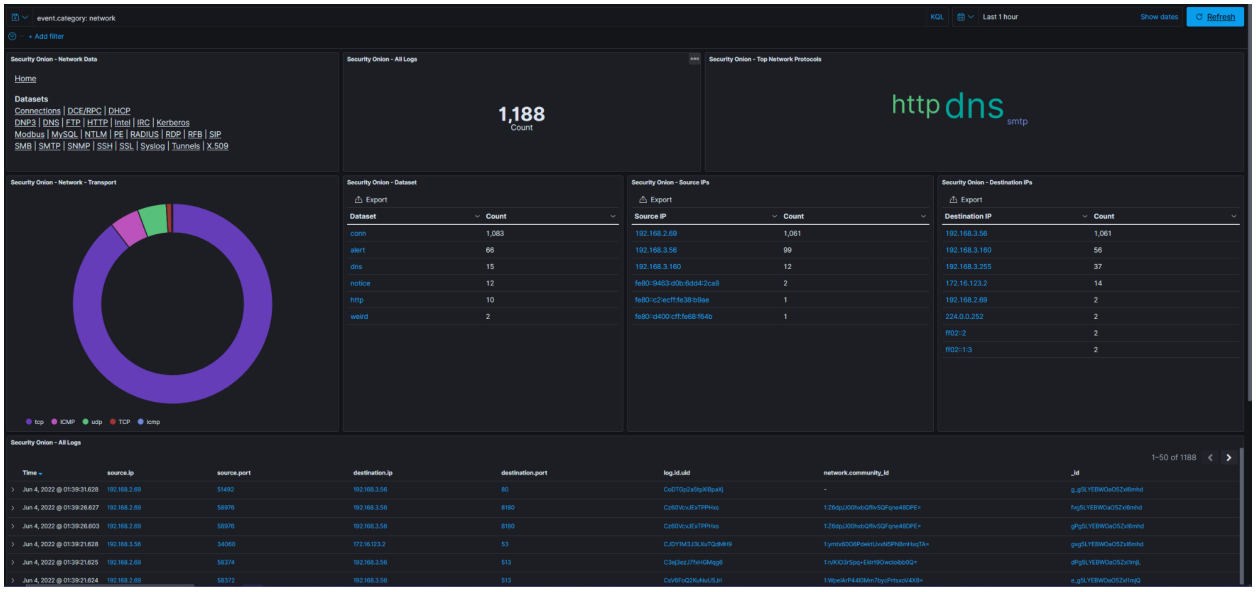
```
root@proxmox:~# apt install openvswitch-switch ethtool
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ethtool is already the newest version (1:5.9-1).
openvswitch-switch is already the newest version (2.15.0+ds1-2+deb11u1).
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
root@proxmox:~#
```

    c. After the repo gets installed we will now configure port mirroring on a specific interface. I'm going to give you the one-liner you will need to execute the span as well as a thorough breakdown so that you can apply it to your device.
        i. ovs-vsctl -- --id=@q get port tap110i1 -- --id=@m create mirror name=spann select-all=true output-port=@q -- set bridge vmbr15 mirrors=@m
            1. Ovs-vsctl -- --id=@q get port tap110i1
                a. This part of the script assigns a variable @q to a new tap interface. This tap corresponds with the VM id (110) and then our second interface (i1)
                    i. You will need to potentially change the VM id to reflect your SecurityOnion2 VM id in your proxmox deployment
            2. -- --id=@m create mirror name=spann select-all=true output-port=@q
                a. This creates a mirror @m that selects all traffic that it sees and sends it to port @q which we defined earlier as our new tap interface
                    i. You shouldn't have to change anything
            3. -- set bridge vmbr15 mirrors=@m
                a. We are assigning a mirror to vmbr15 which sniffs all traffic it sees
                    i. As long as you followed my network diagrams you shouldn't have to change any settings
    d. If you were able to successfully run the one-liner you should have received a uuid value of your new port. You can then use the command ovs-vsctl list Mirror to show if you have any mirrors inside your Proxmox deployment

```
root@proxmox:~# ovs-vsctl -- --id=@q get port tap110i1 -- --id=@m create mirror name=spann selec
t-all=true output-port=@q -- set bridge vmbr15 mirrors=@m
48f92b2a-e607-4e4d-9c36-52bffe7fdfbd
root@proxmox:~# ovs-vsctl list mirror
_uuid               : 48f92b2a-e607-4e4d-9c36-52bffe7fdfbd
external_ids        : {}
name                : spann
output_port         : c8f6443c-25c4-46d3-acc7-7de3a4233089
output_vlan         : []
select_all          : true
select_dst_port     : []
select_src_port     : []
select_vlan         : []
snaplen             : []
statistics          : {tx_bytes=0, tx_packets=0}
root@proxmox:~#
```

   e. You're now done configuring a mirror! The last thing to mention is that if for whatever reason your SecurityOnion2 VM or proxmox itself restarts your mirror will be gone and you will have to Putty/SSH into Proxmox and retype the command. You can write a cronjob/script to resolve this issue if you so please.

7. Congrats! If you were to login to your SOC/Kibana instance you should now start seeing network traffic. As an example I SSH'd into my metasploitable box from my Kibana and was able to successfully see logs! Again, I can't overstate how complicated these steps are and if they don't work for you the first time don't get discouraged and just keep trying!



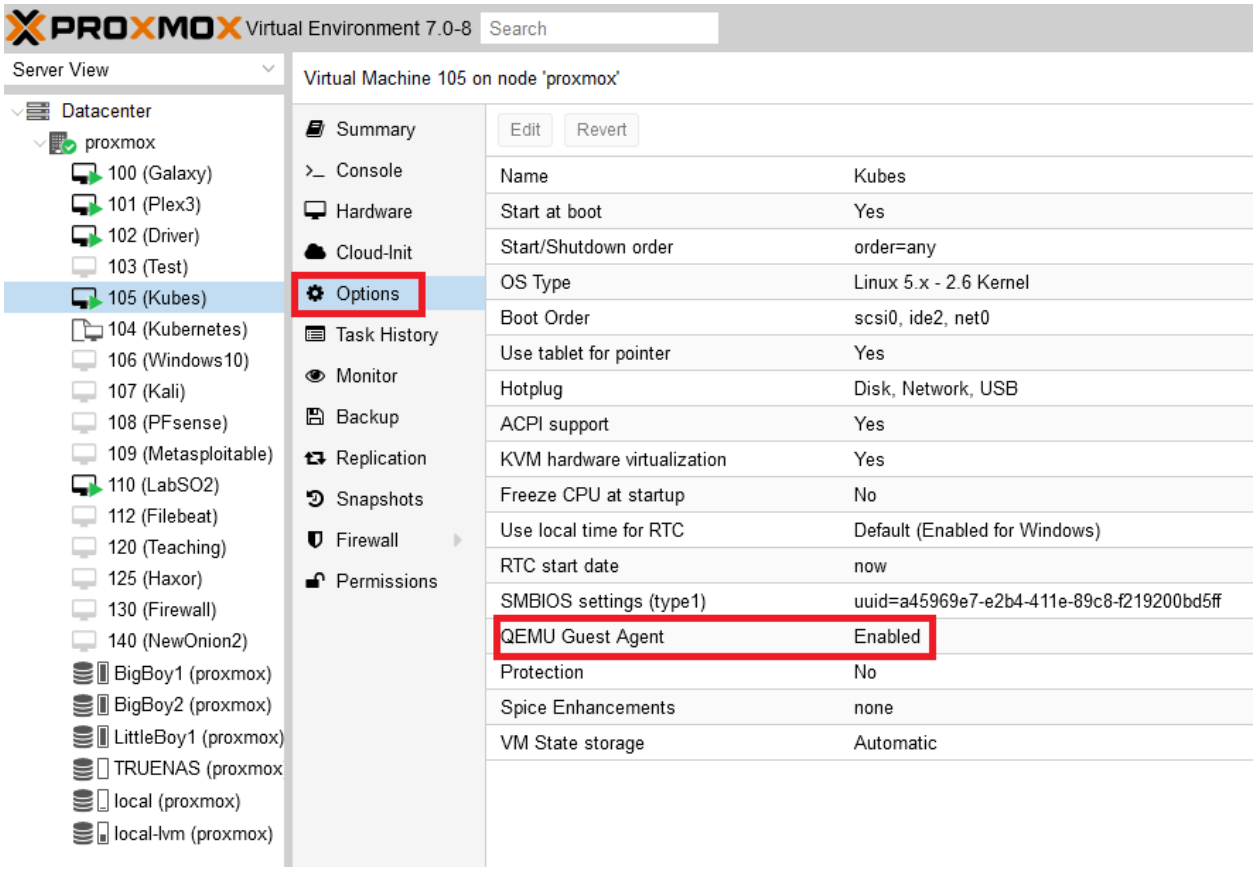
6. Ubuntu/Kubernetes

While not necessary by any means, I thought I would include a walkthrough on setting up your own Kubernetes cluster on a single node deployment (one vm) and show you how to run and manage a service (Wireguard) inside Kubernetes using a docker container.

1. The first step is installing an OS. In this case I will be using **Ubuntu 20.04.3 Live Server amd64 edition.**
    a. Download your OS of choice (I recommend a server installation for less overhead) and upload it into your Proxmox.
2. From here you will want to go to the proxmox web interface and create a new VM by clicking “create VM” in the top left. From here select the following options on each screen.
    a. Name the VM something appropriate and take note of the VM id. In this case I’ll be using VM ID 104. Additionally, if you want your Kubernetes services to always be up when your proxmox is up; make sure you click the advanced tab at the bottom and make sure “Start at Boot:” is selected.
    b. Select your ISO image and make sure the Guest OS type is Linux, version is 5.x-2.6 kernel.
    c. Under System you can leave everything default.
    d. Under Hard Disk assign hard drive space according to your needs. For this implementation I’m only going to need 40GiB .
    e. Assign as many as you can afford, I’m going to give it 6 cores.
    f. Under Memory, I’m going to enable Ballooning Device and set the Memory to 4096 and the Minimum memory to 2048. This way my machine will always have 2GB allocated but can go up to 4GB if needed.
    g. I will leave my network configuration the same.
    h. I click confirm and start my VM.
3. Now we will install the Ubuntu Server OS. From here follow the steps, but use your preferences.
    a. Choose English and press enter.
    b. Click continue without updating as we will be doing that later anyways.
    c. Press done to choose the standard keyboard.
    d. Press done to accept the default DHCP addressing of your NIC. Take note of the IP address assigned.
    e. Press enter to skip the proxy.
    f. Press enter to leave the default mirror. This is where Ubuntu will look for new repos. We will be adding to this later.
    g. You will want to assign the whole disk as this is the space we allocated for it previous. Just tab down until you can select done.
    h. Press done to confirm hard drive formatting.
    i. Press continue to confirm wiping of virtual drive.
    j. Give your setup a name and username. Later on, we will be giving your username root privileges for docker so keep that in mind.

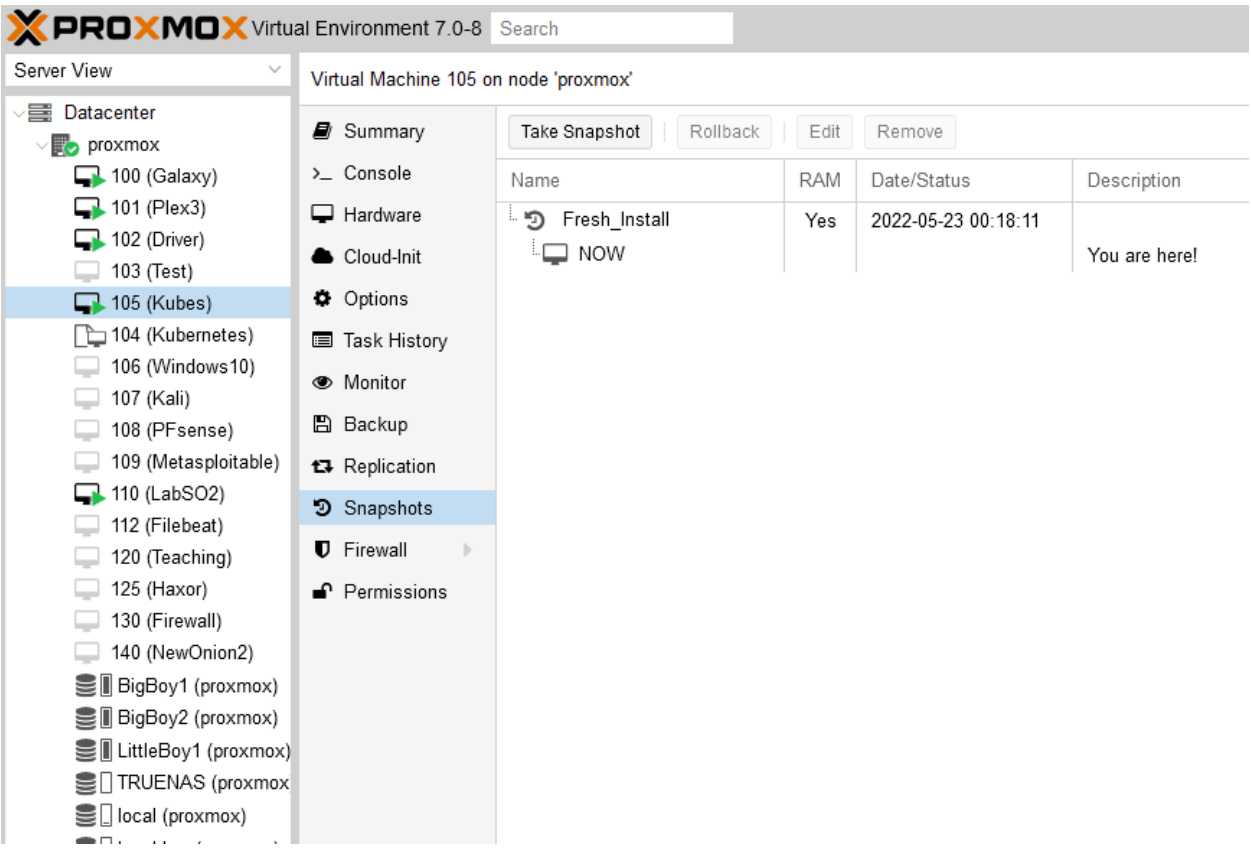
k. You can choose whether or not to enable OpenSSH. I would recommend it because we will be using a terminal emulator like putty after finishing the install.
l. Press enter to confirm default packages. Again we will only be installing the necessary packages later.
m. Wait for the install, once finished we will reboot and be ready to putty/ssh into the server.

4. Now that the OS is installed we will be installing Docker and Rancher. Rancher is a Kubernetes management interface. Allowing us to manage multiple docker containers. Because this is very basic, we will only be installing docker containers on this server (you can have multiple servers called nodes).
    a. SSH/Putty into your server using the credentials you just created. (This will enable the ability to easily copy/paste)
    b. Now we will update the system, you can do show using these 2 commands (while typing in your root password)
        i. *Sudo apt-get update*
        ii. *Sudo apt upgrade -y*
    c. (Optional) From here it's a good idea to take a snapshot incase you need to revert back to a know good state. First we have to install the qemu guest agent. To do so, type this command.
        1. *Sudo apt-get install qemu-guest-agent*
        ii. After it's successful, you will want to shut down your VM and then go into the options tab of the Ubuntu VM.



iii. From here enable the QEMU Guest Agent and reboot your machine. You should now have the guest agent installed.

iv. Now you can simply go to the Snapshots tab and click “Take Snapshot” to take a snapshot of your fresh install that you can rollback to at any time.



d. Now we will install docker through a curl command. Copy and paste this into your terminal to install.

   i. *curl https://releases.rancher.com/install-docker/20.10.sh | sh*

e. Now let’s give our user root privilege over docker containers (substitute XXX with your username)

   i. *Sudo usermod -aG docker XXX*

f. Now you will need to logout and back in for the changes to save.

g. Once logged back in, confirm that you can run docker commands by typing in the following command:

   i. *Docker version*

```
mitchell@kubes:~$ docker version
Client: Docker Engine - Community
 Version:           20.10.7
 API version:       1.41
 Go version:        go1.13.15
 Git commit:        f0df350
 Built:             Wed Jun  2 11:56:38 2021
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true

Server: Docker Engine - Community
 Engine:
  Version:          20.10.7
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.13.15
  Git commit:       b0f5bc3
  Built:            Wed Jun  2 11:54:50 2021
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.4
  GitCommit:        212e8b6fa2f44b9c21b2798135fc6fb7c53efc16
 runc:
  Version:          1.1.1
  GitCommit:        v1.1.1-0-g52de29d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
mitchell@kubes:~$ ~7
```

h. We will now want to install Rancher with the proper configuration settings. Copy and paste these lines into your session. If that doesn't work, use the link provided to copy and paste. https://rancher.com/docs/rancher/v2.5/en/installation/other-installation-methods/single-node-docker/

   i. docker run -d --restart=unless-stopped \
      -p 80:80 -p 443:443 \
      --privileged \
      rancher/rancher:latest

i. You can now run the command *docker ps* to verify that you both have the docker container running, and that rancher is active on your network.

```
mitchell@kubes:~$ docker run -d --restart=unless-stopped \
> -p 80:80 -p 443:443 \
> -v /opt/rancher:/var/lib/rancher \
> rancher/rancher:latest
Unable to find image 'rancher/rancher:latest' locally
latest: Pulling from rancher/rancher
a8ac3a907045: Pull complete
ddc011b5e45a: Pull complete
183428406682: Pull complete
4383b61779a0: Pull complete
69e5bbd60a3f: Pull complete
263c8e71aebe: Pull complete
44edf2f8127e: Pull complete
df6122681172: Pull complete
805a19149124: Pull complete
51cba9d98e19: Pull complete
7bf4da5aef95: Pull complete
f3455eff5db9: Pull complete
17042d61ef59: Pull complete
0eca1f00aa64: Pull complete
d14813cf6598: Pull complete
cfa7ba581d24: Pull complete
7efc8b2f77be: Pull complete
5f99bef3c776: Pull complete
1b950e7852e4: Pull complete
aadf9192f288: Pull complete
Digest: sha256:ae5135c25b2141bb2aac8a03a9afd77e845f36b9a6c000377c858233aae355d4
Status: Downloaded newer image for rancher/rancher:latest
e902ed17bcd2aafe98a54697676c37a60edc21400e2e93a2ff8250fab9b2ac1a
mitchell@kubes:~$ docker ps
CONTAINER ID   IMAGE                  COMMAND           CREATED         STATUS                                 PORTS     NAME
S
e902ed17bcd2   rancher/rancher:latest   "entrypoint.sh"   3 seconds ago   Restarting (1) Less than a second ago            peac
eful_cohen
mitchell@kubes:~$ _
```

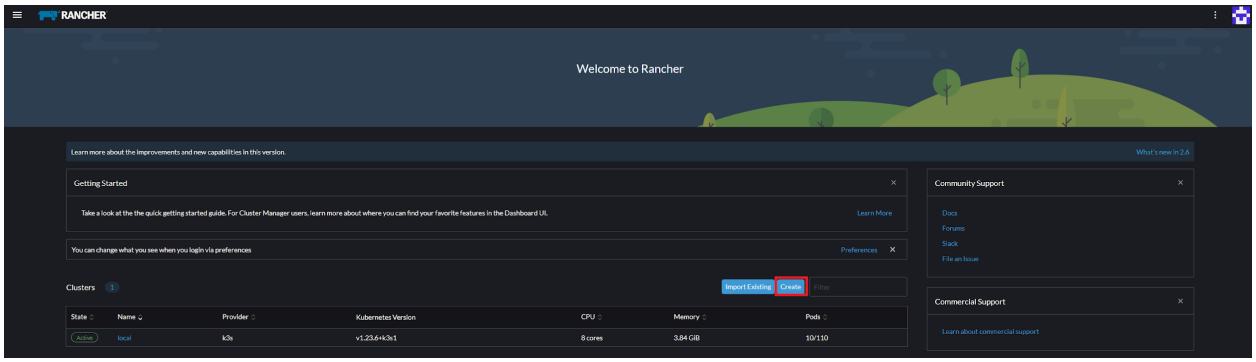
j. The last step is we will need to get our “first password” in order to login to rancher that can be done using the command below: (Where XXX is the Container ID when running the command *docker ps*)

   i. *Docker logs XXX 2>&1 | grep “Bootstrap Password:”*

```
mitchell@kubes:~$ docker logs blad7e16efce 2>&1 | grep "Bootstrap Password:"
2022/05/20 19:33:29 [INFO] Bootstrap Password: w7wl45kvw679bhqt4st2245jph6444kpb7ljjhl5f66fcgft7t9vcd
```

k. Now we login to our Rancher environment to begin setting up kubernetes! To access the webgui, navigate to the IP address of your server and use the Bootstrap Password (you can copy from a putty terminal using right-click)

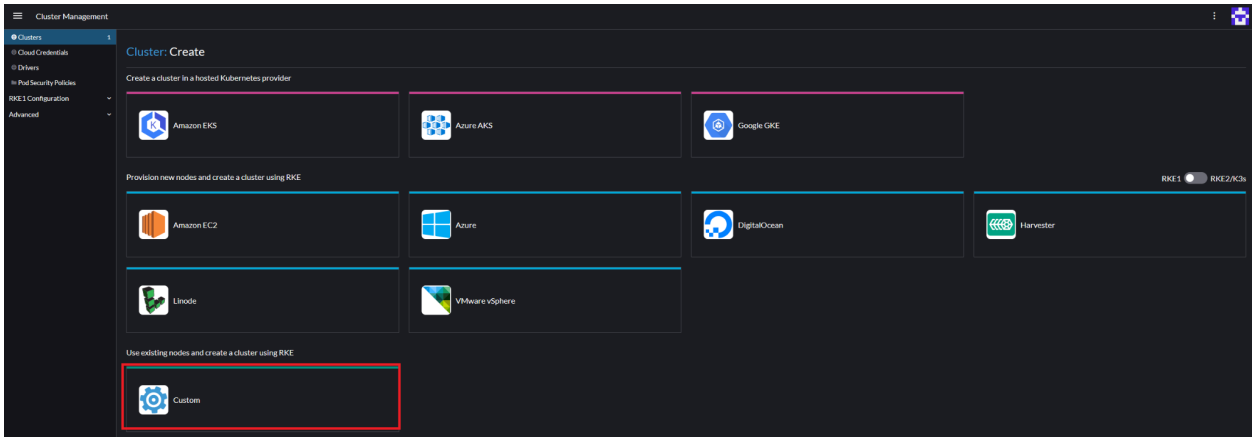
l. From here you will be prompted to create a new password. Also click the agree to the EULA.



5. Now we can begin creating configuring our environment! The first thing we will do is create a new cluster, then as a demo configure wireguard to give a running example of a proper setup.

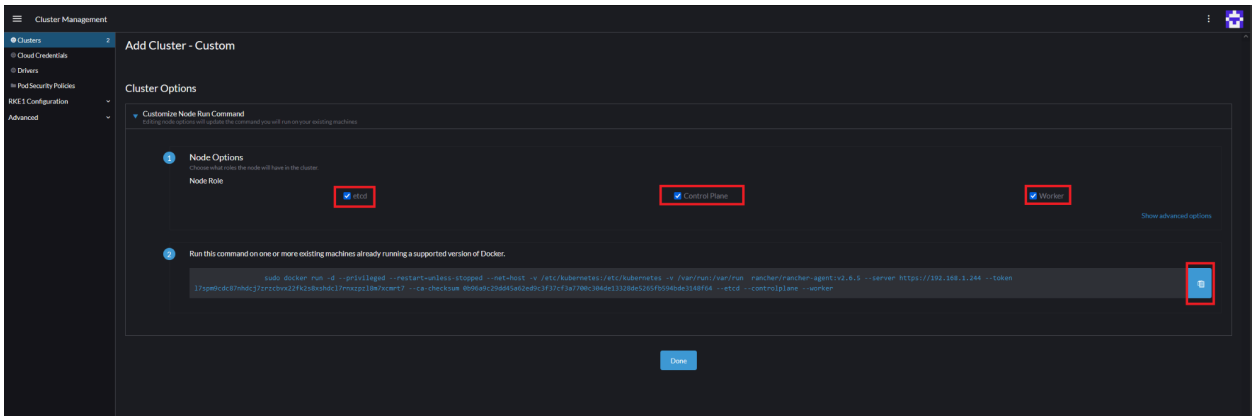
   a. Once you’ve logged into the WebUI, the first thing we will do is create a cluster, to do so click the Create button as highlighted above.

b. Then under custom we will want to create a Custom template.



c. Next, the only setting you will have to change is giving a name to your cluster.

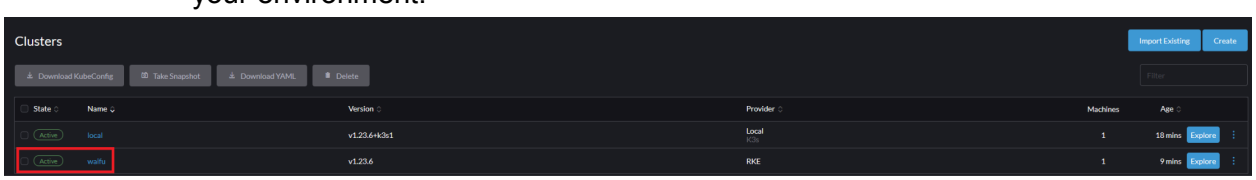
d. Next you will be prompted what node options you want and will be presented with a long string of code. Make sure to select all the node role options and then copy the contents of the command onto your clipboard.



e. Next you will paste the contents into your terminal to bind the docker image to kubernetes and have it be managed with rancher.

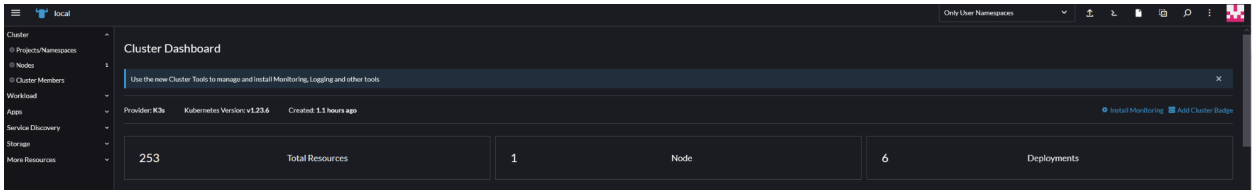
```
mitchell@kubes:~$ sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v
/var/run:/var/run  rancher/rancher-agent:v2.6.5 --server https://192.168.1.244 --token 17spm9cdc87nhdcj7zrzcbvx22fk2s8xshdcl
7rnxzpzl8m7xcmrt7 --ca-checksum 0b96a9c29dd45a62ed9c3f37cf3a7700c304de13328de5265fb594bde3148f64 --etcd --controlplane --wor
ker
```

f. Lastly, you will just need to wait ~10 minutes for the cluster to become active in your environment.

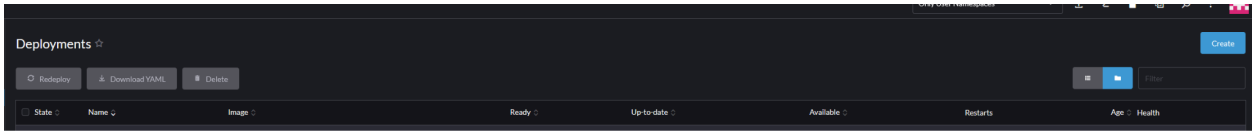


6. The last step is a bonus Wireguard instructional tutorial built using Kubernetes. This will require some potential tweaking on your end but is how I got mine working. The main steps are deploying a new workload on your cluster, then configuring you settings inside the workload, configure port forwarding on your router and then testing your connection.

   a. The first step will be to make a new workload, to do that click on your cluster in the top lefthand corner, here I'm using local.



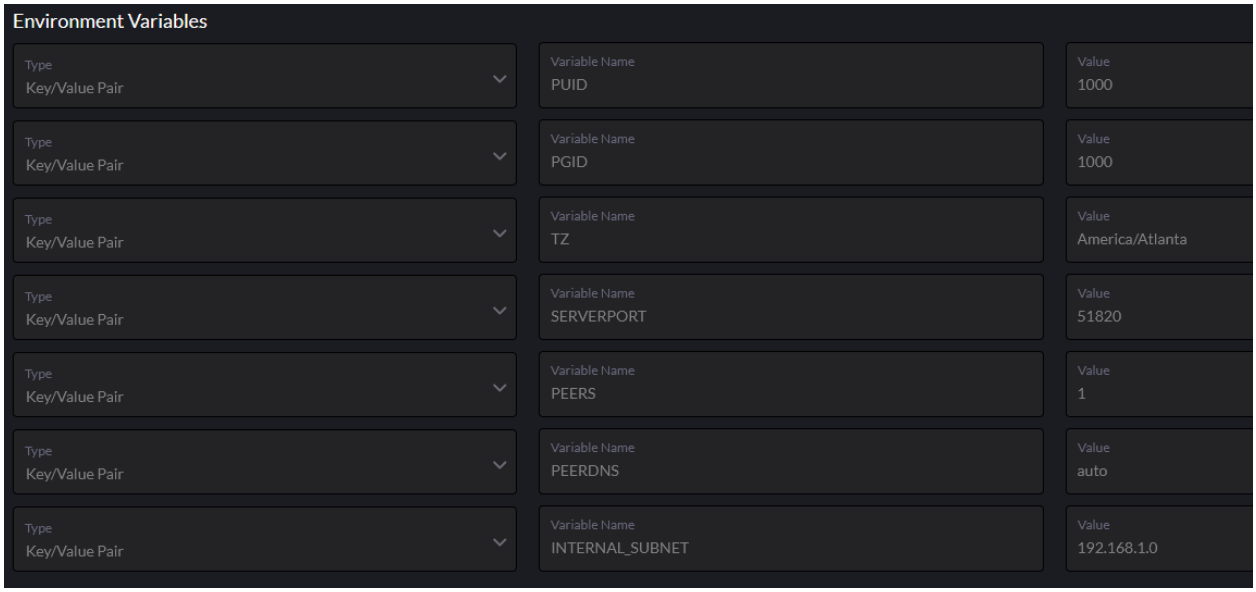
b. Then you will want to click on the Workload tab and then Deployments. Then click Create in the upper right-hand corner.



c. Here is where you will spend most of your time configuring Wireguard. Under each tab I will give you a list of the configurations I used that worked for me but your mileage may vary. The first thing you will want to do is give your workload a name, I named mine wireguard and leave the replicas at 1.

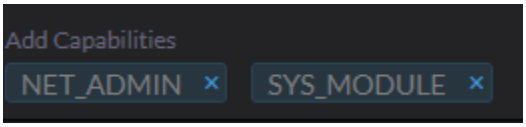
   i. General

      1. Under “Image” put linuxserver/wireguard
      2. Under “Ports” add a new Cluster IP service, name it wireguard, put the Private Container Port at 51820, the protocol UDP, and the Public Host Port at 51820
      3. Under Environment Variables you will be adding quite a few Key/Value Pairs, the first are your PUID and PGID, the value for these should be 1000. You can check this by logging into your server and typing id. Next will be your TZ variable which will be whatever timezone you want. The serverport variable will be the port that we listen on 51820. The Peers will be the number of devices allowed to connect, I set mine to 1 as only my laptop will be able to connect via VPN. You can set the PeerDNS to auto as it doesn’t matter. Lastly set your Internal subnet to the subnet of your internal IP schema that your server resides in.



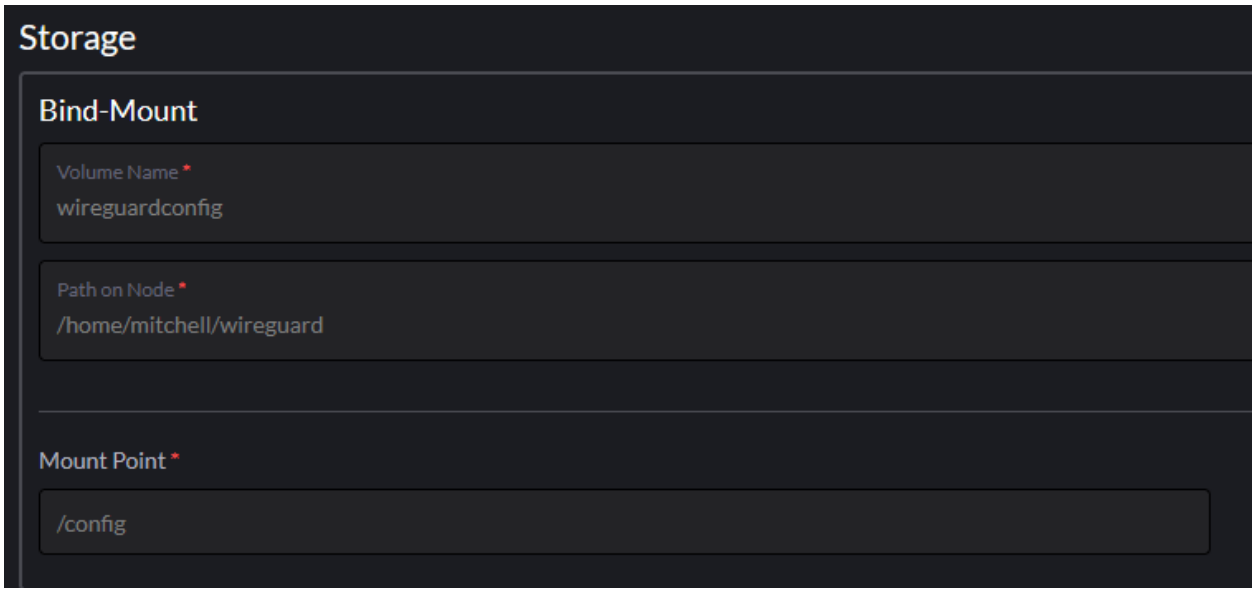
ii. "Scaling and Upgrade Policy"
    1. Choose the option "Recreate: Kill ALL pods, then start new pods"

iii. "Security Context"
    1. Add these 2 capabilities to give your docker image these authoritative rights.
        a. NET_ADMIN & SYS_MODULE

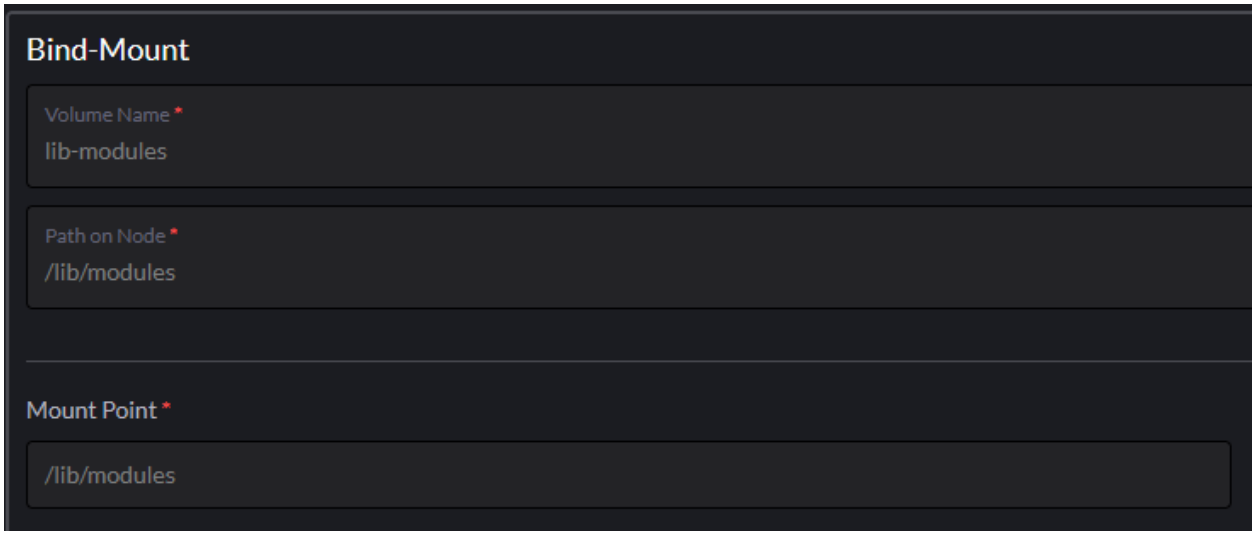


iv. "Storage"
    1. First you will create a new directory on your server to put the config file wireguard will need to create. I created mine on my users homepage.

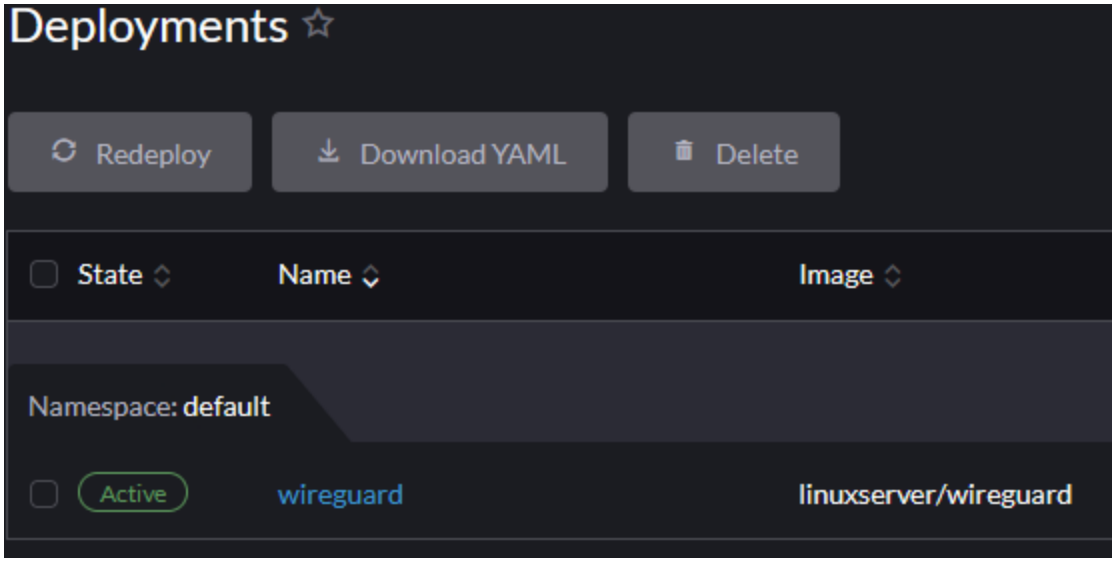
    2. You will then click "Add Volume" and select "bind-mount". Name your volume mount wireguardconfig and under "Path on Node" type out the absolute filepath for the directory you just made. Then under mount point put /config



3. You will then add another bind-mount volume, except this time name it lib-modules and for the path put /lib/modules and the mount point put /lib/modules. This will point to the dependencies wireguard needs to run.



d. Click create at the bottom to start your new deployment. Don't worry if you receive a "Method PUT" error, and just wait a couple of minutes and go to your Deployments tab on the left and see if your new workload is active.



e. In order for this to properly work you will need to configure port forwarding on your home router to allow traffic that hits your router to be forwarded to you Kubernetes cluster. This varies from router/vendor but look up the appropriate guide to do port forwarding. A quick tip is to make sure that when setting up port forwarding is the listening port will be 51820 UDP.

f. To make sure everything is working you will want to click on the “Pods” tab on the lefthand side click on the ellipses next to your running Pod (docker container) and click “View logs”. You should see a QR code that you can use to connect another host as well as private information such as the public IP address exposed etc.

g. The last steps would be setting up wireguard on a client and then connecting to your VPN. You will know it works if you get an IP address back in your private IP space. From here you could access internal drives, enable remote desktop and more. This was just a brief example on how to get Kubernetes up and running.

Special Thanks

If you have read this far, thanks! I really appreciate you taking the time to read through this madness and I hope that this proves to be a valuable resource to you.
I just wanted to give a quick shoutout to Christopher Traxler and Brycen Guilfoyle for always helping me troubleshoot and answering the tough technical questions as well as opening my eyes to homelab deployments. Another huge shoutout to my wife for always being supportive and allowing me to purchase and setup this in the first place.
If you have any questions or problems please feel free to reach out to me! If I got anything wrong also reach out to me and let me know (also sorry in advance), I'm always willing to learn from smart people! My personal email is mitchellgibsonnm@gmail.com