Bare Metal ISO Load Using Staging Server

Jan 2014
Rob Garrett
SoftLayer Sales Engineer

Document History

Revision History

Date of this revision: 13th Jan 2015	Date of next revision *(date)*

Revision No	Revision Date	Summary of Change	Author	Changes Marked
1	13 Jan 2015	Initial version	Rob Garrett	N

Disclaimer

LIMIT OF LIABILITY/DISCAIMER OF WARRANTY:

THE AUTHOR MAKE NO REPRESENTATIONS OR WRRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETETENSS OF THE CONTETNS OF THIS WORK AND SPECIFICALLY DISCALIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WRRANTIES OF FITNESS FOR A PARTICUALR PURPOSE. THE SOWFTWARE VERSION OR SOFTLAYER'S OFFERING MAY HAVE CHANGED OR DISAPPERAED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ

Table of Contents

1 – Introduction

The purpose of this document is to demonstrate how you can load an operating system onto a bare metal server using an ISO, using a Windows Server for staging.

Considerations

- The operating systems in this guide are not supported by SoftLayer
- No performance or compatibility tests where done following the operating system load
- There are alternative ways of loading operating systems onto bare metal servers, this is the approach that I found the simplest
- Document assumes working knowledge of SoftLayer, Windows and Linux

2 - SoftLayer Environment

For this document I configured two servers, a public cloud Windows server and a Bare Metal server running CoreOS. These where both hourly instances, on the same private network, with 20GB NAS Storage configured on the Windows Server (can be configured on either device)

Staging Server Configuration

Name: Cloud Server
Domain: Edit domain information for this server

Hourly Total: $0.19
Quantity: 1

		Hourly	Setup
Data Center			
London 2			
Computing Instance		$0.021	$0.00
1 x 2.0 GHz Core			
RAM		$0.077	$0.00
4 GB			
Operating System		$0.024	$0.00
Windows Server 2008 R2 Standard Edition (64bit)			
First Disk		$0.005	$0.00
100 GB (LOCAL)			
Public Bandwidth		$0.000	$0.00
0 GB Bandwidth			
Uplink Port Speeds		$0.040	$0.00
1 Gbps Public & Private Network Uplinks			
Public Network Port	1 Gbps Public Uplink	$0.000	$0.00
Private Network Port	1 Gbps Private Uplink	$0.000	$0.00
Anti-Virus & Spyware Protection		$0.000	$0.00
McAfee VirusScan Anti-Virus - Windows			
Advanced Monitoring		$0.000	$0.00
Monitoring Package - Basic			
Network Attached Storage		$0.020	$0.00
20 GB NAS			

Bare Metal server configuration

Name: Bare Metal Server
Domain: Edit domain information for this server

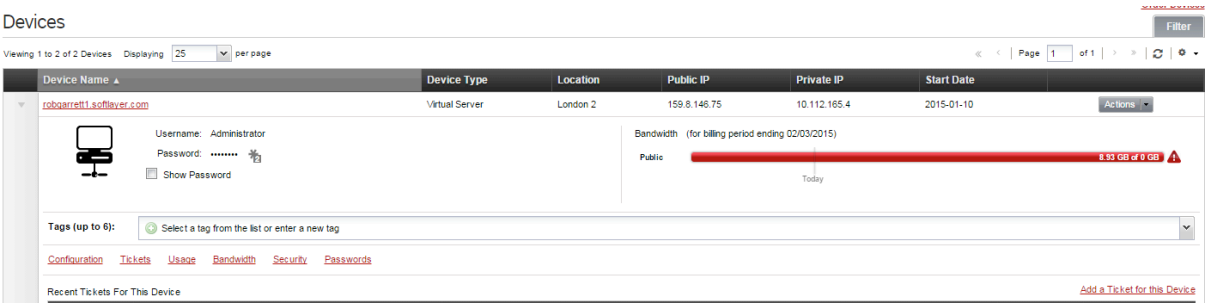
Hourly Total: $1.20
Quantity: 1

		Hourly	Setup
Data Center			
London 2			
First Hard Drive		$0.223	$0.00
400 GB SSD			
Second Hard Drive		$0.223	$0.00
400 GB SSD			
Disk Controller		$0.000	$0.00
Non-RAID			
RAM		$0.427	$0.00
32 GB Registered DDR3 1333			
Server		$0.221	$0.00
Single Processor Quad Core Xeon 1270 V3 - 3.50GHz (Haswell) - 1 x 8MB			
Operating System		$0.000	$0.00
CentOS 7.x (64 bit)			
Public Bandwidth		$0.000	$0.00
0 GB Bandwidth			
Uplink Port Speeds		$0.040	$0.00
1 Gbps Public & Private Network Uplinks			
Public Network Port	1 Gbps Public Uplink	$0.000	$0.00
Private Network Port	1 Gbps Private Uplink	$0.000	$0.00

3 - Staging Server Configuration

Firstly establish an RDP session to the Windows Server; I used Windows Server 2008 R2 for this document.

If you expand the device information from the device list, you will get the public IP and the administrative password, which you will require to establish the RDP session.



Server Base Configuration

You will need to install the latest version of Java to allow the KVM to function, https://java.com/en/download/

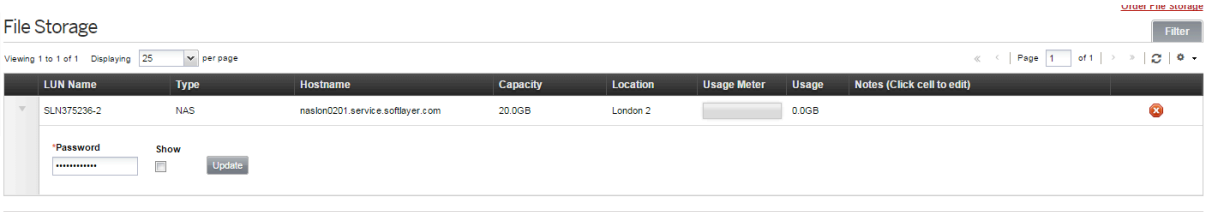
If you are loading CoreOS, you will need Putty and PuttyGen: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

I also installed the Google Chrome browser, but that is a personal preference.

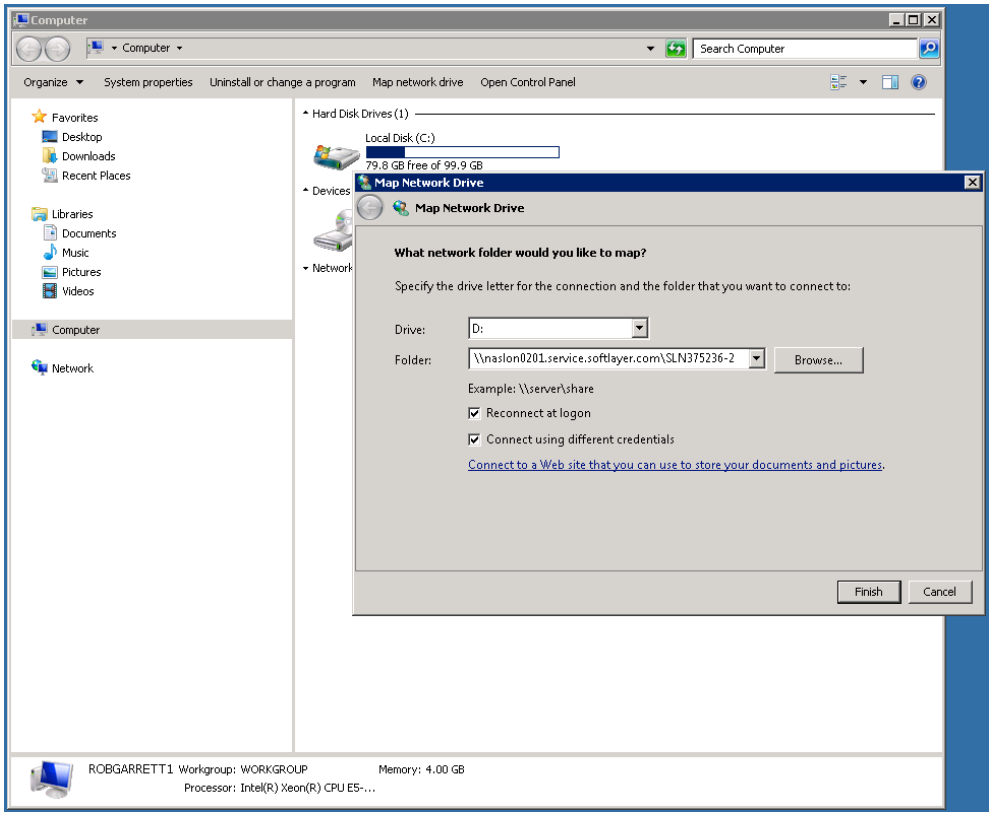
Map NAS Storage

The NAS storage will be required by the bare metal server to load the ISO file from. Firstly you need to map the NAS storage to the server to allow you to load the ISO file onto the storage.

You can get the NAS details from the portal under *Storage/File Storage*.



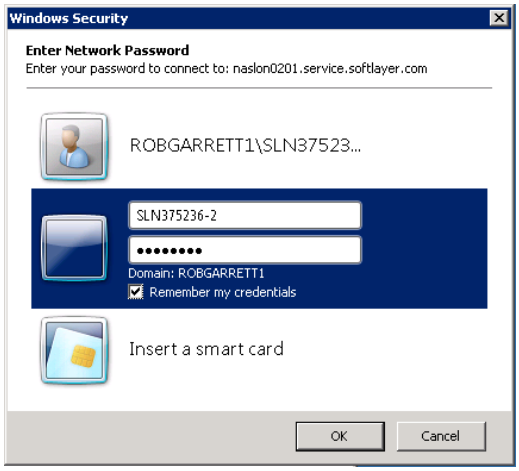
You will need the Hostname, LUN Name, and Password. Open Windows Explorer and click map network drive:



Select a drive letter, and enter the folder, which is made up from the information you gathered for your NAS instance:

\\ {Hostname} \ {LUN Name}

When prompted, enter your LUN Name as your username, and the password for your NAS storage.



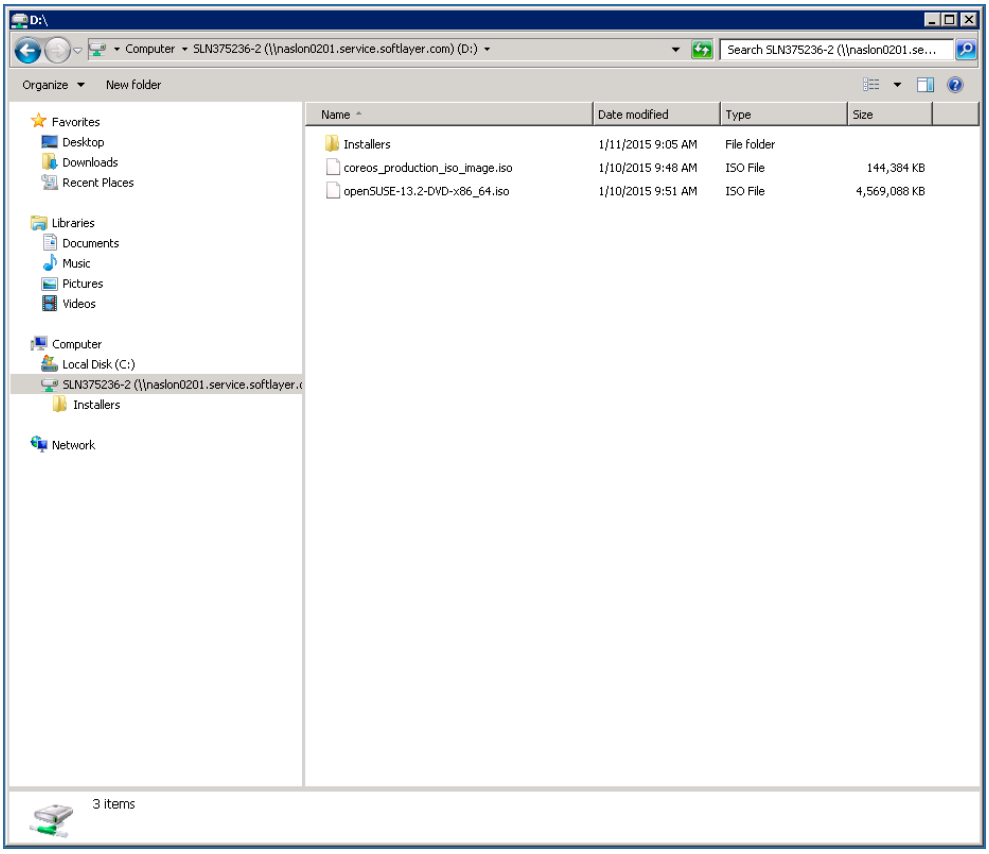
You need to put the ISO of the OS you are loading into the route of the mapped drive:

CoreOS Live CD

https://coreos.com/docs/running-coreos/platforms/iso/

OpenSUSE 64bit Install ISO

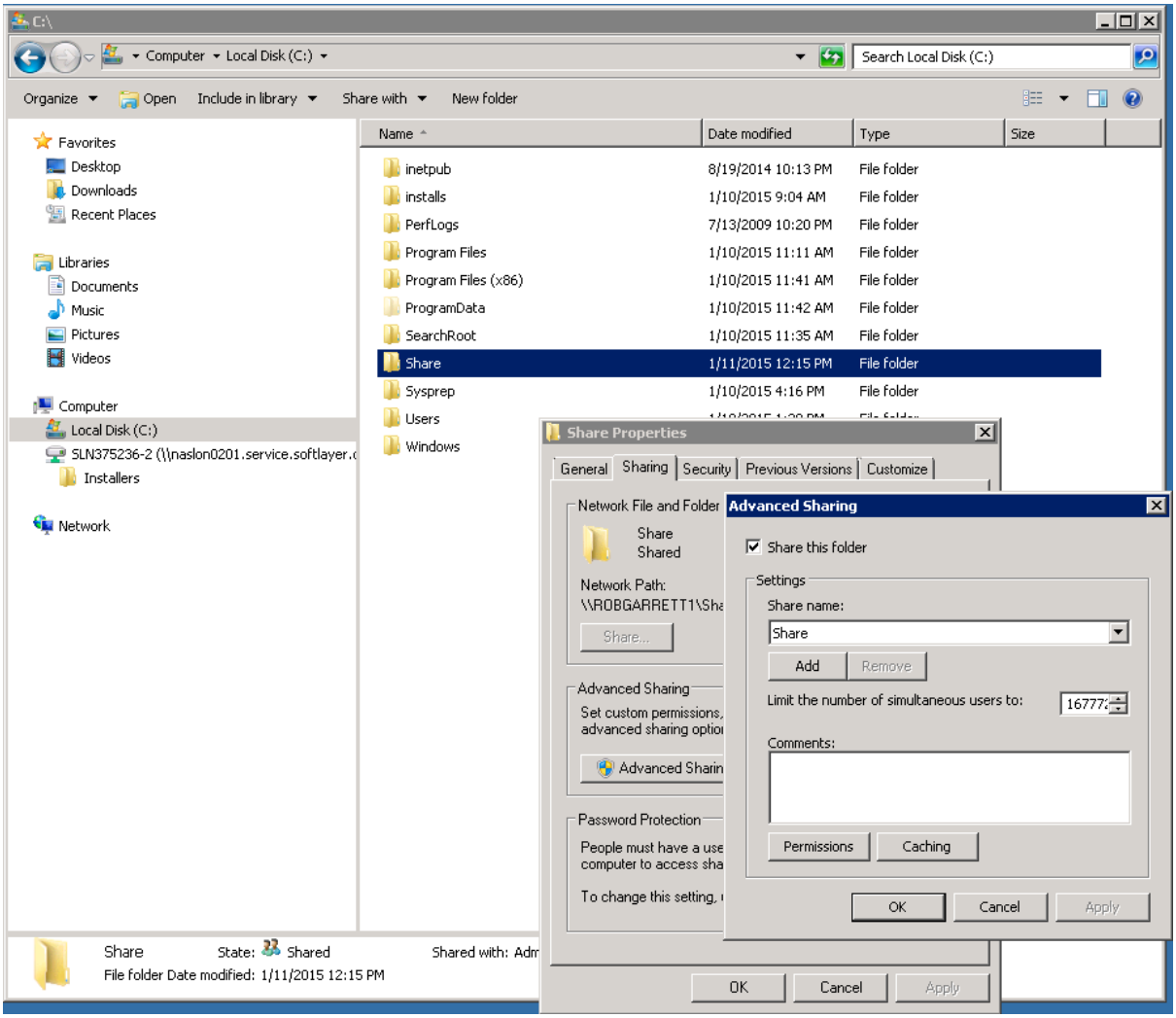
http://software.opensuse.org/132/en



Windows Share (required for CoreOS)

I create a shred folder to transfer the configuration file during the load of CoreOS, if you don't plan to load CoreOS you can skip this step.

Create a share on the C: drive called Share. I give administrator full permissions on this Share, but read only would be sufficient.

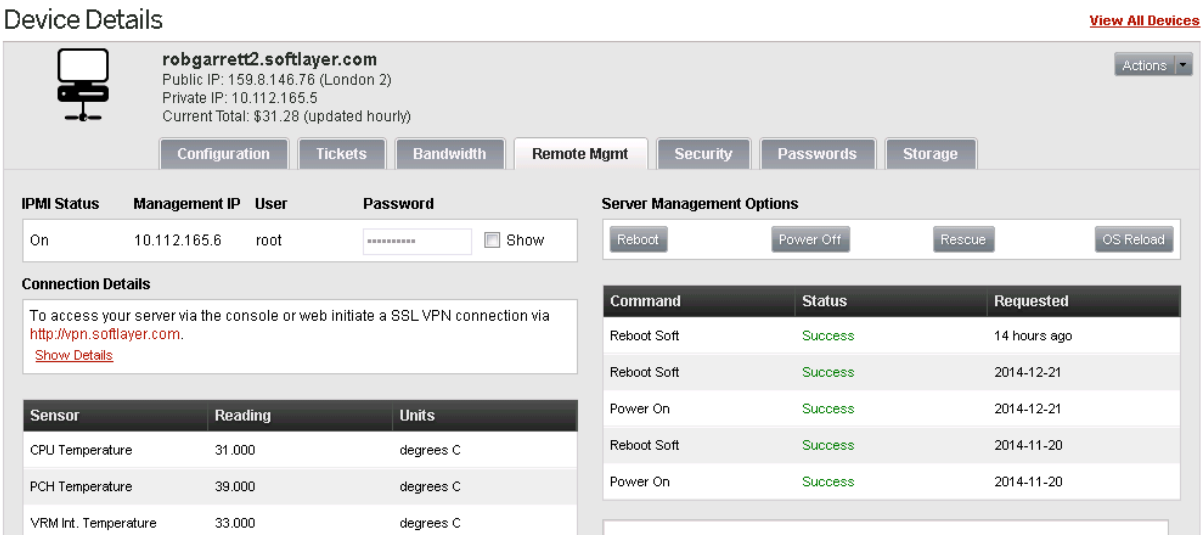


4 - Bare Metal Server Details

The following information will be required to complete the ISO load. This can all be retrieved from the SoftLayer portal.

Setting	Example
Staging Server Private IP	10.112.165.4
Public Adaptor Name	Eth1
Public IP Address	159.8.146.76
Public Subnet Mask	255.255.255.240
Public Gateway	159.8.146.65
Private Adaptor Name	Eth0
Private IP Address	10.112.165.5
Private Subnet Mask	255.255.255.192
Private Gateway	10.112.165.1
Management Username	Root
Management Password	**********

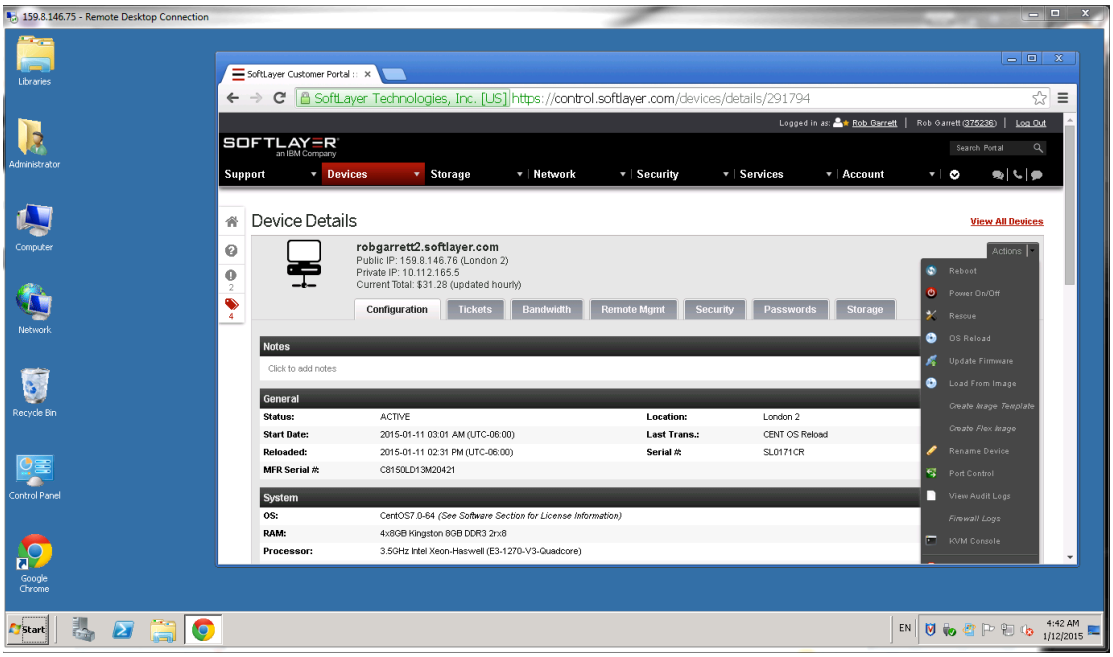
All of the details above can be retrieved from the *Configuration* and *Remote Mgmt* sections in the Device Details in the SoftLayer Portal.



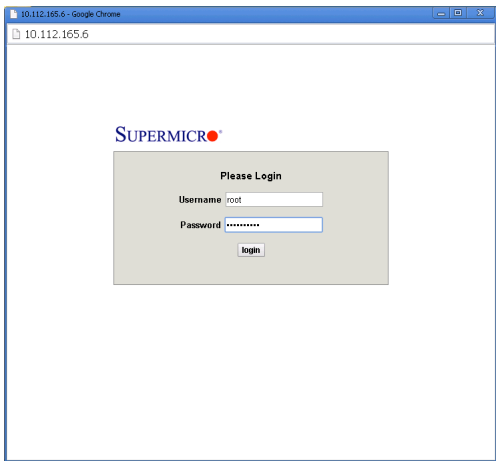
I complete the following from the staging server, which removes the need to establish a vpn connection the private network.

Establish a KVM Session

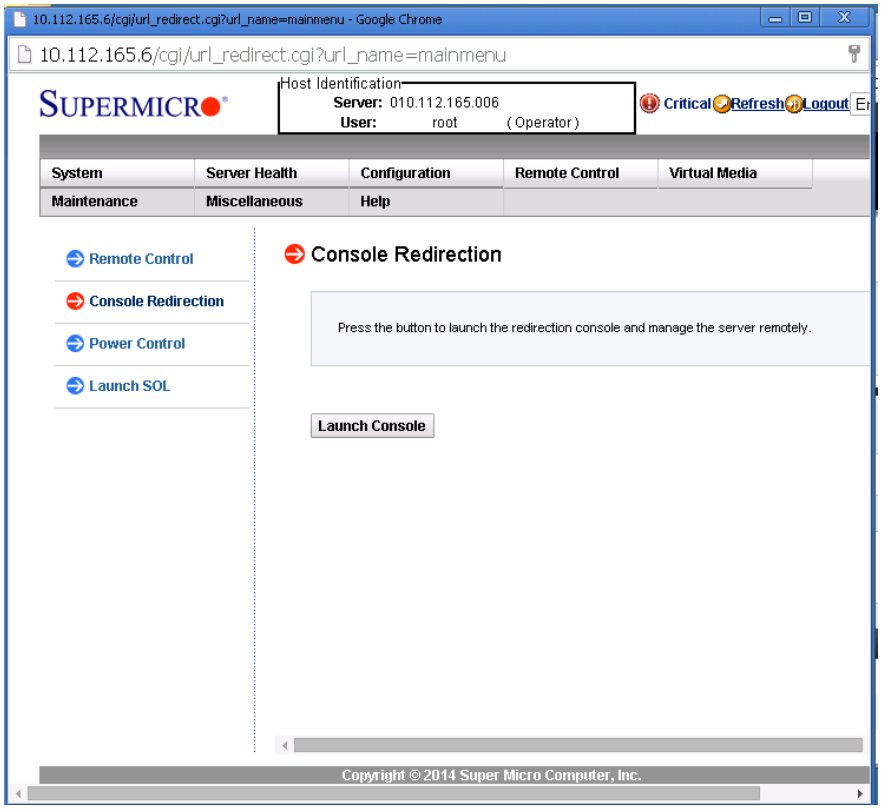
Firstly you need to establish a KVM session to your bare metal server. Log into the SoftLayer portal from the staging server, and select the bare metal server from the device list. From the actions menu, select KVM console.



A new browser window will open; enter the management log in details you gathered earlier.

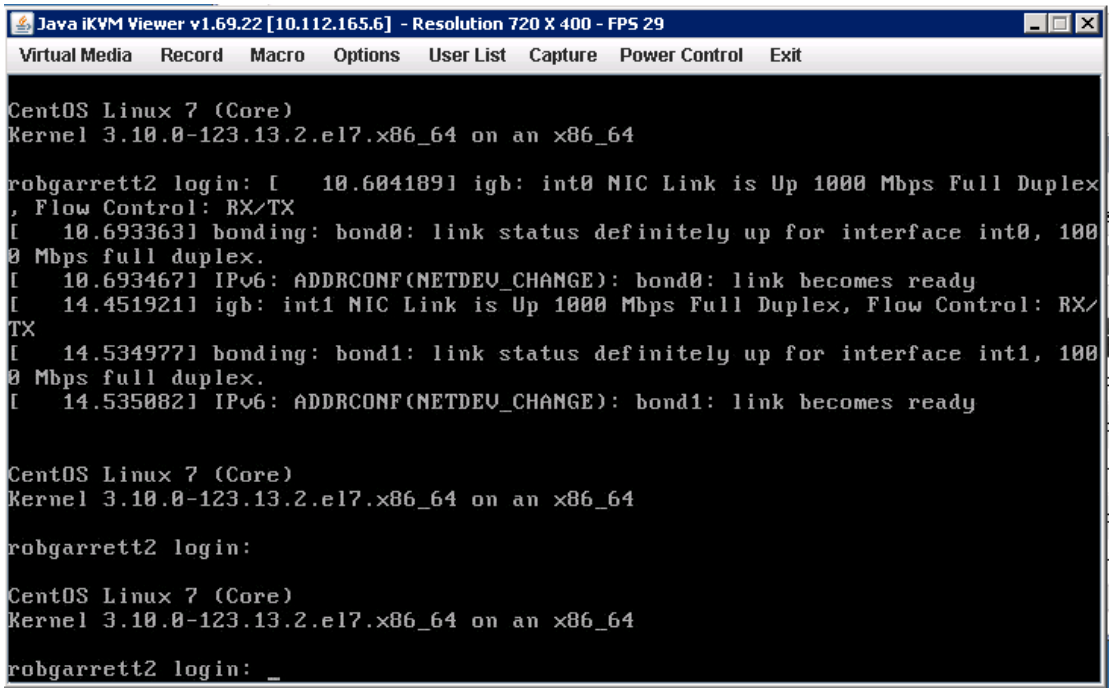


Once the session is open, select Remote Control from the top menu, and then Console Redirection from the left hand side menu.



Click Launch Console, which will trigger a file download. Save the file to disk, and then run this file. When prompted, select to run the application.

Once the KVM session is complete, you should get a console view of the server. Mine is running CentOS currently as shown below.



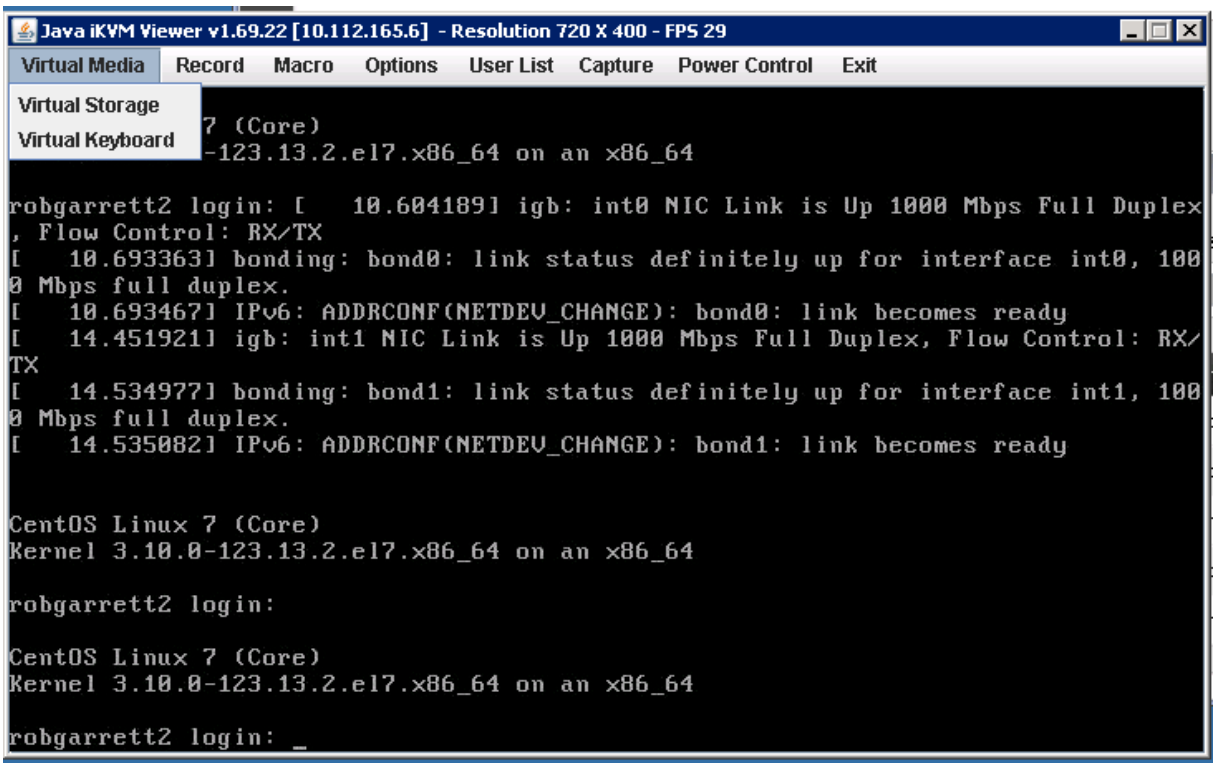
5 - CoreOS Load

References

https://coreos.com/docs/running-coreos/bare-metal/installing-to-disk/
https://coreos.com/docs/cluster-management/setup/network-config-with-networkd/
http://www.freedesktop.org/software/systemd/man/systemd.network.html

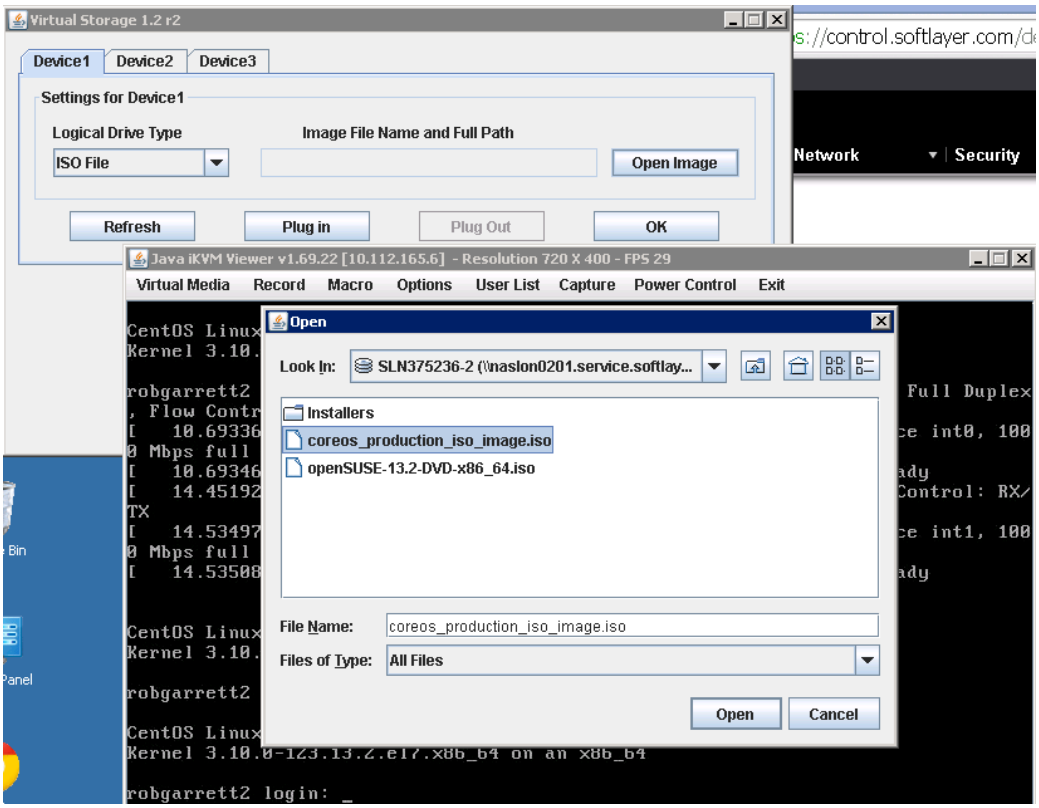
Install CoreOS

From your KVM session, select *Virtual Media/Virtual Storage*.

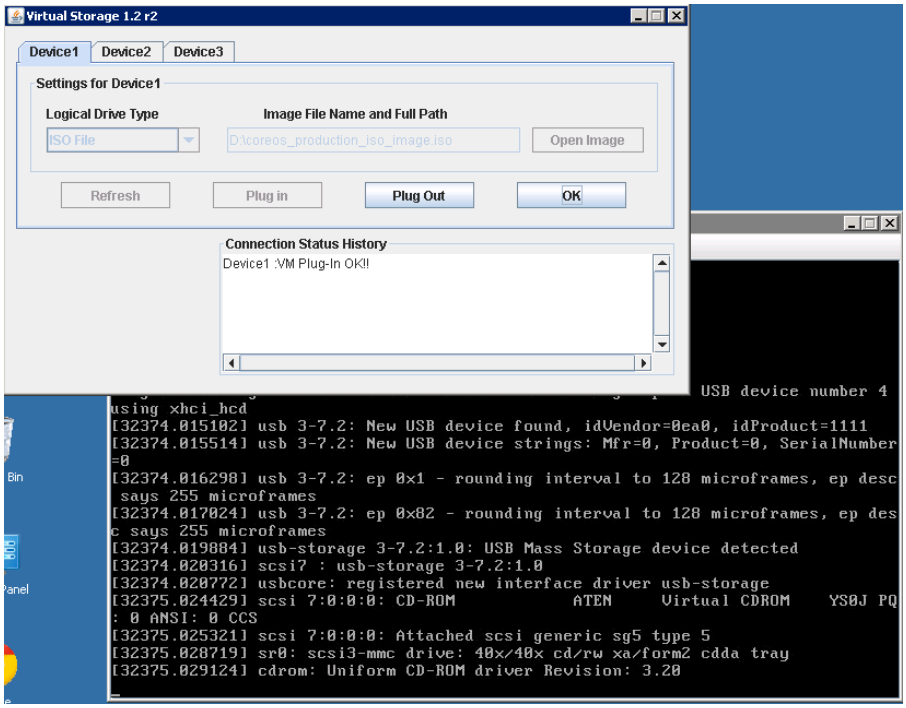


You are going to attach the ISO as device 1, so select ISO as the Logical Device Type.

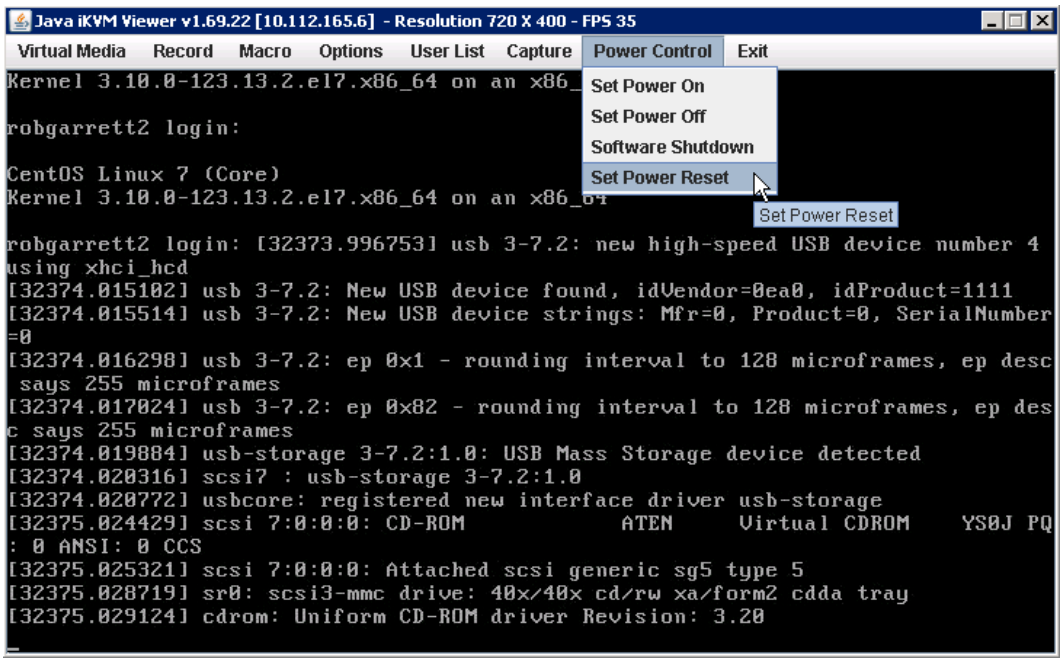
Select *Open Image*, in the *Look In* drop down box, you will see your NAS drive listed, select this and you will see the ISO files that you saved into your NAS Storage previously. Select the CoreOS ISO file, and select *Open*.



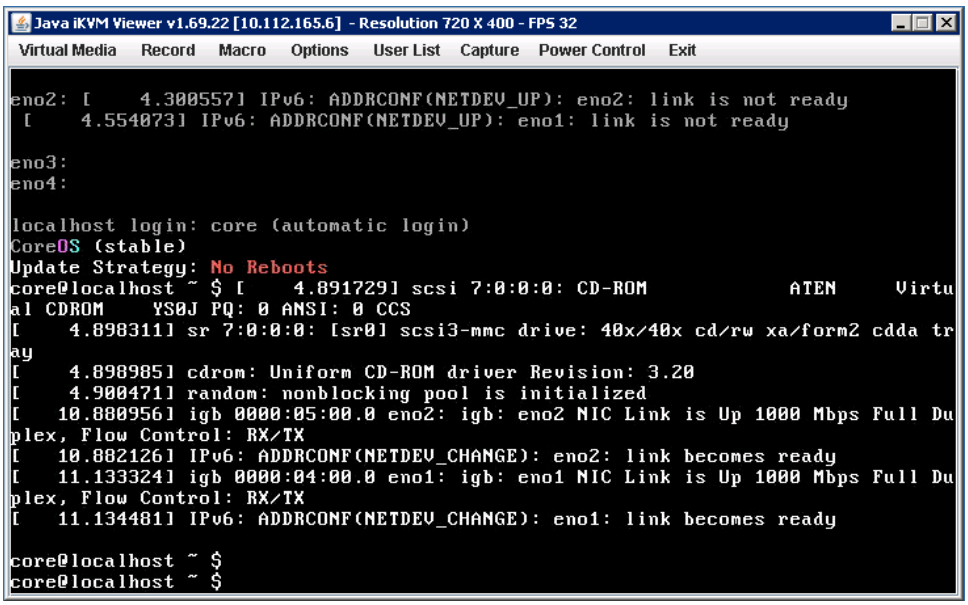
Now select *Plug In*, and the ISO will attach to the server. Click *OK* to close the dialog box.



Now you need to reboot the server, which will then load from the ISO file. By default the bare metal server should load from an attached ISO before the hard drive, if not then you will need to go into the bios and amend this boot order.

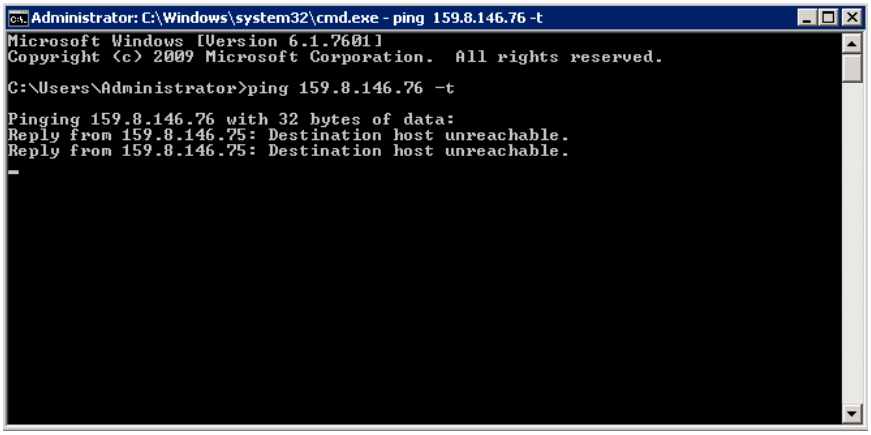


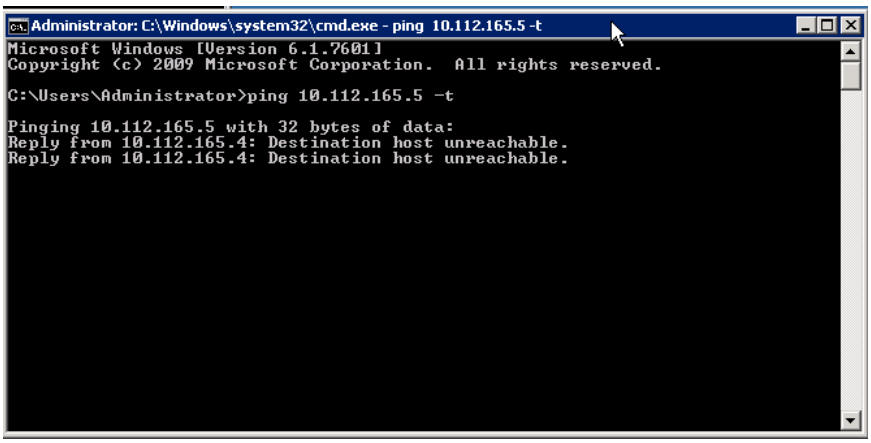
Once rebooted, you should be presented with a CoreOS prompt.



The CoreOS Live CD allows us to install the full OS, at the moment CentOS is still installed and un-changed on the server.

At this point I like to open a couple of command prompts on the staging server, and leave a ping going to both the public and private addresses of the bare metal server. At this point, neither should respond, as there are no DHCP servers on the SoftLayer network, and therefore we will have to go and configure the network settings required.





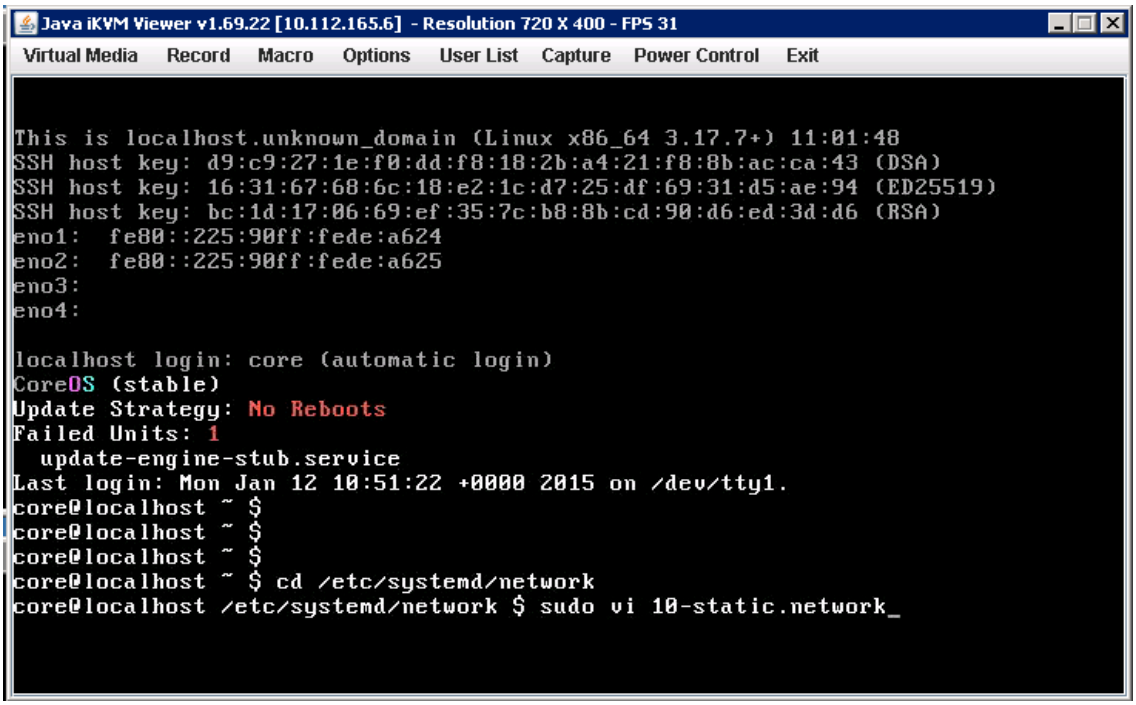
Now you need to configure the Ethernet adaptors, to allow the install files to be retrieved from the Internet. This requires configuration of both the public and private adaptors, as the DNS servers are on the services network, accessed via the private network.

Running *$ ip a*, will show you the adaptors present. Depending on your chassis, you may see a different number of adaptors, but in my case you can see 4, 2 x private and 2 x public. From the information we gathered before, we know that in this case eth0 and eth1 are the adaptors we want to configure, which are showing up below as eno1 and eno2 respectively. As I have no redundant links ordered, I can configure the adaptors individually, whereas a redundant pair would need to be part of a bond.

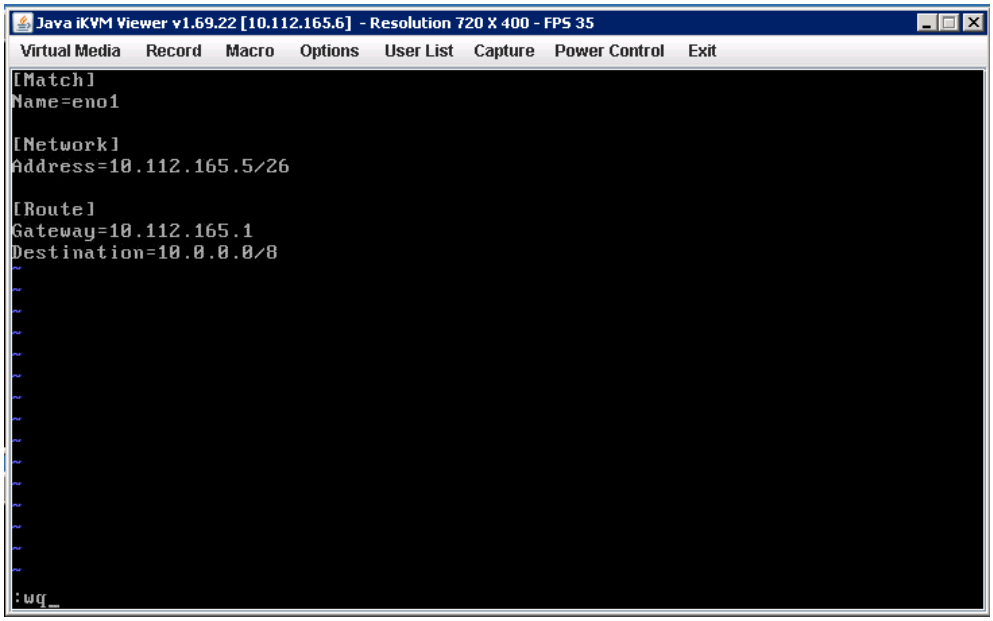
```
Java iKVM Viewer v1.69.22 [10.112.165.6] - Resolution 720 X 400 - FPS 35
Virtual Media   Record   Macro   Options   User List   Capture   Power Control   Exit
[   11.134481] IPv6: ADDRCONF(NETDEV_CHANGE): eno1: link becomes ready

core@localhost ~ $
core@localhost ~ $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    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: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 00:25:90:de:a6:24 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::225:90ff:fede:a624/64 scope link
       valid_lft forever preferred_lft forever
3: eno2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 00:25:90:de:a6:25 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::225:90ff:fede:a625/64 scope link
       valid_lft forever preferred_lft forever
4: eno3: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN qlen 1
000
    link/ether 00:25:90:de:a6:26 brd ff:ff:ff:ff:ff:ff
5: eno4: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN qlen 1
000
    link/ether 00:25:90:de:a6:27 brd ff:ff:ff:ff:ff:ff
core@localhost ~ $ _
```

You need to create 2 files, one for each adaptor. We will create *10-static.network* and *20-static.network* in the folder */etc/systemd/network/*. It doesn't matter which a adaptor we configure in each file.

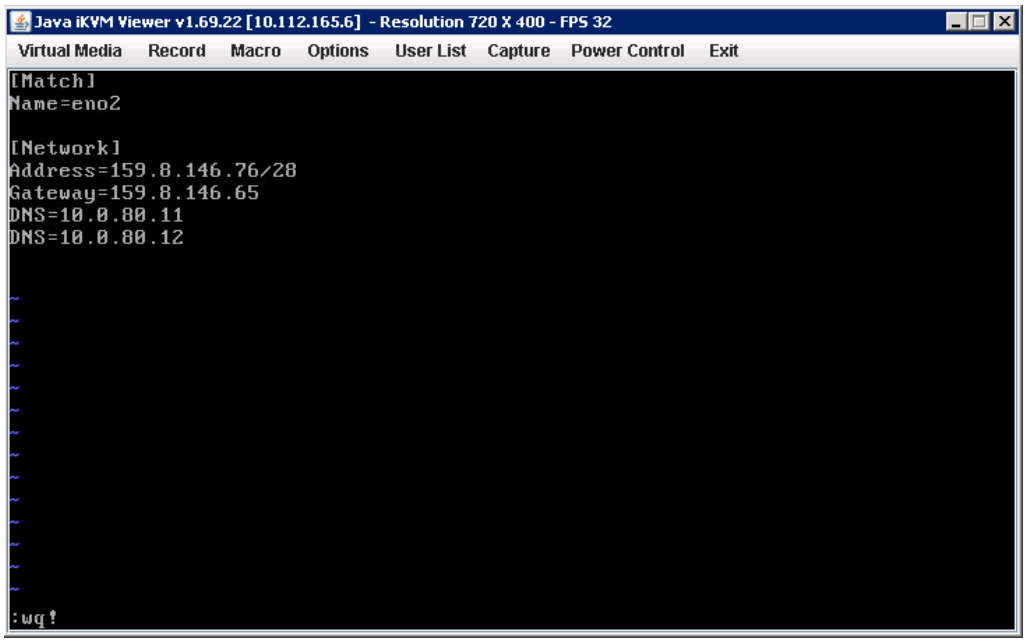
Navigate to the folder above, and create the *10-static.network* file.



The file I created is shown below. This configuration is for the private network , and you will see that a static route to the 10.0.0.0/8 network is defined. If you add the default gateway in the *[Network]* section, CoreOS will create a def ault route for this adaptor. This stops name resolution to the internet working, as the traffic is routed down the wrong adaptor following a DNS lookup.

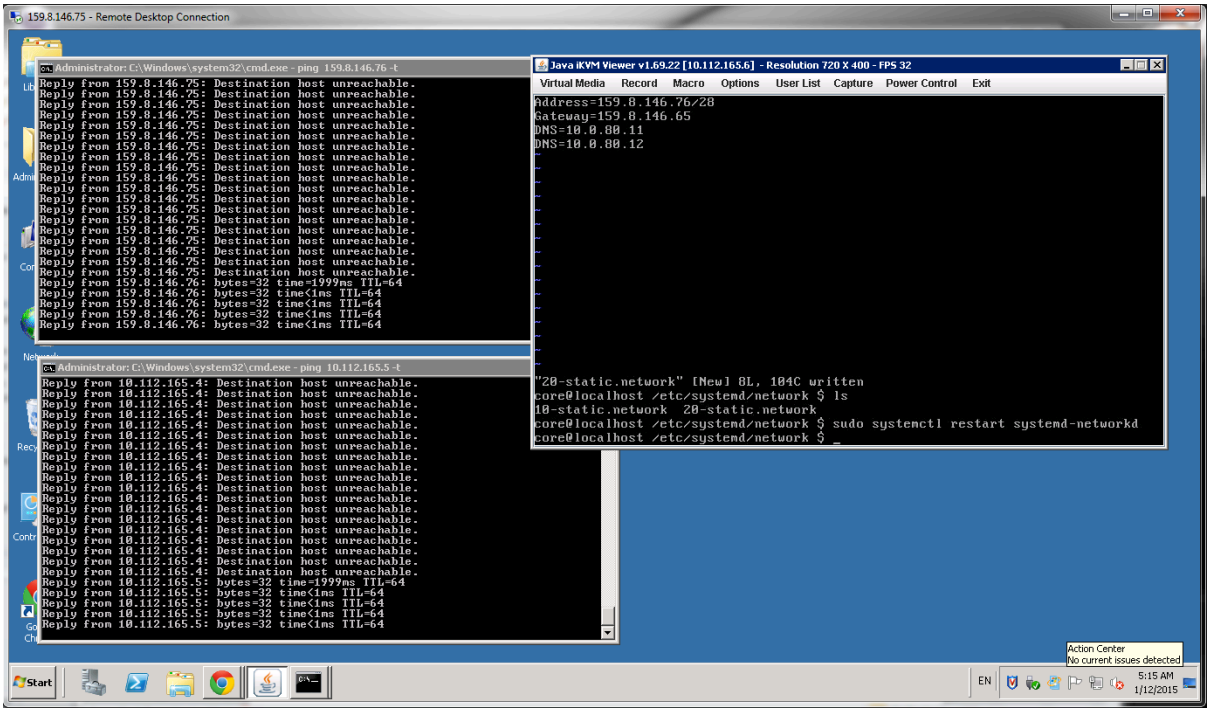


Now complete the process for the *20-static.network* file, in here we will add the D NS server details.



Once these files are in place, you can restart the network services, and both addresses should start responding to the pings we started earlier.

Command to restart network services *$ sudo systemctl restart systemd-networkd*



It is worth running a *$ route* command at this point, to see how the routing table has been set up. As you can see the public adaptor forms the default route, with a static route to the 10.0.0.0/8 network via the private adaptor.

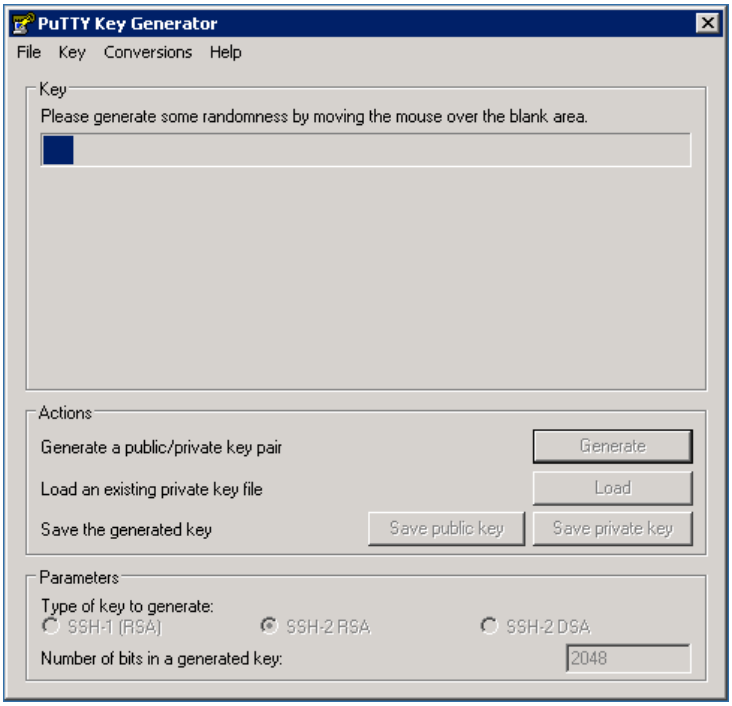
```
core@localhost /etc/systemd/network $ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         159.8.146.65-st 0.0.0.0         UG    0      0        0 eno2
10.0.0.0        10.112.165.1    255.0.0.0       UG    0      0        0 eno1
10.112.165.0    *               255.255.255.192 U     0      0        0 eno1
159.8.146.64-st *               255.255.255.240 U     0      0        0 eno2
core@localhost /etc/systemd/network $ _
```

You should now be able to ping an external address, I tested with google.com. This proves that the DNS is working which will be reuqired to get the installer files.

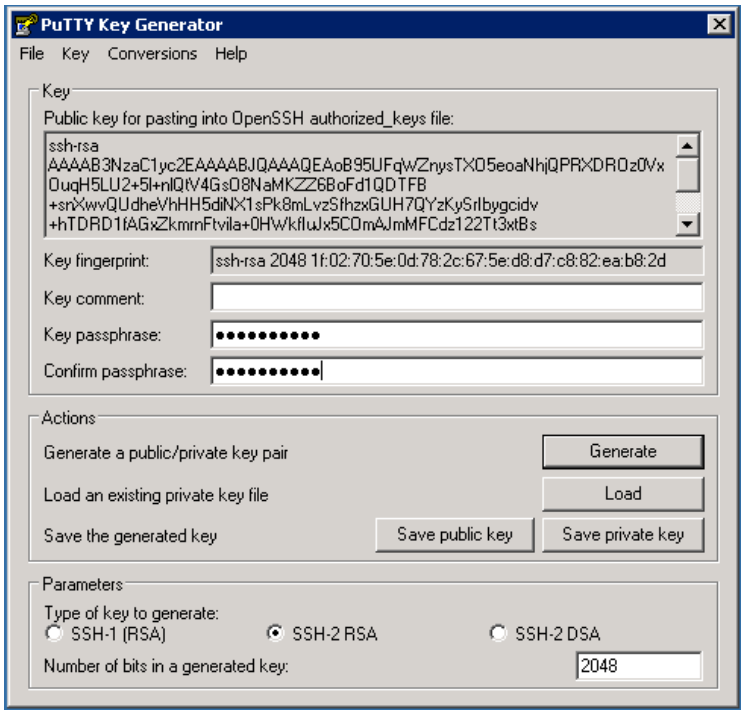
```
Java iKVM Viewer v1.69.22 [10.112.165.6] - Resolution 720 X 400 - FPS 35
Virtual Media   Record   Macro   Options   User List   Capture   Power Control   Exit
~
~
~
~
~
~
"20-static.network" [New] 8L, 104C written
core@localhost /etc/systemd/network $ ls
10-static.network  20-static.network
core@localhost /etc/systemd/network $ sudo systemctl restart systemd-networkd
core@localhost /etc/systemd/network $ ping -c 4 google.com
PING google.com (216.58.208.46) 56(84) bytes of data.
64 bytes from lhr08s07-in-f14.1e100.net (216.58.208.46): icmp_seq=1 ttl=58 time=
1.09 ms
64 bytes from lhr08s07-in-f14.1e100.net (216.58.208.46): icmp_seq=2 ttl=58 time=
1.09 ms
64 bytes from lhr08s07-in-f14.1e100.net (216.58.208.46): icmp_seq=3 ttl=58 time=
1.10 ms
64 bytes from lhr08s07-in-f14.1e100.net (216.58.208.46): icmp_seq=4 ttl=58 time=
1.09 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 1.093/1.097/1.109/0.007 ms
core@localhost /etc/systemd/network $ _
```

By default there isn't a password or any other way to log into a fresh CoreOS system. The easiest way to configure accounts, add systemd units, and more is via cloud config. I create my cloud config file from my staging server, as we need to generate a SSH keypair to allow us to log in following the isntallation.

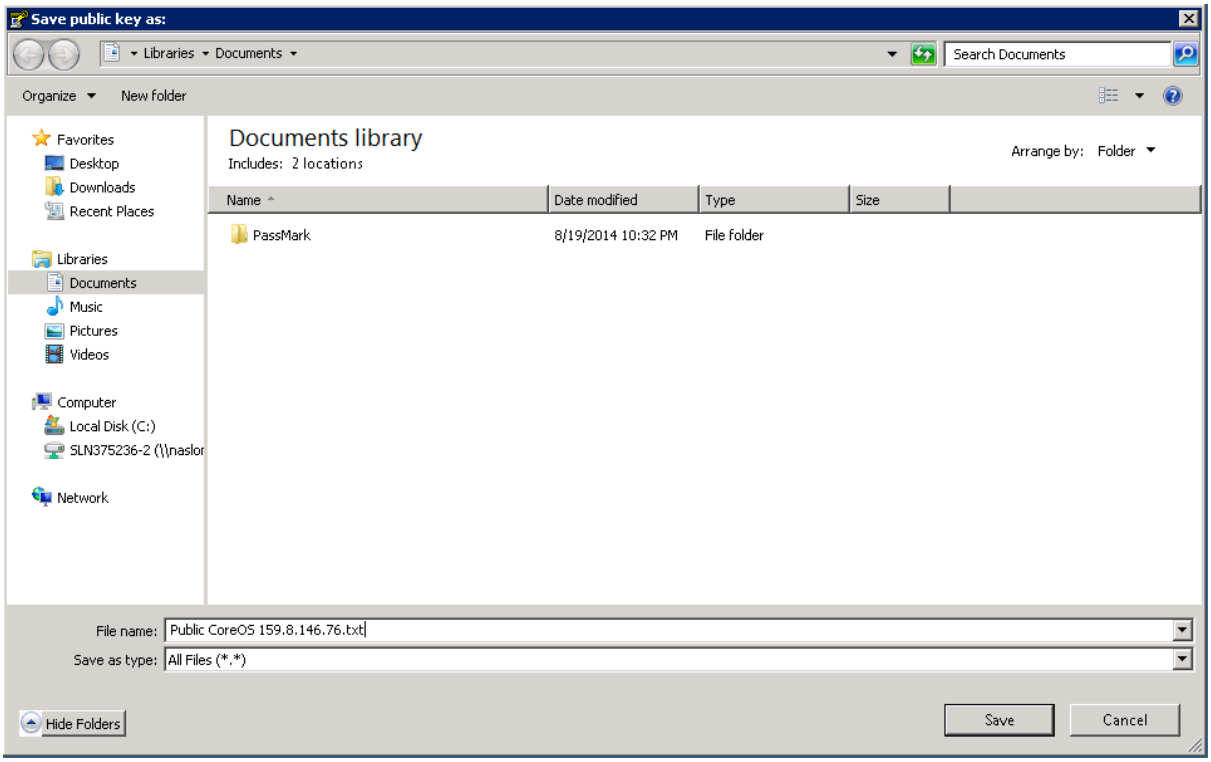
Firstly you need to generate a SSH key pair to use to SSH to your CoreOS server once it is installed. Run PuttyGen and generate a key.

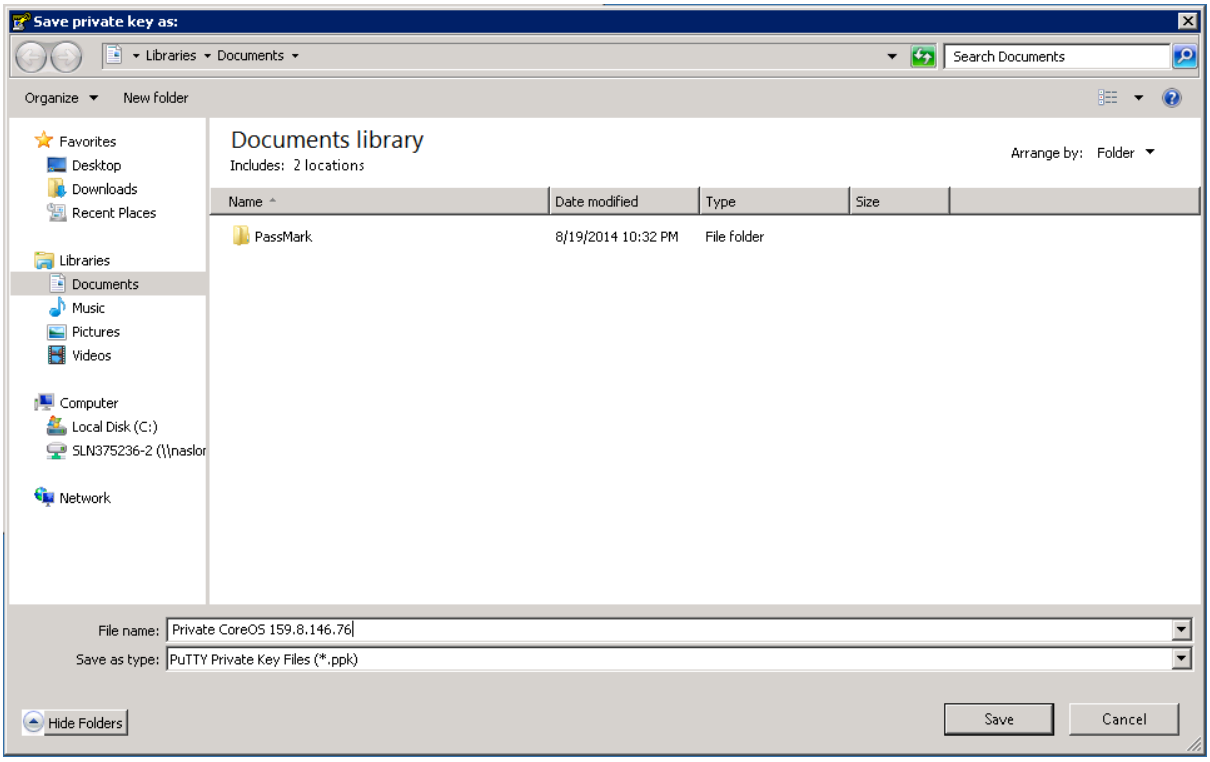


Once your key is generated, remove the key commet, and add a key passphrase.



Save both your public and private key. Remember you will need the private key on any machine that you wish to SSH from, so keep it safe!!

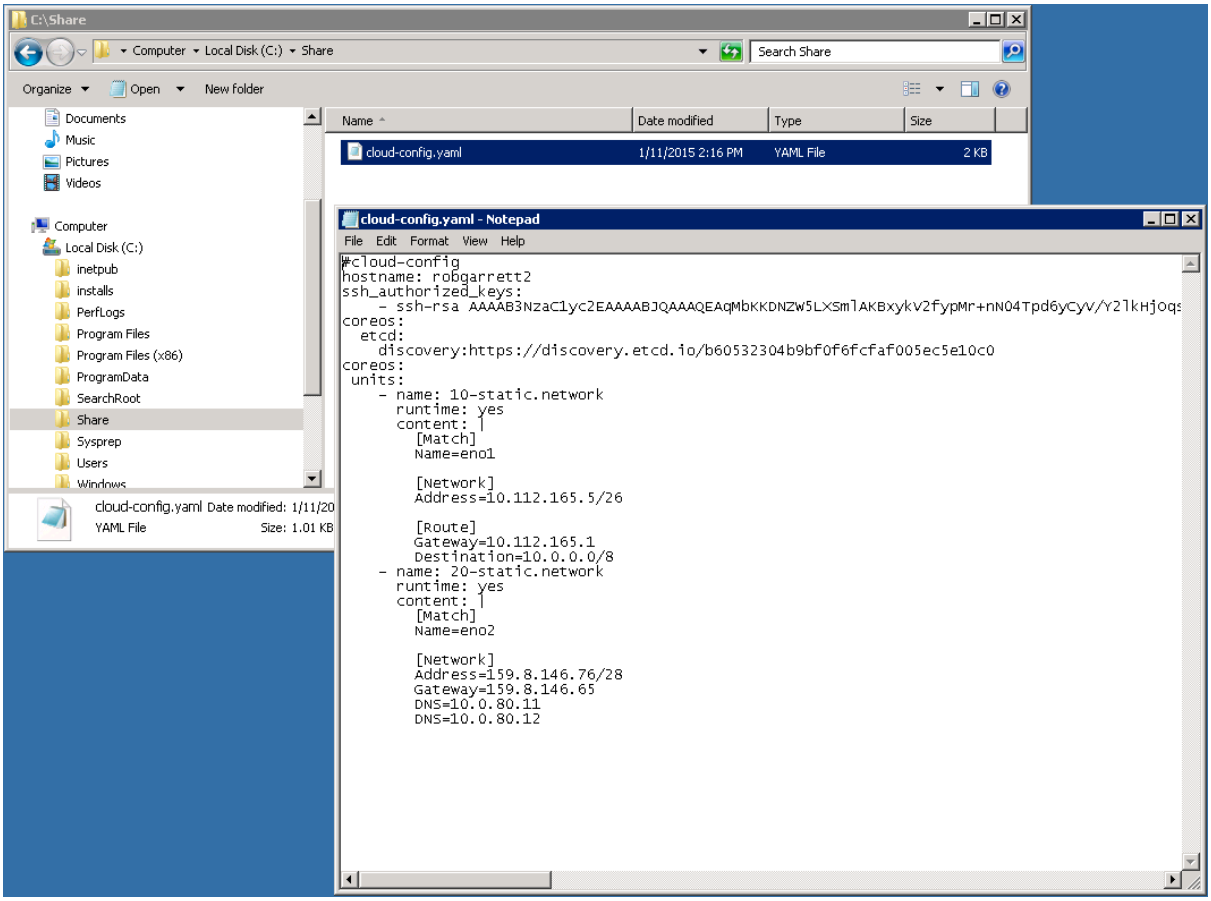




Now you need to create your cloud-config.yaml file in your shared directory. Mine is shown below, you can read more about this file and its construct online. You can use what I have below as an exmaple, you just need to:

1. Copy and paste your public key in
2. Change your IP addresses
3. Generate a new etdc ID: https://discovery.etcd.io/new

Also note that you should not use any tabs in your file, ony spaces. You can verify your config file by pasting it into this website prior to loading it, this could save you a lot of time: https://coreos.com/validate/



Now you have your cloud-config.yaml file saved in your shared folder, you need to copy it across to your CoreOS server.

Here are the commands I run in the picture below to mount the drive and copy the file.

$ sudo mkdir /mnt/win

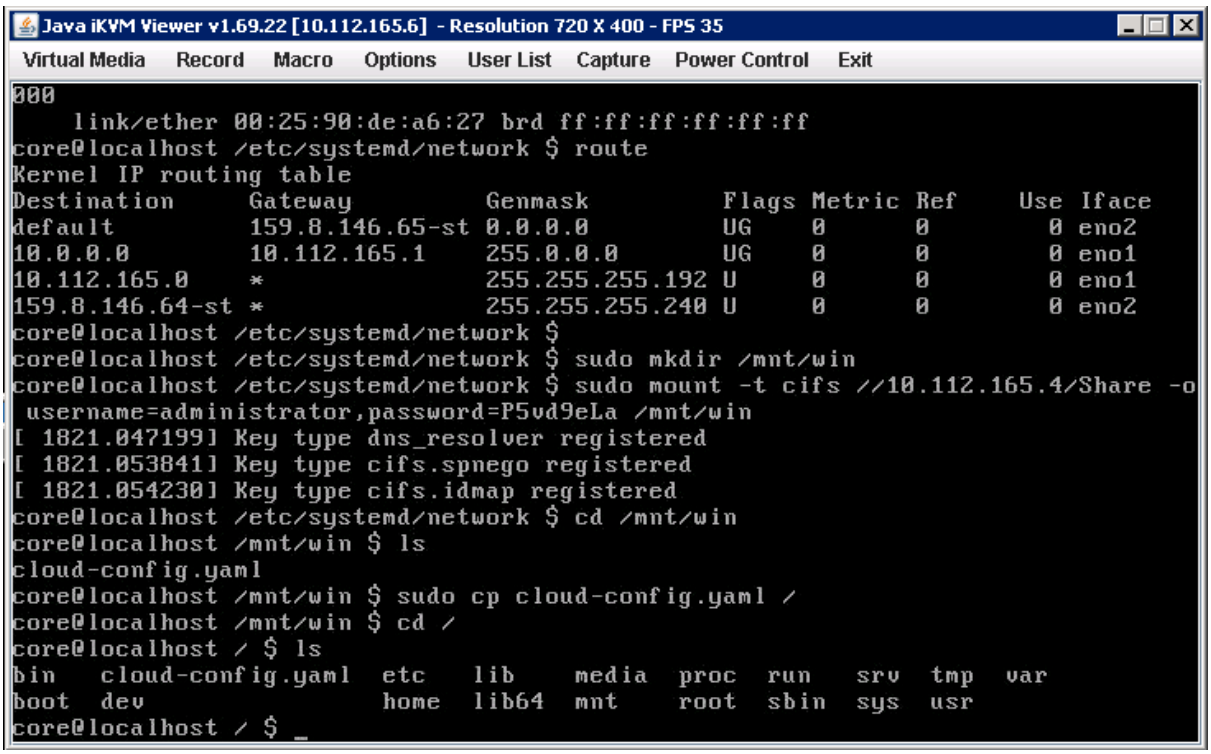
$ sudo mount –t cifs //{IP of Window Server}/Share –o username=adminstrator,password={password} /mnt/win

$ cd /mnt/win

$ sudo cp cloud-config.yaml /

$ cd /

$ ls



Once completed you should see a local copy of you cloud-config.yaml file as above . Finally we need to clear the existing drive, as the installer will detect the partitio ns and fail as it doesn't see a whole drive.

$ sudo fdisk –l will show the current disk structure

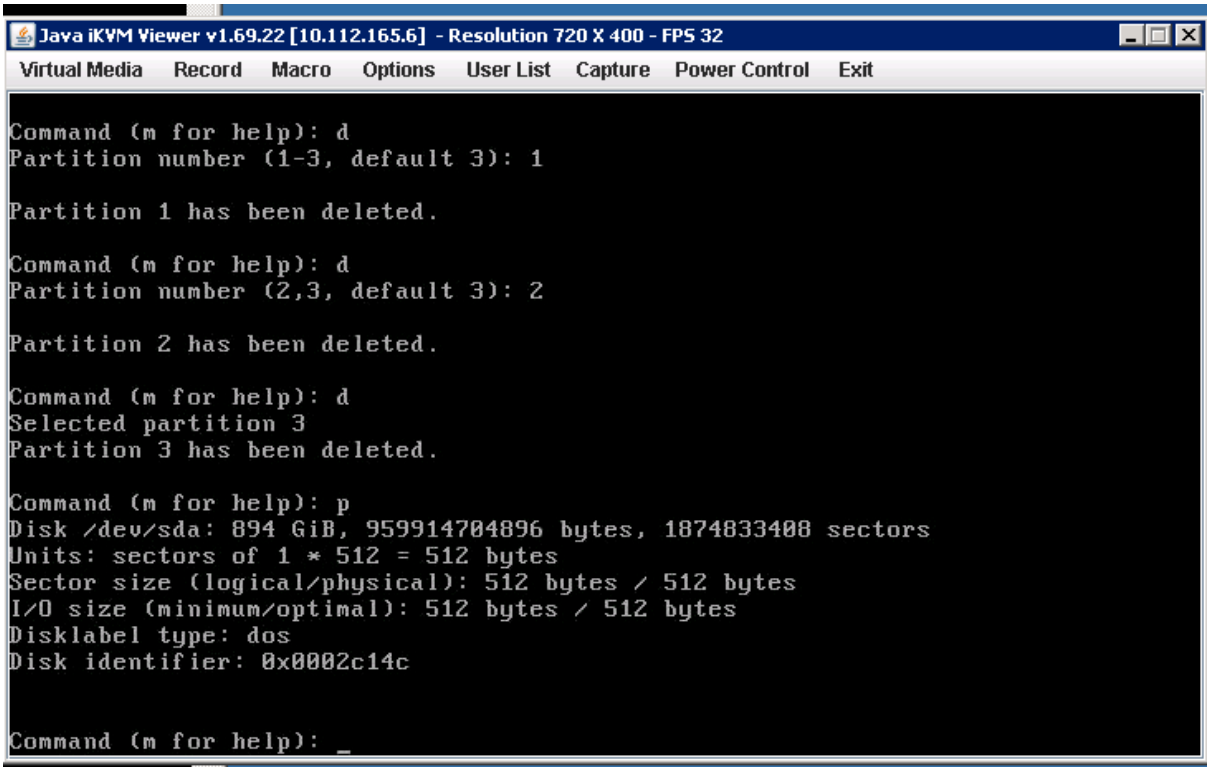
```
Java iKVM Viewer v1.69.22 [10.112.165.6] - Resolution 720 X 400 - FPS 32
Virtual Media  Record  Macro  Options  User List  Capture  Power Control  Exit
boot  dev                home  lib64  mnt    root  sbin  sys  usr
core@localhost / $ sudo fdisk -l

Disk /dev/sda: 894 GiB, 959914704896 bytes, 1874833408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0002c14c

Device     Boot      Start        End     Blocks  Id System
/dev/sda1 *           2048     526335     262144  83 Linux
/dev/sda2           526336    2623487    1048576  82 Linux swap / Solaris
/dev/sda3          2623488 1874833407 936104960  83 Linux


Disk /dev/sdb: 894 GiB, 959914704896 bytes, 1874833408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x000331b1


core@localhost / $
```

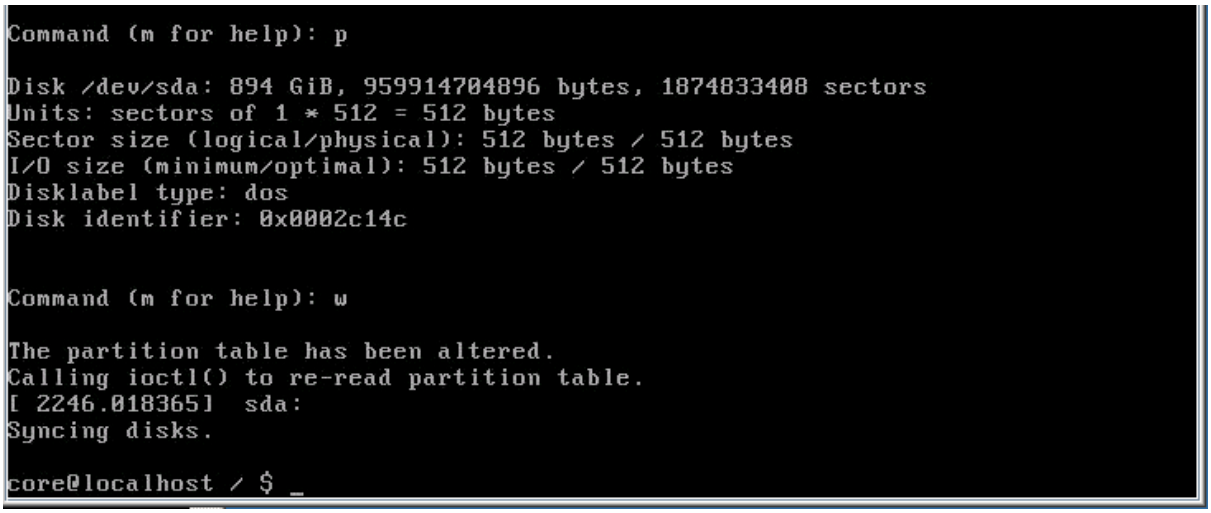
The picture above shows that disk /dev/sda has three partitions configured.

You will use the fdisk utility to clear these, run *$ sudo fdisk /dev/sda*.

```
Java iKVM Viewer v1.69.22 [10.112.165.6] - Resolution 720 X 400 - FPS 32
Virtual Media  Record  Macro  Options  User List  Capture  Power Control  Exit
Command (m for help): ^C
[ 1994.152190]  sda: sda1 sda2 sda3
core@localhost / $
core@localhost / $
core@localhost / $ sudo fdisk /dev/sda

Welcome to fdisk (util-linux 2.24.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): p
Disk /dev/sda: 894 GiB, 959914704896 bytes, 1874833408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0002c14c

Device     Boot      Start        End     Blocks  Id System
/dev/sda1 *           2048     526335     262144  83 Linux
/dev/sda2           526336    2623487    1048576  82 Linux swap / Solaris
/dev/sda3          2623488 1874833407 936104960  83 Linux

Command (m for help): _
```

You can use *p* to show the partitions, and *d* to delete the partitions. I deleted all 3 partitions I had setup, but bear in mind this will erase any data on the disk!! Make sure you use *w* to write the changes.





Now you are ready to install CoreOS!!

The following command will install the latest stable release of CoreOS to /dev/sda, using your cloud-config file. Make sure you run this from the same location as you r cloud-config file, or enter the path!

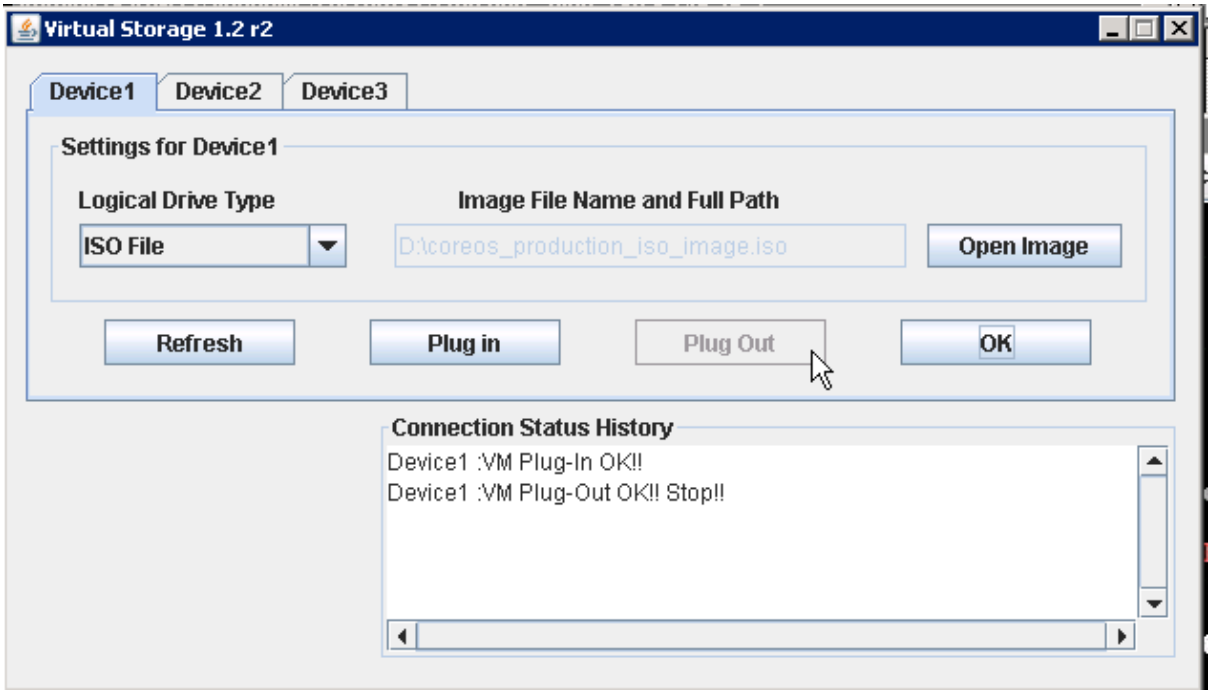
$ sudo coreos-install –d /dev/sda –C stable –c cloud-config.yaml

```
core@localhost / $ sudo coreos-install -d /dev/sda -C stable -c cloud-config.yam
l
Checking availability of "local-file"
Fetching user-data from datasource of type "local-file"
Downloading the signature for http://stable.release.core-os.net/amd64-usr/522.4.
0/coreos_production_image.bin.bz2...
2015-01-12 11:34:53 URL:http://stable.release.core-os.net/amd64-usr/522.4.0/core
os_production_image.bin.bz2.sig [543/543] -> "/tmp/coreos-install.2mfLy0eONB/cor
eos_production_image.bin.bz2.sig" [1]
Downloading, writing and verifying coreos_production_image.bin.bz2...
```

Once the installation is complete you should get a success message, if you don't get this the installation has not completed!

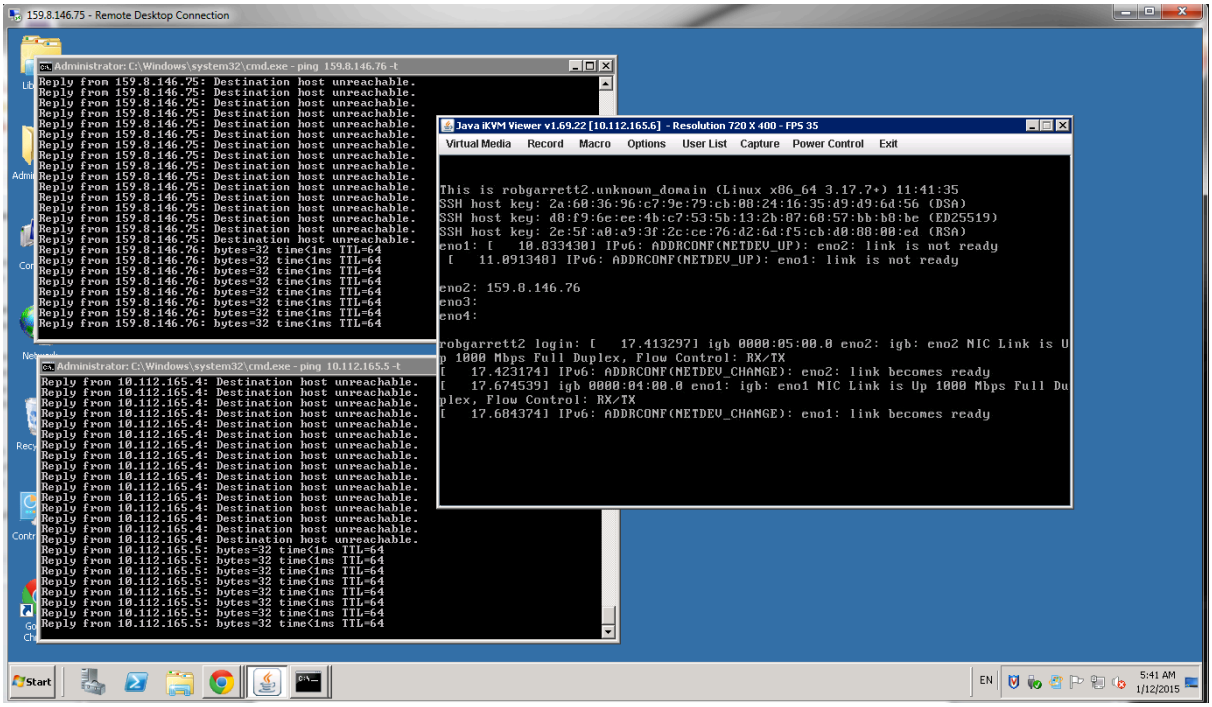
```
Java iKVM Viewer v1.69.22 [10.112.165.6] - Resolution 720 X 400 - FPS 31
Virtual Media   Record   Macro   Options   User List   Capture   Power Control   Exit
[ 2689.287463] GPT:Primary header thinks Alt. header is not at the end of the di
sk.
[ 2689.288166] GPT:9289727 != 1874833407
[ 2689.288544] GPT:Alternate GPT header not at the end of the disk.
[ 2689.288935] GPT:9289727 != 1874833407
gpg: key 93D2DCB4 marked as ultimately trusted[ 2689.289321] GPT: Use GNU Parted
 to correct GPT errors.
[ 2689.289330]  sda: sda1 sda2 sda3 sda4 sda6 sda7 sda9

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: Good signature from "CoreOS Buildbot (Offical Builds) <buildbot@coreos.com>
" [ultimate]
[ 2689.305131] GPT:Primary header thinks Alt. header is not at the end of the di
sk.
[ 2689.305804] GPT:9289727 != 1874833407
[ 2689.306180] GPT:Alternate GPT header not at the end of the disk.
[ 2689.306560] GPT:9289727 != 1874833407
[ 2689.306939] GPT: Use GNU Parted to correct GPT errors.
[ 2689.307323]  sda: sda1 sda2 sda3 sda4 sda6 sda7 sda9
Installing cloud-config...
[ 2689.821963] BTRFS info (device sda9): disk space caching is enabled
Success! CoreOS stable 522.4.0 is installed on /dev/sda
core@localhost / $
```

Before you reboot, you need to unplug the ISO image, else you will reboot back int o the live CD.



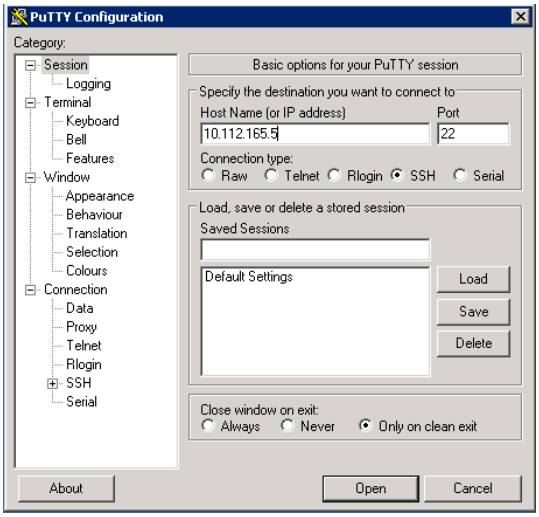
Now reboot your server again..

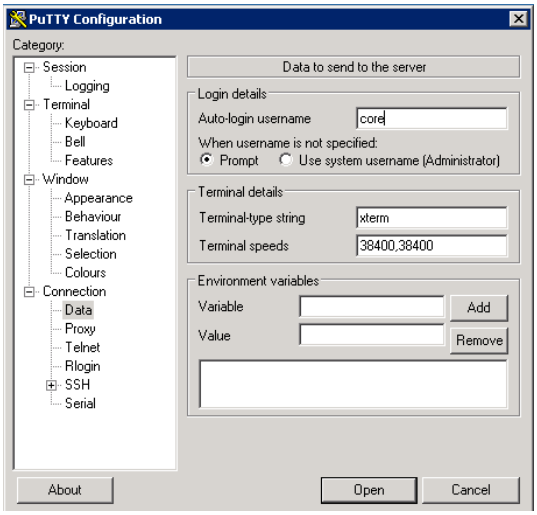
You should be presented with a CoreOD operating system, and shortly after that yo ur network interfaces should come up...



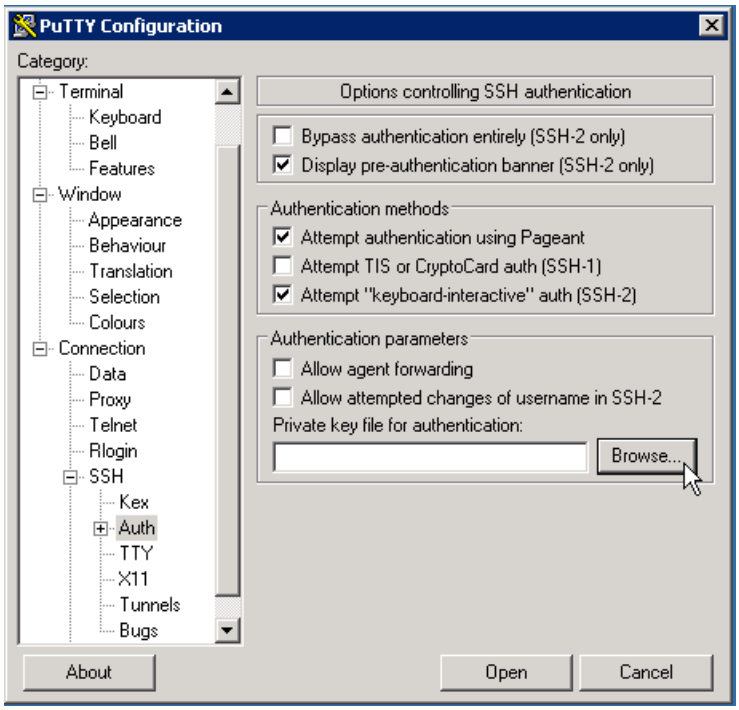
You cannot log into CoreOS from the console by default, so now you need to SSH i nto the server.

Open Putty, and enter the IP address of the CoreOS server. In *Connection/Data*, ent er the auto login username *core*.

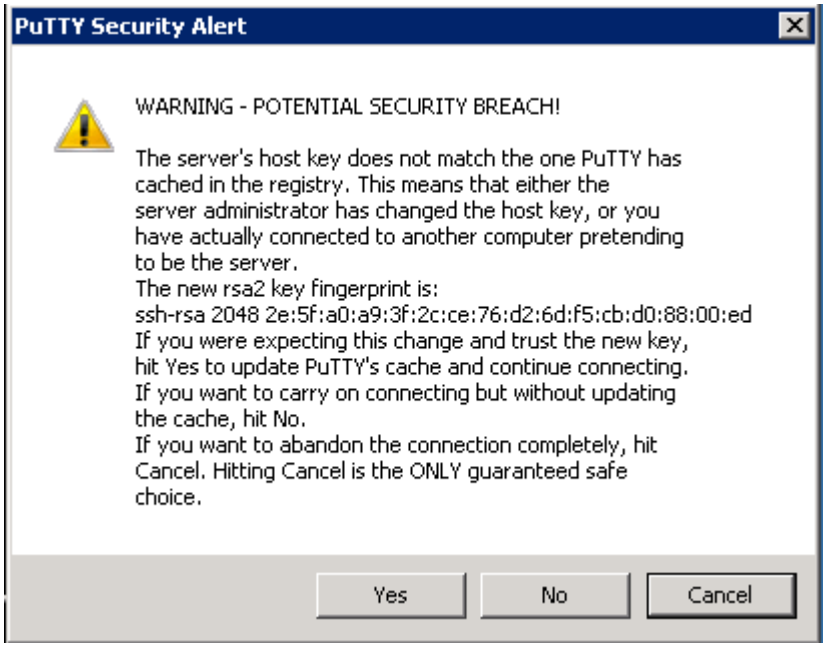




Under Connection/SSH/Auth, browse for the private SSH key you saved earlier, and open the connection.



If it is the first time you are connecting to the server, you will be prompted to update the SSH Cache.



You should be asked for the passphrase that you set.

If you get here, you have successfully installed CoreOS on a bare metal server running in SoftLayer!!

6 - OpenSUSE Load

wedwed