Cabrillo College

# CIS 192 Linux Lab Exercise

## Lab 2: Joining a network
## Fall 2011

**Lab 2: Joining a network**

The purpose of this lab is to configure the network settings of several systems to join one or more networks. This includes setting the IP address, network mask, default gateway, and DNS settings for different distributions of Linux. Once joined, the connectivity will be tested and network traffic observed.

**Supplies**

- Frodo, Elrond and William VMs (CIS Lab or VLab)

**Forum**

Use the forum to ask for help, post tips and any lessons learned when you have finished.
Forum is at: http://opus.cabrillo.edu/forum/viewforum.php?f=39

**Background**

For a Linux system to join a LAN (Local Area Network) an IP address and subnet mask must be configured on one of its NIC interfaces. The IP address can be dynamic (obtained automatically from a DHCP server) or static (manually configured IP address). To reach beyond the LAN to the Internet, a default gateway must be configured. To be able to use hostnames, e.g. google.com, rather than numerical IP address a DNS name server must be configured.

The IP and gateway configuration can be configured temporarily (lasts till the next reboot) using the ifconfig and route commands. For a more permanent solution, the IP and gateway information must be added to the appropriate network configuration files in the /etc directory. These files are used by the network service during system startup.

Finally, there are a number of commands (utilities) that can be used to check that traffic is flowing correctly on the networks.

**Procedure**

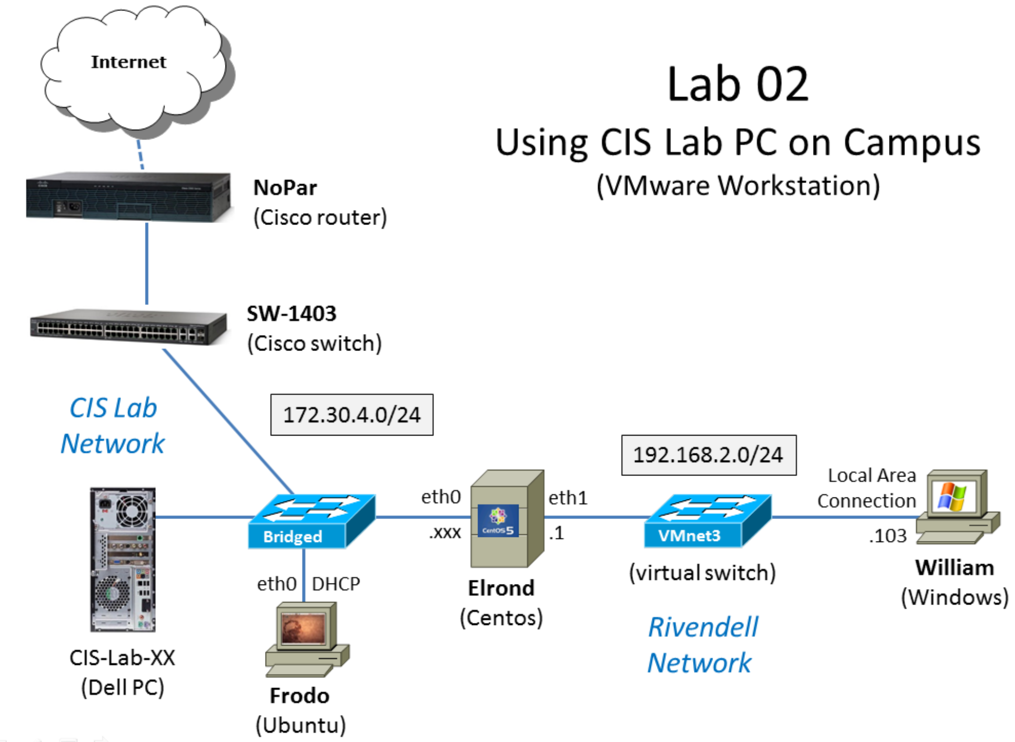
You will be cabling Frodo, Elrond and William according to the diagram below. Once cabled you will join the VMs to networks, test connectivity and observe some packet traffic.

Results will be recorded in a lab2 text file that gets submitted at the end of the lab.

# Lab 02

## Using CIS Lab PC on Campus

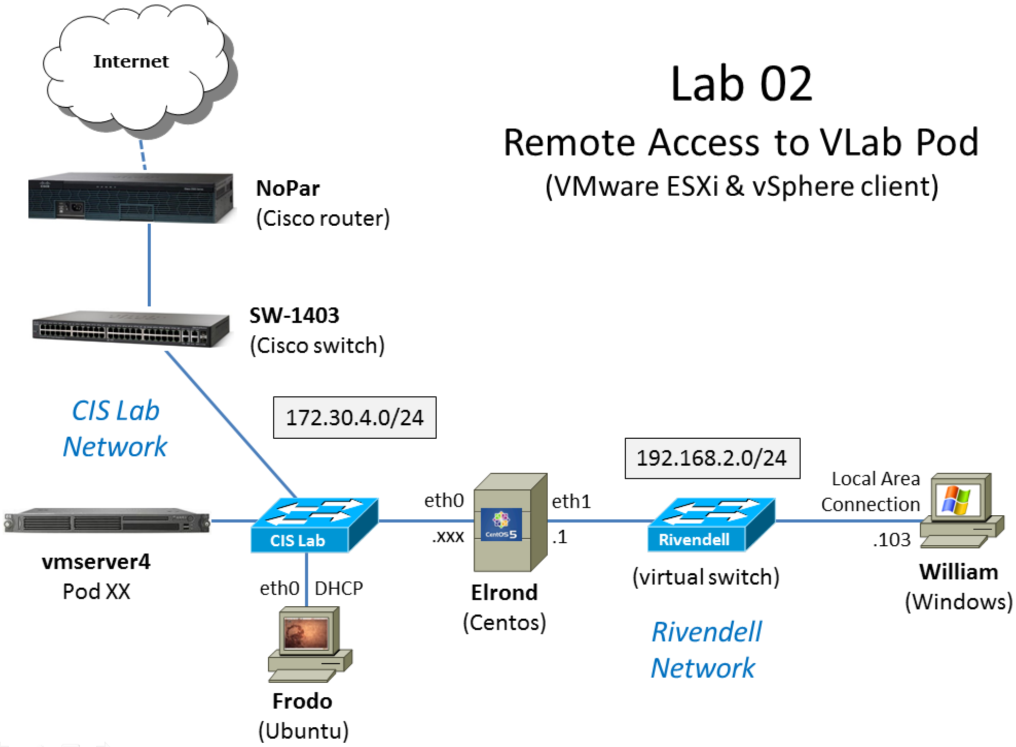
(VMware Workstation)



# Lab 02

## Remote Access to VLab Pod

(VMware ESXi & vSphere client)



**Step 1 - Preparation**

□ Revert the VMs back to their "Pristine" snapshots:

- Frodo
- Elrond
- William

□ On Opus, make a copy of the lab02 template file in /home/cis192/depot in your home directory. Edit the header of this file with your own information.

□ You may wish to print off the diagram above that is appropriate to where you do this lab. It's nice to have a network diagram handy during a lab to mark up with IP addresses and other notes.

**Step 2 - Cabling**

□ Cable each system to the networks shown in appropriate diagram above.

- Frodo
- Elrond
- William

Use the following table to help decide how to make the connections:

| | CIS Lab Network | Rivendell Network |
|---|---|---|
| VMware Workstation (CIS Lab PCs) | Bridged | VMnet3 |
| VMware ESXi (VLab VMs) | CIS Lab Network | Rivendell |

**Step 3 - Verify Frodo's connection to the network**

In this step we examine the network settings on then Ubuntu VM that gets configured using a DHCP service. The pristine version of Frodo has been preconfigured to connect to the network using DHCP. Snickers, a Windows server in the 2501 closet, provides a DHCP service for room 2501 and the CIS Lab in the CTC. The network settings (IP address, mask, gateway and name server) get configured automatically during system startup.

□ Power up Frodo, which is an Ubuntu VM. The goal of Ubuntu is to create an easy-to-use Linux desktop. The Ubuntu distribution is based on Debian.

□ Login as cis192 and then **sudo -i** to root.

□ Use **ping -c 3 google.com** to verify we have an IP address, subnet mask, gateway and name server configured. The ping should work.

□ Examine the NIC hardware and driver on Frodo:

a. Use **lspci** and locate your NIC hardware.
b. Try **lspci | grep -i ether** instead.
c. Use **lspci -k** and locate your NIC driver.
d. Try **lspci -k | grep -iA4 ether** to see your NIC driver.
e. Use **lsmod** and locate your driver in the output.

□ Examine the interface:

a. Issue the **ifconfig** command and check that eth0 and lo are up and running.
b. lo is the loopback interface to access local services without having to go out on the network.
c. For eth0, make sure you can identify the MAC address (HWaddr), IPv4 address (inet addr), IPv6 address (inet6 addr) subnet mask (Mask) and broadcast address (Bcast).

□ Examine the routing table:

    a. Use **route -n** and look for an entry with 172.30.4.1 under the Gateway column and a “G” under the Flags column. That is the default gateway.

□ Examine the domain name system (DNS) servers:

    a. Use **cat /etc/resolv.conf** to show the name servers.

❖ Record the Frodo network information in your lab02 file.

**Step 4 - Frodo trouble-shooting**
In this step you are going to break Frodo then fix Frodo.

□ Use the **rmmod** command to remove the NIC driver.
□ Verify that you can no longer ping anything.
□ Without restarting Frodo or reverting to a snapshot see if you can fix Frodo.
□ Verify that you can ping google.com again.

❖ Record in your lab02 file what you did to repair Frodo.

**Step 5 - Configure Elrond temporarily with static IP addresses**
In this step you will be configuring both of Elrond’s interfaces with Static IP addresses to match the diagram above.

□ Power up Elrond which is running the CentOS Linux distribution. CentOS is a replica distribution of the Red Hat Enterprise Linux distribution.

    a. Elrond has two NICs so it can be configured as a router in later labs.
    b. Note that Elrond is configured to boot up to run level 3 (no graphics).

□ Log in as cis192, then **su -** to root.
□ Clear out any existing name servers with **> /etc/resolv.conf**
□ Try to ping google.com, does it work? It shouldn’t.
□ Try to ping Opus at 207.62.186.9, does it work? It shouldn’t.

You shouldn’t be able to ping anything because Elrond is configured to boot up without any network settings.

□ IMPORTANT! Use http://simms-teach.com/docs/static-ip-addrs.pdf to lookup a **non-duplicate** static IP address for Elrond’s eth0 interface. This should be based on your station or pod number!
□ Use **ifconfig eth0 172.30.4.*xxx* netmask 255.255.255.0** to configure eth0 with that IP address. This is Elrond’s interface to the CIS Lab network. This network is in use by other CIS students.
□ Check local network connectivity:

    a. Use **ifconfig eth0** to check eth0 is up with your IP address.
    b. Use **ping 172.30.4.1** to ping the lab router. It should work now.
    c. Make sure you can ping Frodo too.

□ Check Internet connectivity:

    a. Can you **ping 207.62.186.9** now? What error do you get?
    b. We need to specify a router that we can send packets destined for the Internet. This will be our default gateway. Set the default gateway with **route add default gw 172.30.4.1** and use **route -n** to verify. Try pinging 207.62.186.9 again. It should work now.

□ Check name resolution for the internet:

a. Can you **ping opus.cabrillo.edu** now? What error message did you get? Without a name server configured the ping command cannot resolve hostnames to IP addresses. Without an IP address, packets cannot be sent to the Internet.
b. Edit the /etc/resolv.conf file and add the line **nameserver 192.168.0.8**
c. Retry pinging opus.cabrillo.edu. This should succeed now that a DNS name server has been configured.
d. You should be able to **ping google.com** as well. Congratulations ... you just configured your first interface on Linux!

□ Configure Elrond's second interface.

a. Look at the appropriate diagram above and determine what IP address *xxx.xxx.xxx.xxx* should be set on eth1. This is Elrond's interface to the Rivendell network.
b. Use **ifconfig eth1** _xxx.xxx.xxx.xxx_***/24*** to set that IP address on eth1. It's OK to assign the same IP address to this interface that other students are using on their station or pod. Because no one else can get to your Rivendell network you don't have to worry about having a duplicate IP address.
c. Use **ifconfig | more** to check both eth0 and eth1 interfaces are alive and well and using the static IP address you assigned.

**Step 6 - Configure Williams Local Area Network interface**

In this step you will be configuring William's network settings to match the diagram above.

□ Power up William which is running Windows XP.
□ Login as cis 192.
□ Right click on My Network Places and select Properties.
□ Right click on Local Area Connection and select Properties.
□ Select Internet Protocol (TCP/IP) and click Properties button.
□ Select "Use the following IP address:"
□ Set the IP address to 192.168.2.103
□ Set the subnet mask to 255.255.255.0
□ Don't worry about setting up the default gateway or the DNS name server. Until we learn how to configure routing on Elrond, William will not be able to get out to the Internet. We will learn how to do that in the next lab!
□ Click OK and close any open dialog boxes.
□ Start > Run... > cmd to get a command prompt.
□ Use **ipconfig** to check your static IP was set correctly.
□ Ping Elrond using **ping 192.168.2.1** which should work.
□ Try pinging Frodo or the CIS Lab router (172.30.4.1) which should not work.

**Step 7 - Traversing the VM's using Putty and SSH**

Even though we don't have routes set up between the CIS Lab and Rivendell networks we can still use Putty/SSH to traverse from system to system. Starting with Putty on William you can Putty to Elrond. From there you can ssh to Frodo, and from Frodo you can ssh to opus.cabrillo.edu.

□ On William, run Putty and login as cis192 on Elrond. What IP address should you use for this? This should be the IP address you put on Elrond's eth1 interface.
□ Still in Putty, from Elrond, use **ssh cis192@172.30.4.**_xxx_ to login to Frodo. Use **ifconfig eth0** on Frodo if you need to remember Frodo's IP address.

□ Still in Putty, from Frodo, use **ssh** *xxxyyy192***@opus.cabrillo.edu** to login to Opus using your student account.

□ Use **who** on Opus to see if anyone else you know is logged on.

□ Can you ping Frodo from Opus? This should not work and we will learn why in later lessons when we cover NAT/PAT and firewalls.

□ Use **exit** to log off Opus.

□ Use **exit** to log off Frodo.

□ Use **exit** to log off Elrond.

**Step 8 - Permanently add static IP addresses on Elrond**

We set static IP addresses earlier on Elrond using the ifconfig command. This instantly added the IP addresses to the interfaces. If the system is restarted this configuration information, which is kept in memory, will be lost. In this step we will permanently configure the network settings which is done by editing network configuration files in the /etc directory.

□ Restart Elrond using **init 6**

□ Can you ping the Lab router or google.com? You shouldn't be able to because the network configuration, kept in memory, was lost during the system restart.

□ Edit the **/etc/sysconfig/network-scripts/ifcfg-eth0** file and add the same non-duplicate static IP address you used previously. Use the appendix below for help on this.

□ Edit the **/etc/sysconfig/network-scripts/ifcfg-eth1** file and add the same static IP address you used previously.

□ Edit the **/etc/sysconfig/network** file and add the default gateway. Use the appendix below for help on this.

□ Verify the settings you made in **/etc/resolv.conf** are still there.

□ Use **service network restart** to restart the network service which will read and apply the information in the network configuration files. The same thing will be done after a system restart.

□ Use **ifconfig | more** and **route -n** to verify your settings.

□ Make sure you can ping the William and google.com.

□ To prove the settings are permanent, use **init 6** to restart the system.

❖ Record in your lab02 file the four Elrond primary network configuration files and output from ifconfig and route -n. Tip: use redirection to capture the output into a file and use scp to copy that file to your Opus home directory:

- **cat /etc/sysconfig/network-scripts/ifcfg-eth* > notes**
- **cat /etc/sysconfig/network >> notes**
- **cat /etc/resolv.conf >> notes**
- **ifconfig >> notes**
- **route -n >> notes**
- **scp notes** *xxxyyy192***@opus.cabrillo.edu:**

**Step 9 - Testing with ping**

Do some initial connectivity checks using echo requests (pings):

□ On Elrond,

- Make sure you can ping the loopback address (127.0.0.1), Frodo, William, the CIS Lab router (172.30.4.1) and google.com.

□ On Frodo,
  - Make sure you can ping the loopback address (127.0.0.1), Elrond, the CIS Lab router and google.com.
  - Frodo should not be able to ping William or Elrond's eth1 interface. When pinging these Rivendell addresses you will not get an error message. Why not? Because the pings are heading out (the wrong way) to the Internet with no route back. We will learn how to configure routes in Lab 03.

□ On William,
  - Make sure you can ping Elrond. William should not be able to ping anything else.

**Step 10 - Install some additional commands**

We need to install some more commands to do this step:

□ On Frodo,
  - Use **apt-get install traceroute** to install that command.

□ On Elrond,
  - Use **yum install mtr traceroute tcpdump** to install those commands.

**Step 11 - Observe where packets go**

The **mtr** and **traceroute** commands let you view the route a packet takes. For this lab we will just observer where they go. In future labs we will control where they go.

□ On Frodo,
  - Use **mtr opus.cabrillo.edu** to see the route packets take to Opus.
  - Use **traceroute -I opus.cabrillo.edu** for an alternate view. The -I option on traceroute is to get past the Cabrillo firewalls blocking requests to UDP ports.
  - Use **mtr google.com** to see the route pings take travelling to google.com.
  - Use **traceroute google.com** for an alternate view.
  - Use **mtr** and **traceroute** with William's IP address (192.168.2.103) to observe why pings never get there.

**Step 12 - Sniffing packets with tcpdump**

**tcpdump** is like a command-line version of Wireshark. It allows you to capture packets and view them.

□ On Elrond,
  - Use **tcpdump -n icmp or arp** to sniff only icmp and arp traffic. The -n prevents doing DNS lookups on IP address to get the hostnames.
  - Use **Ctrl-s and Ctrl-q** to pause and continue

□ On Frodo,
  - Empty Frodo's arp cache with **ip neigh flush all**
  - Ping Elrond and observe the arp request, echo request and echo reply.

❖ Record in your lab02 file the arp and icmp packets you captured on Elrond when pinging from Frodo. Tip: use redirection to capture the output into a file and use scp to copy that file to your Opus home directory:
  - **tcpdump -n icmp or arp > capture**
  - **scp capture *xxxyyy192*@opus.cabrillo.edu:**

**To turn in**

When finished with this lab, update your total “TBA hours” in your lab02 file. This should reflect the total number of hours you spent preparing for and getting this lab done.

Check your work for completeness then submit as many times as you wish up until the due date deadline. Remember, **late work is not accepted**, so start early, plan ahead for things to go wrong and use the forum to ask questions.

```
cp lab02 /home/rsimms/turnin/lab02.$LOGNAME
```

**Grading rubric (30 points)**

1 points for a correct submittal into the turnin directory
2 points for a professional quality lab write-up that can be read using vi
2 points for a complete header, including the amount of time you spent on this lab
5 points for correct Frodo network information
5 points for a correct description of how to repair Frodo
5 points for correct Elrond network configuration
5 points for correct Elrond ifconfig and route output
5 points for correct arp and icmp packet capture

**Extra Credit (1 point each)**

1) Install arpwatch on Elrond and collect at least 10 IP/MAC pairs. Copy your arp.dat file contents to your lab02 file.
2) From Frodo, ssh to Elrond using IPv6. Copy the who command output on Elrond to your lab02 file showing the IPv6 address for your session.
3) Install Wireshark on Frodo and capture IPv6 pings to Elrond. Be sure to run Wireshark as root. Export the captured packets as text and include one ICMPv6 request and reply to your lab06 file.
4) Explain in your lab02 file why Frodo cannot ping William.
5) What is the IPv6 address for Opus? Add this to your lab02 file.

**Appendix**

| **General Linux commands** | |
|---|---|
| **su -** | To become root (superuser). The - is very important as it provides root’s shell environment. |
| **sudo su -**<br>or **sudo -i** | To become root on the Ubuntu VMs. |
| **cp** *source destination* | Linux command to copy file(s) from the source pathname to the destination pathname.<br><br>Example:<br>**cp /home/cis192/depot/lab01 .**<br>will copy the file named lab01 in the /home/cis192/depot directory to your current directory. |

| | |
|---|---|
| **vi** *pathname* | Run the vi text editor on the specified file.<br>Example: **vi lab01** |
| **who** | Show logged in users and the IP address or hostnames they logged in from. |
| **echo $PATH** | Shows your path.  The shell uses the path to locate any commands entered. Entering a command that is not located on the path will result in a "command not found" error. |
| **cat /etc/*-release** | Shows the name of the Linux distribution being run. |
| **>** *filename* | *filename* is created if it does not exist and emptied.<br>Example: **> output**<br>would empty the file named output or create it if it did not exist already. |
| *command* **>** *filename* | *filename* is emptied, then the output of the command is redirected into *filename.*<br>Example: **ifconfig > output**<br>would save the output of the ifconfig command in a file named output. |
| *command* **>>** *filename* | Output of the command is appended to the end of *filename.*<br>Example: **route -n >> output**<br>would append the routing table to the end of the file named output. |
| **ssh** *account***@***hostname*<br>**ssh** *account***@***xxx.xxx.xxx.xxx* | Login to a remote Linux computer on the network.<br>Example:<br>**ssh cis192@172.30.4.153** |
| **ssh** *account***@***hostname* **'***command***'** | Run a command on a remote system.<br>Example:<br>**ssh root@172.30.4.164 'ifconfig'**<br>would run the ifconfig command on the remote system and show the output of the command on the local system. |
| **ssh** *account***@***IPv6address***%eth***n* | **ssh** works with IPv6 addresses too but the outgoing interface being specified.<br>**ssh cis192@fe80::20c:29ff:fe2a:5717&eth0**<br>(all on one line) |
| **scp** *pathname account***@***host***:***pathname* | Copy files from one system to another. |

| | |
|---|---|
| **scp** *account@host*:*pathname pathname* | Example:<br>**scp output simben192@opus.cabrillo.edu:**<br>(above all on one line)<br>would copy the local file named output to the user simben192's home directory on Opus. |
| **hostname** | Shows the hostname of the system being used. |
| **tty** | Shows the current terminal being used. |
| **exit** | End a terminal login session |
| **init 0** | A fast way to gracefully shutdown a VM. You must be the root user to perform this command. Note: no warning is given to users that the system will be shut down. |
| **yum provides** *command* | Find the package containing the command or program to install.<br><br>Example:<br>**yum provides mail** |
| **yum install** *package* | Download and install the software package on Red Hat family distributions. Just specify the name of the package to get the correct version for your distribution.<br>Examples:<br>**yum install traceroute**<br>**yum install mtr tcpdump**<br>**yum install mailx** |
| **apt-get install** *package* | Download and install the software package on Debian family distributions. Just specify the name of the package to get the correct version for your distribution.<br>Examples:<br>**apt-get install traceroute**<br>**apt-get install mtr tcpdump**<br>**apt-get install wireshark ipcalc** |
| **VMware commands and operations** | |
| On PC Keyboard:<br>Method 1: While holding down the **Ctrl-Alt** keys, tap **spacebar** then tap f1, f2, ... or f7.<br><br>Method 2: While holding down **Alt** key, tap f1, f2, ... or f7. Does not always work but simpler than method 1. | Change to a different virtual terminal on the VM.  F7 is graphics mode for the Ubuntu VMs.  The Centos VMs do not have a graphics mode (init level 3 only)<br><br>Note: the spacebar does not need to be tapped on a physical (non-VM) system. |

| | |
|---|---|
| On Mac keyboard:<br>Hold down **Control** and **Option** keys, tap the **spacebar**, hold down **fn** key (in addition to **Control** and **Option** keys) and tap f1, f2, ... or f7. | This is just required for changing virtual terminals on VMware VMs. |
| **Linux network or network-related commands** | |
| **dhclient eth0** | Obtain an IP address for the eth0 interface from a DHCP server. |
| **dhclient -r** | Release the IP address previously obtained. |
| **ifconfig** or **/sbin/ifconfig** | Show the interface configurations.<br>The full absolute pathname may be required if user is not logged in as root and /sbin is not in the user's path. |
| **ifconfig eth*n***<br>(where *n* is the interface number) | Show settings for selected interface.<br>Example: **ifconfig eth1** will show information on the eth1 interface. |
| **ifconfig eth*n* down**<br>(where *n* is the interface number) | Bring an interface down<br>Example: **ifconfig eth1 down** will disable the eth1 interface. |
| **ifconfig eth*n* up**<br>(where *n* is the interface number) | Bring an interface down<br>Example: **ifconfig eth1 up** will enable the eth1 interface. |
| **route -n** | Show the current routing table.  The -n (numerical) option makes it faster. This option disables DNS lookups to replace IP addresses with hostnames in the output. |
| **cat /etc/resolv.conf** | Show the DNS servers to use for resolving hostnames to IP addresses. |
| **ping** *hostname*<br>**ping** *xxx.xxx.xxx.xxx* | Test connectivity with another computer on the network. Use **Ctrl-C** to stop pinging.<br>The c option can be used to limit the number of pings.<br>The b option can be used to ping a broadcast address.<br>Example **ping -c3 google.com** will ping |

| | |
|---|---|
| | Google three times then stop. |
| **ping6 -I eth***n IPv6-address* | Works like the IPv4 ping except the outgoing interface must be specified.<br>Example:<br>**ping6 -I eth0 fe80::20c:29ff:fe2a:5717** |
| **mtr** *hostname*<br>**or mtr** *xxx.xxx.xxx.xxx*<br>Use q to quit | Displays the full route to the host and will refresh travels times. |
| **traceroute** *hostname*<br>**or traceroute** *xxx.xxx.xxx.xxx*<br>Use q to quit | Displays the full route to the host and will refresh travels times. |
| **arp -n** | Display arp cache |
| **ip neigh flush all** | Flush arp cache |
| **tcpdump** | Will start sniffing packets.<br>http://www.alexonlinux.com/tcpdump-for-dummies |
| **tcpdump -n arp or icmp**<br>Use **-n** to prevent DNS lookups<br>Use **Ctrl-s** or **Ctrl-q** to pause and continue | Packet sniffing command to capture only arp and icmp packets |
| **tcpdump -n host** *xxx.xxx.xxx.xxx* **and host** *xxx.xxx.xxx.xxx*<br>(all on one line)<br>Use **-n** to prevent DNS lookups<br>Use **Ctrl-s** or **Ctrl-q** to pause and continue | Packet sniffing command to capture only traffic between two hosts.<br>Example:<br>tcpdump -n host 172.30.4.25 and host 172.30.4.1<br>(all on one line) |
| **tcpdump -ne -i eth***n* **port** *nn* **or port** *nn* | Example:<br>**tcpdump -ne -i eth1 port 80 or port 22**<br>• no DNS lookups (-n)<br>• shows mac addresses (-e)<br>• will listen on eth1 interface (-i eth1)<br>• only captures ssh and http traffic (port 80 or 22) |
| **Interface configuration (temporary, till next restart)** | |
| **ifconfig eth***n xxx.xxx.xxx.xxx* **netmask** *xxx.xxx.xxx.xxx*<br>where *n* is the interface number and *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address or netmask | Configure an interface with an IP address and subnet mask.<br>Example:<br>**ifconfig eth0 172.30.4.149 netmask** |

| | |
|---|---|
| | **255.255.255.0**<br>(all on one line)<br>Would configure eth0 with that IP address and mask. |
| **ifconfig eth***n* *xxx.xxx.xxx.xxx***/***pp*<br><br>where *n* is the interface number, *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address and *pp* is the network prefix | Same as previous command but the subnet mask is specified instead using a CIDR network prefix.<br><br>Example: **ifconfig eth0 172.30.4.149/24** |
| **ip address flush dev eth***n*<br>(where *n* is the interface number) | Removes all settings from the selected interface.<br><br>Example: **ip address flush dev eth0** will remove all interface settings, including the IP address, from eth0. |
| **Red Hat interface configuration (permanent)** | |
| Edit **/etc/sysconfig/network-scripts/ifcfg-eth***n* and add or modify these lines:<br><br>**NM_CONTROLLED="***xx***"**<br>**ONBOOT="***xx***"**<br>**BOOTPROTO="***xx***"**<br>**IPADDR=** *xxx.xxx.xxx.xxx*<br>**NETMASK=** *xxx.xxx.xxx.xxx*<br><br>Example 1 - eth0 is not configured:<br>**/etc/sysconfig/network-scripts/ifcfg-eth0**<br>**DEVICE="eth0"**<br>**NM_CONTROLLED="yes"**<br>**ONBOOT="no"**<br><br>Example2 - eth0 has a static IP configured:<br>**/etc/sysconfig/network-scripts/ifcfg-eth0**<br>**DEVICE="eth0"**<br>**NM_CONTROLLED="no"**<br>**ONBOOT="yes"**<br>**BOOTPROTO=”static”**<br>**IPADDR=172.30.4.149**<br>**NETMASK=255.255.255.0**<br><br>Example 3 - eth0 is configured for DHCP:<br>**/etc/sysconfig/network-scripts/ifcfg-eth0**<br>**DEVICE="eth1"**<br>**NM_CONTROLLED="no"**<br>**ONBOOT="yes"**<br>**BOOTPROTO=”dhcp”**<br><br>For the new interface settings to take effect without | Each interface has an associated ifcfg-**eth***n* file in the **/etc/sysconfig/network-scripts** directory.<br><br>These files are used at system startup to configure the interfaces.<br><br>Set NM_CONTROLLED to “yes” or “no” to use or not use Red Hat NetworkManager utility.<br><br>Set ONBOOT to “yes” to bring up the interface or “no” to disable the interface at system startup.<br><br>Set BOOTPROTO to “static” to configure a static IP address or “dhcp” to configure a dynamic IP address.<br><br>For static IP addresses, set IPADDR to the static IP address. Be sure this is a unique IP address for your system to avoid duplicate IPs on the network! Set NETMASK to the subnet mask. |

| | |
|---|---|
| restarting the system, use:<br>**service network restart**<br>or **/etc/init.d/network restart** | |
| **Routing table configuration (temporary, till next restart)** | |
| **route add default gw** *xxx.xxx.xxx.xxx*<br><br>where *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address for the default gateway router. | Adds the default gateway to the routing table.  Unless there is another more specific route in the routing table this is the route will be used to send outbound packets.<br><br>Example:<br>**route add default gw 172.30.4.1**<br>adds the lab router as the default gateway. |
| **route del default gw** *xxx.xxx.xxx.xxx*<br><br>where *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address for the default gateway router. | Deletes the default gateway in the routing table.<br><br>Example:<br>**route del default gw 172.30.4.1**<br>deletes the lab router as the default gateway. |
| **Red Hat routing table configuration (permanent)** | |
| Edit (e.g. with **vi**) the **/etc/sysconfig/network** file and add or modify:<br><br>**GATEWAY=** *xxx.xxx.xxx.xxx*<br><br>where *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address for a router.<br><br>Example:<br>**/etc/sysconfig/network**<br>**NETWORKING=yes**<br>**HOSTNAME=elrond.localdomain**<br>**GATEWAY=172.30.4.1**<br>The default gateway on Elrond has been set to the CIS Lab router (172.30.4.1).<br><br>For the new interface settings to take effect without restarting the system, use:<br>**service network restart**<br>or **/etc/init.d/network restart** | Edit this file to add a permanent default gateway to the routing table.  The new settings do not take effect until the system or network service is restarted. |
| **Specify DNS name servers (permanent)** | |

| | |
|---|---|
| Edit (e.g. with **vi**) the **/etc/resolv.conf** file and add one or more<br>**nameserver** *xxx.xxx.xxx.xxx*<br>lines to specify DNS servers for name resolution services.<br><br>where *xxx.xxx.xxx.xxx* is the dotted decimal form of the IP address for a DNS name server. | Use this file to specify one or more DNS server. The first server listed will be the primary name server. The second will be the secondary name server and so forth.<br><br>Example:<br>**/etc/resolv.conf**<br>**nameserver 192.168.0.8**<br>**nameserver 10.240.1.2**<br><br>configures the CIS VLab DNS server (192.168.0.8) as the primary and the campus DNS server (10.240.1.2) as the secondary. |
| **> /etc/resolv.conf** | Clears all DNS name servers |
| **Linux hardware and driver commands** | |
| **echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts** | Enables Linux system to respond to broadcast pings. |
| **lspci**<br><br>or **/sbin/lspci** | Shows PCI devices including what NIC or NICs (Network Interface Controllers) are being used to physically connect the system to the network.<br><br>The full absolute pathname may be required if user is not logged in as root and /sbin is not in the user's path.<br><br>Example:<br>**lspci \| grep -i ether**<br>will show all the ethernet NICs on the system. |
| **lspci -k** | Show the drivers kernel modules used by the PCI devices including any NICs.<br><br>Example:<br>**lspci -k \| grep -iA4 ether**<br>will show the drivers used by the NICs on your system. |
| **lsmod**<br><br>or **/sbin/lsmod** | Shows the kernel modules that are currently loaded. Example NIC drivers (implemented as kernel modules) are e100 (Intel), e1000 (Intel), pcnet32 (AMD) and vmxnet (VMware).<br><br>The full absolute pathname may be required if user is not logged in as root |

| | |
|---|---|
| | and /sbin is not in the user's path. |
| **rmmod** *module* | Use to unload (remove) a running kernel module (e.g. a NIC driver).<br>Example: **rmmod e1000** would unload the Intel gigabit NIC driver if it was loaded. |
| **modprobe** *module* | Use to load a kernel module (e.g. NIC driver).<br>Example: **modprobe e1000** would load the Intel gigabit NIC driver if not loaded already. |