Lesson Module Checklist

- Slides
- Whiteboard with 1st minute quiz

- Flashcards
- Web Calendar summary
- Web book pages
- Commands
- Howtos

- Practice/real test
- Lab tested
- Opus - lab template in depot
- Youtube Videos, if any, uploaded

- VMs (VLab) - extra gondor and arnor switches made for each pod

- Backup slides, Confer links, handouts on flash drive
- 9V backup battery for microphone

Course history and credits

Jim Griffin

- Jim created the original version of this course
- Jim's site: http://cabrillo.edu/~jgriffin/

Rick Graziani

- Thanks to Rick Graziani for the use of some of his great network slides
- Rick's site: http://cabrillo.edu/~rgraziani/

CIS 192 - Lesson 4

Instructor: **Rich Simms**
Dial-in: **888-450-4821**
Passcode: **761867**


Solomon


Sean C.


Chris


Corey


Bryan


Sean F.


Tony


David


Donna

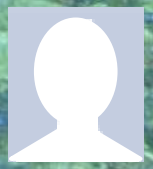

Dave


Evan


Gabriel


Elia


Tajvia


Carlos


Adam

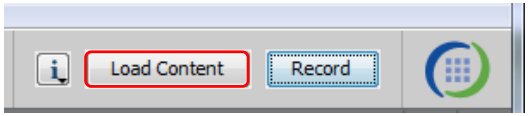

Ben


Laura

For tonight please have Frodo and Celebrian powered up

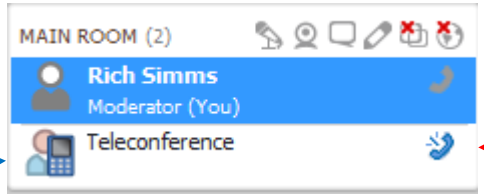
Email me (risimms@cabrillo.edu) a relatively current photo of your face for 3 points extra credit

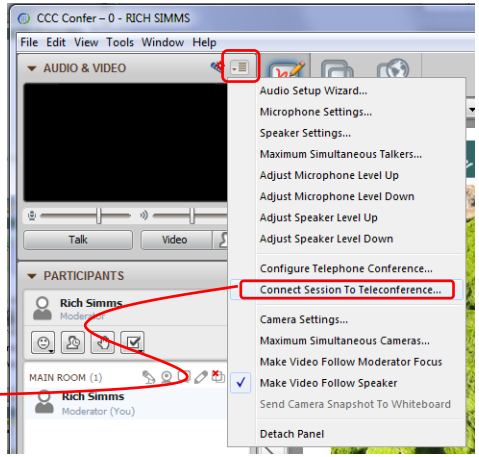
[ ] Preload White Board with *cis*lesson??*-WB*



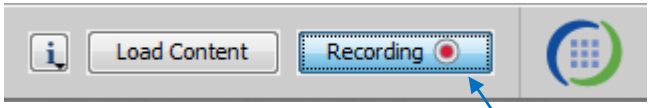
[ ] Connect session to Teleconference

Session now connected to teleconference





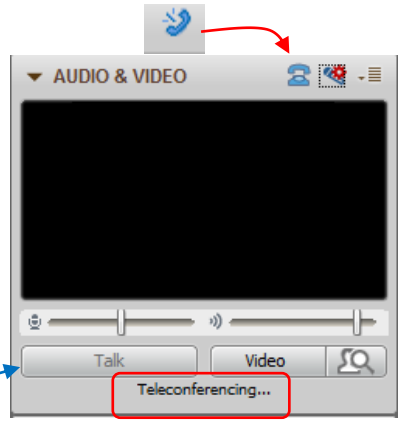
[ ] Is recording on?



Red dot means recording

[ ] Use teleconferencing, not mic

Should be greyed out

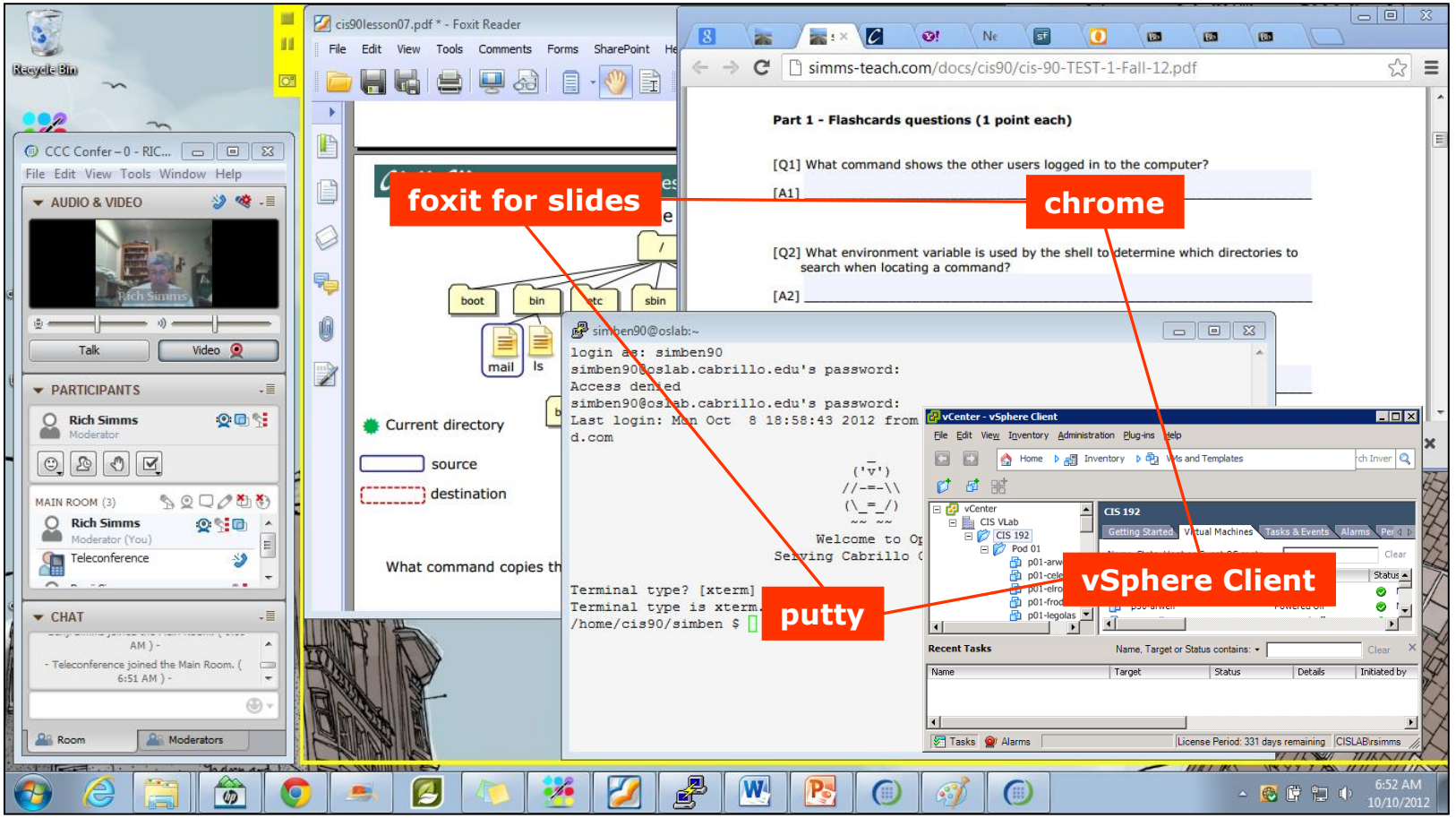


CCC Confer

Don't Forget

[ ] **Video (webcam) optional**

[ ] **layout and share apps**

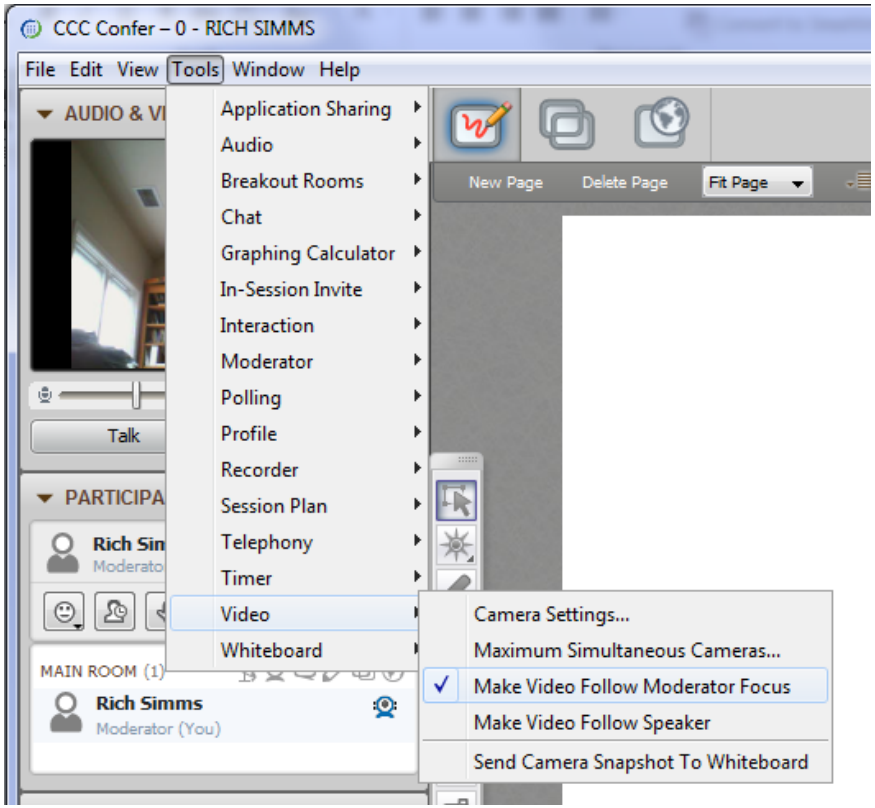


CCC Confer

[ ] Video (webcam) optional

[ ] Follow moderator

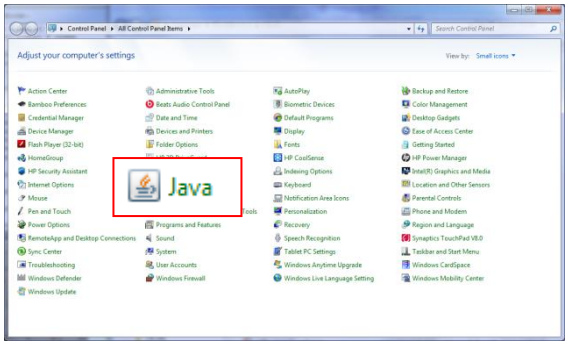
[ ] Double-click on postages stamps



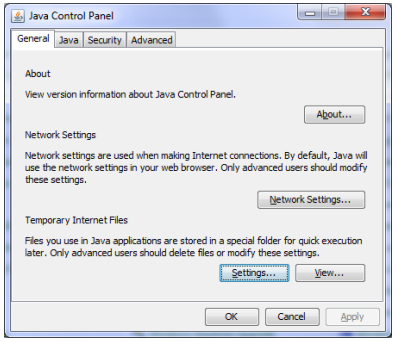
Universal Fix for CCC Confer:

1) Shrink (500 MB) and delete Java cache
2) Uninstall and reinstall latest Java runtime

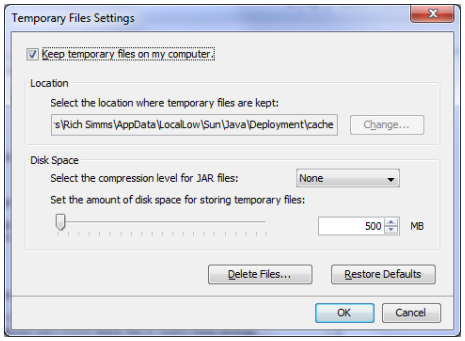
Control Panel (small icons)



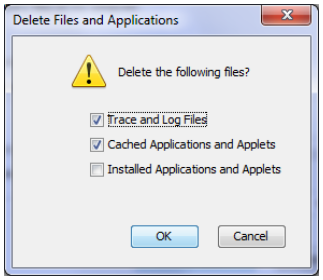
General Tab > Settings…



500MB cache size



Delete these



Google Java download

First Minute Quiz

Please answer these questions **in the order** shown:

Use CCC Confer White Board

For credit email answers to:
risimms@cabrillo.edu
within the first few minutes of class

Routing Continued and Transport Protocols

Objectives	Agenda
• Configure appropriate IP addresses, network and subnet masks, and broadcast addresses based on the size and number of network segments required. • Connect multiple network segments together using Linux servers as routers and configuring the appropriate routing tables. • Use a network sniffer to analyze network traffic between two hosts. • Identify, isolate, and correct malfunctions in a computer network. • Define the term 'socket' and describe its importance to the transport layer of the protocol stack.	• Quiz • Questions on previous material • Housekeeping • Virtual/Physical corner • Dynamic Routing • Quagga routing suite for Linux • Skills for doing Lab 4 • Transport Layer • TDP and UDP protocols • Service ports and sockets • Prepping for the test next week • Wrap

Questions on previous material

Questions

Lesson material?

Labs?

How this course works?

Chinese Proverb	他問一個問題，五分鐘是個傻子，他不問一個問題仍然是一個傻瓜永遠。 *He who asks a question is a fool for five minutes; he who does not ask a question remains a fool forever.*

Housekeeping

- Lab 3 due 11:59PM tonight
- Five posts due 11:59PM tonight

Perkins/VTEA Survey

Carl D. Perkins Career and Technical Education Act

POSTREPLY | Search this topic... | Search

Carl D. Perkins Career and Technical Education Act

by **Rich Simms** » Fri Mar 01, 2013 8:08 pm

The Carl D. Perkins Vocational and Technical Education Act was originally authorized by Congress in 1984. It was reauthorized in 1998 and again in 2006. This act provides federal funding for improving career technical education (CTE) within the United States in order to help the economy.

For Cabrillo College to receive a portion of this funding students in technical classes must fill out a survey. The more surveys completed the more funds the college will receive. The survey only needs to be completed once per term by each student.

This survey can be completed online using web advisor:

Log on to WEBADVISOR at https://wave.cabrillo.edu

Select "STUDENTS: Click Here" (navy blue bar)
- Under "Academic Profile" Click on "Student Update Form"
- Use drop down list under "Select the earliest term for which you are registered" and click on the current term.
- Select "SUBMIT"

Scroll down to the "Career Technical Information"
- Answer questions by clicking on the circle to the left of your "Yes" or "No" answers
- You can get details about a question by clicking on blue underlined phrase
- After answering all questions Select "SUBMIT"

Then "LOG OUT"

Thank you for taking a few minutes to help Cabrillo College CS/CIS programs!

- Rich

http://oslab.cabrillo.edu/forum/viewtopic.php?f=63&t=1883

This is an important source of funding for Cabrillo College.

*Send me an email that you completed this survey for **3 points extra credit!***

Help with labs

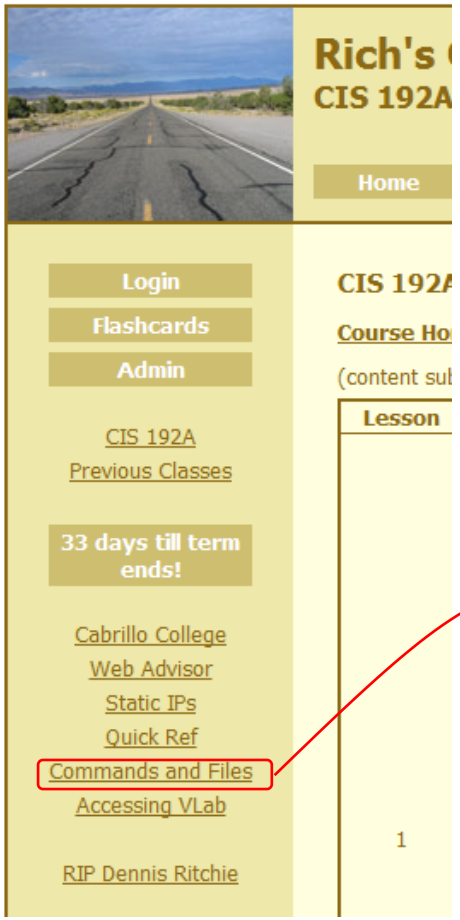
Like some help with labs?

I'm in the CIS Lab Monday afternoons
- See schedule at http://webhawks.org/~cislab/

or see me during office hours

or contact me to arrange another time online

Commands and Files
Quick Reference and Examples



Linux Network Commands & Files

Click on the link in the table below to see commands, configuration files and examples.

General Linux commands - root & shutdown General Linux commands - basic inventory Installing more commands	Network Testing
IP Addressing	
Interfaces Interfaces - DHCP client (temporary) Interfaces - Static IP (temporary) Interfaces - Red Hat family (permanent) Interfaces - Debian family (permanent)	
Name resolution	
ARP commands	
Linux hardware and driver commands	

Network configuration - Debian family (permanent)

~~Edit /etc/network/interfaces~~

Use this "deprecated" script to restart network services:

/etc/init.d/networking restart

It seems this script in now deprecated and each interface must be manually shut down then brought back up!

See: http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=565187

Edit this file to permanently configure networking on Debian and Ubuntu systems.

EXAMPLE - DHCP:

/etc/network/interfaces
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

EXAMPLE - static IP:

/etc/network/interfaces
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 172.30.4.222
netmask 255.255.255.0

gateway 172.30.4.1

Grades Web Page

http://simms-teach.com/cis192grades.php

Code Name	Grading Choice	Quizzes & Tests													Forum				Labs										Final	Extra Credit	Total	Grade
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	T1	T2	T3	F1	F2	F3	F4	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10				
Max Points		3	3	3	3	3	3	3	3	3	3	30	30	30	20	20	20	20	30	30	30	30	30	30	30	30	30	30	60	90	560	
Aragorn	Grade	2																	30	30										3		
Bilbo	Grade	3	3																29	28										6		
Denethor	P/NP	3	3																8	13										3		
Dwalin	Grade		3																	29												
Elrohir	Grade	3	3																30	30										25		
Elrond	Grade	3																	30	30										7		
Faramir	Grade	3	3																30	30										5		
Frodo	Grade	3	3																29	30										5		
Gwaihir	Grade		3																30	27												
Ioreth	Grade	3	3																30	30												
Legolas	Grade	3																	30	29										4		
Nazgul	Grade	3	3																30	30										11		
Pippin	Grade	3	3																30	30												
Samwise	Grade	3	3																30	30												
Saruman	Grade	3	3																30	30												
Strider	Grade	3	3																29	30												
Theoden	Grade	3	3																30	29												
Treebeard	Grade																															

Please check your:

- Grading Choice
- Quiz points
- Lab points
- Extra Credit points

Don't know you secret LOR code name?

… then email me your student survey to get it!

Reviewing graded work

Review graded work in your home directories

```
[simben192@opus ~]$ ls -l
total 52
-rw-r--r-- 1 simben192 cis192  610 Nov  7 08:51 capture
-rw-r--r-- 1 simben192 cis192  360 Nov  1 09:26 lab01
-r-------- 1 simben192 staff  4170 Nov  2 16:05 lab01.graded
-rw------- 1 simben192 cis192 3702 Nov  7 08:49 lab02
-rw-r--r-- 1 simben192 cis192 1350 Oct 31 19:05 labnotes
-rw-r--r-- 1 simben192 cis192 1400 Nov  1 13:15 notes
```

See example correct answers in the answers directory:

```
[simben192@oslab ~]$ ls /home/cis192/answers/
lab01  lab02  quiz01  quiz02
```

Stay on top of deliverables with the Calendar web page

Be ready for the first minute quizzes

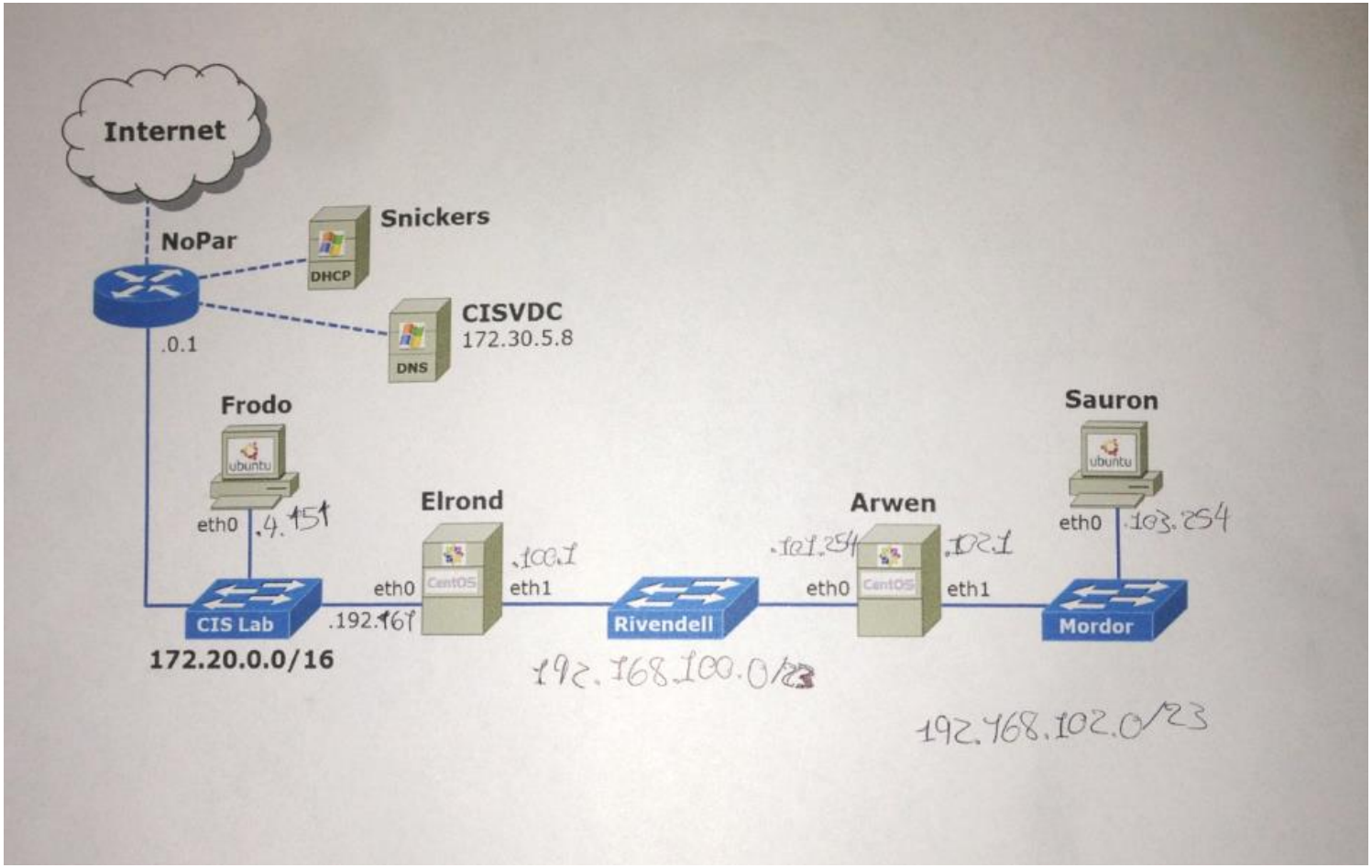
3	2/26	**Quiz 2** **IP Routing and** • Describe the • Describe the • Describe how accomplished • Use the Simp... a routing table • View, add, and delete entries in a routing table • By properly configuring routing tables on hosts and routers, configure a LAN of multiple segments which allows all hosts to communicate with each other. **Materials** • Presentation slides (download) **TBA Assignment** • Lab 3 (Routing) **CCC Confer** • Enter virtual classroom • Class archives	14 15	Lab 2
4	3/5	**Quiz 3** **TCP and the Transport Layer** • SBCs, Dynamips/Dynagen, VirtualBox • Dynamic routing • Quagga routing suite • RIPv2 implementation skills • Transport layer • TCP and UDP protocols • Service ports and sockets **Materials** • Presentation slides (download) **TBA Assignment** • Practice Test 1 • Lab 4 (Dynamic Routing and Port Forwarding) **CCC Confer** • Enter virtual classroom • Class archives	15	Lab 3 5 posts

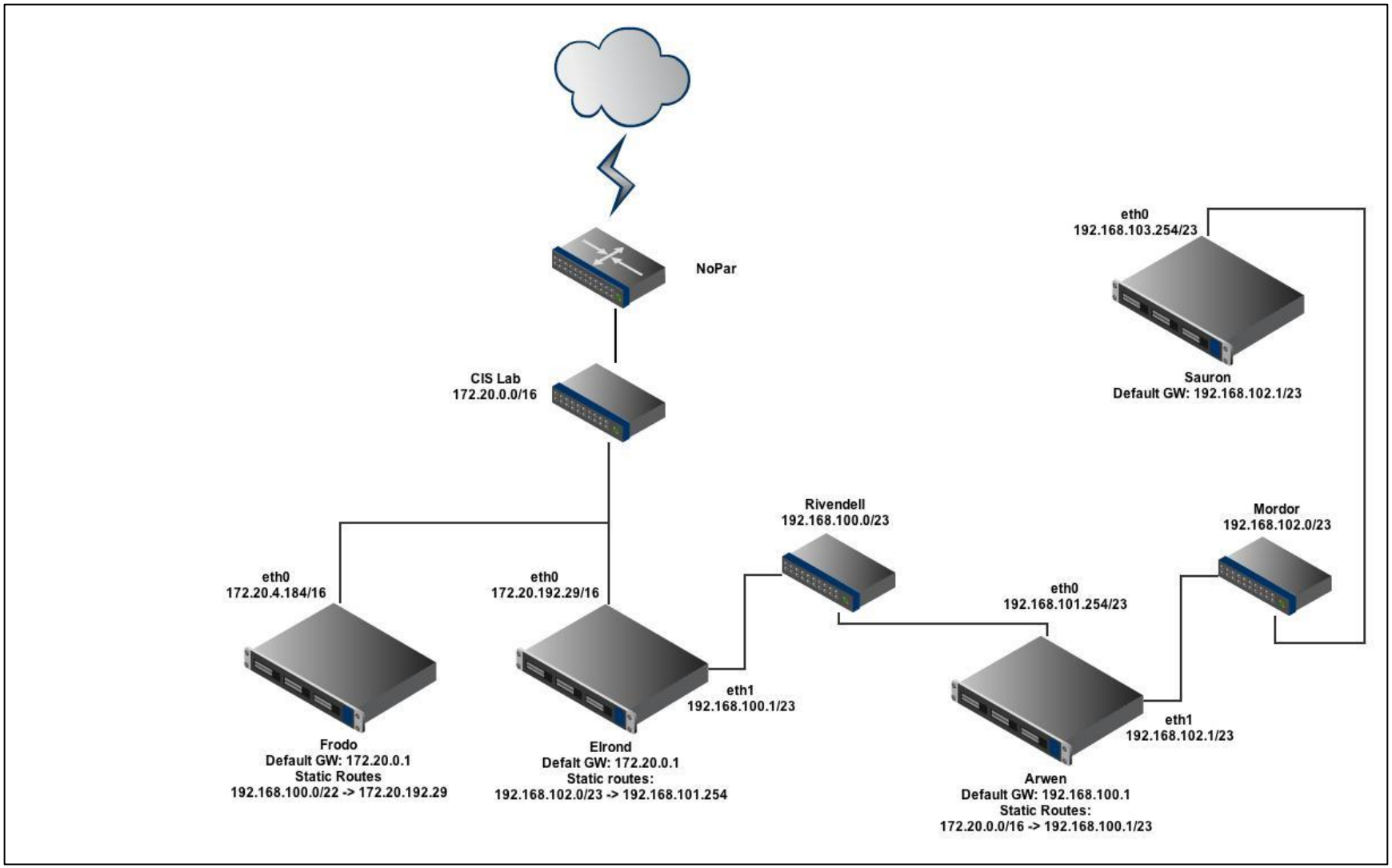
Lab 2 is due 11:59PM tonight

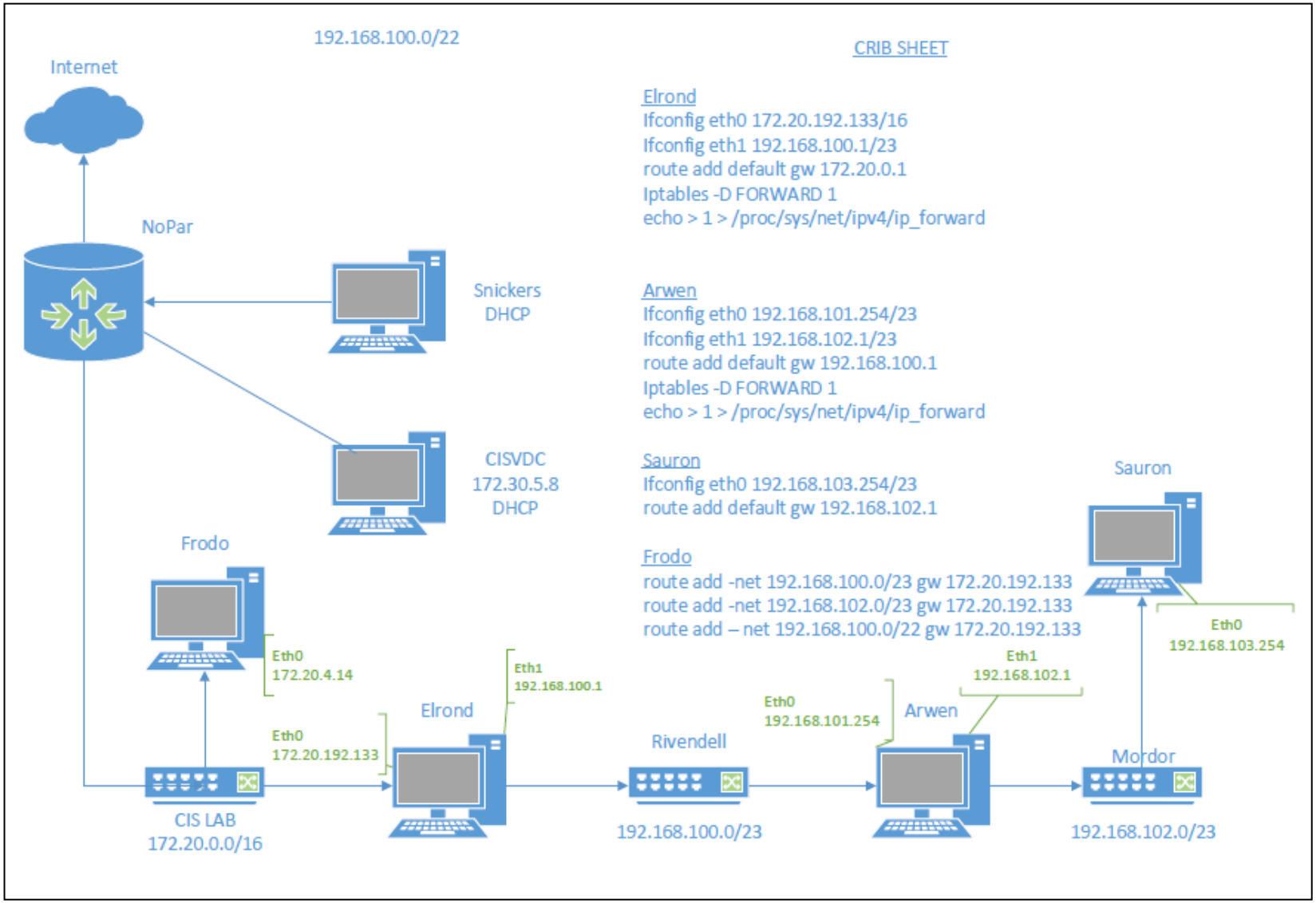
Lab 3 and five forum posts are due 11:59PM March 5th

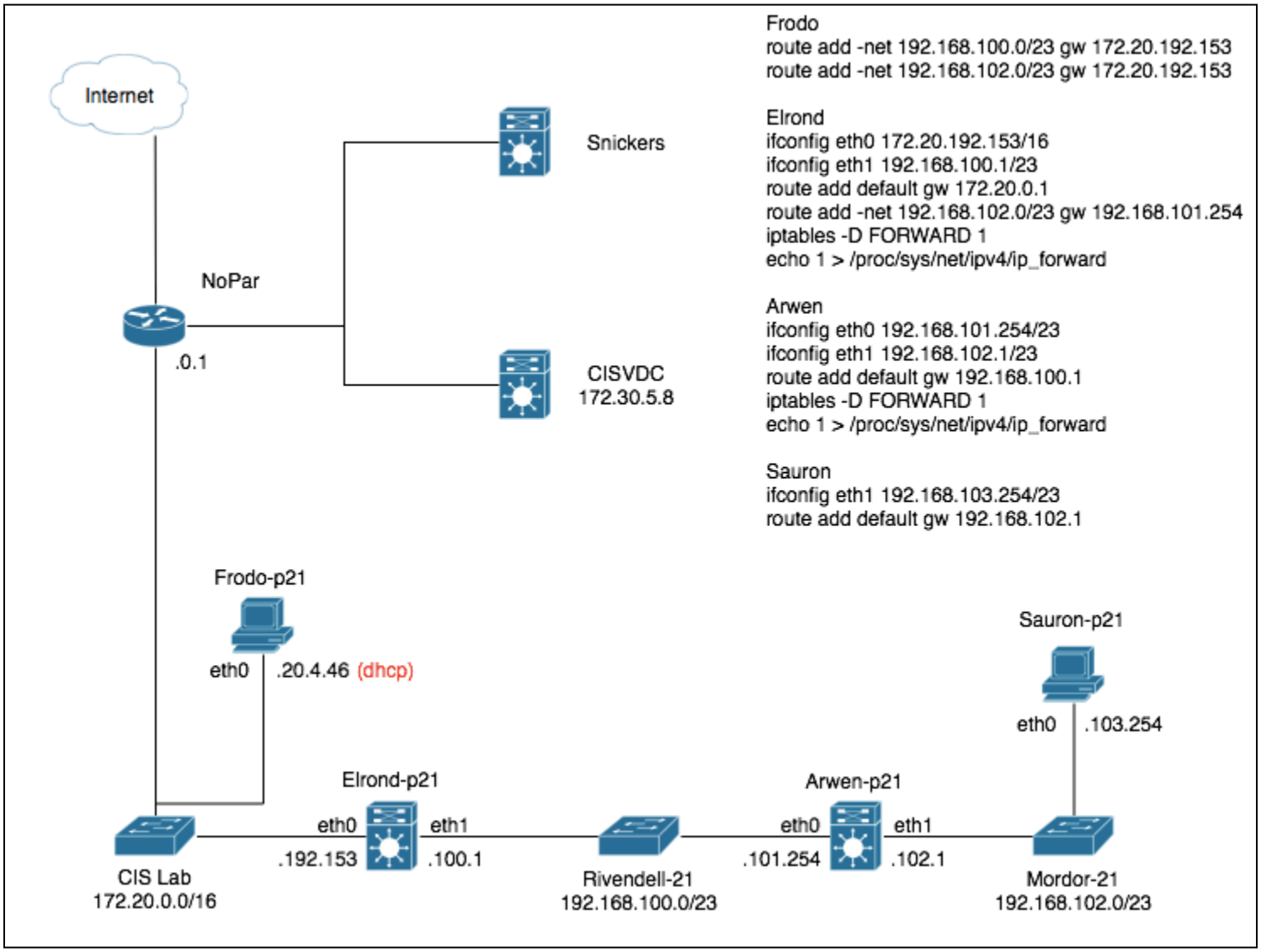
First test is the following week!

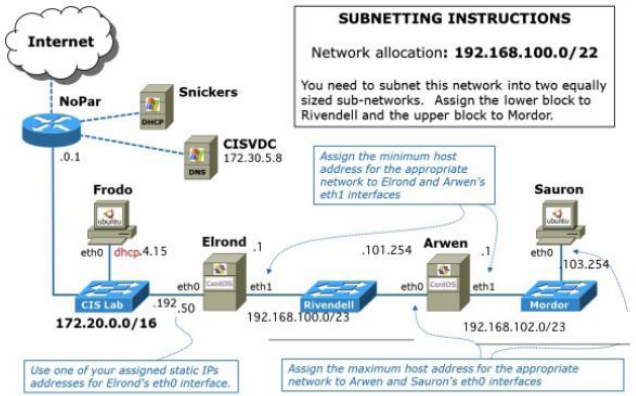
Growing Map Gallery


Internet
NoPar
.0.1
Snickers
DHCP
CISVDC
172.30.5.8
DNS
Frodo
ubuntu
eth0
.4.151
CIS Lab
172.20.0.0/16
Elrond
CentOS
eth0
.192.161
.100.1
eth1
Rivendell
192.168.100.0/23
Arwen
.101.254
eth0
CentOS
.102.1
eth1
Sauron
ubuntu
eth0
.103.254
Mordor
192.168.102.0/23


NoPar
CIS Lab
172.20.0.0/16
eth0
192.168.103.254/23
Sauron
Default GW: 192.168.102.1/23
Rivendell
192.168.100.0/23
Mordor
192.168.102.0/23
eth0
172.20.4.184/16
eth0
172.20.192.29/16
eth0
192.168.101.254/23
eth1
192.168.100.1/23
eth1
192.168.102.1/23
Frodo
Default GW: 172.20.0.1
Static Routes
192.168.100.0/22 -> 172.20.192.29
Elrond
Defalt GW: 172.20.0.1
Static routes:
192.168.102.0/23 -> 192.168.101.254
Arwen
Default GW: 192.168.100.1
Static Routes:
172.20.0.0/16 -> 192.168.100.1/23


192.168.100.0/22
Internet
NoPar
Snickers
DHCP
CISVDC
172.30.5.8
DHCP
Frodo
Eth0
172.20.4.14
Eth0
172.20.192.133
Elrond
Eth1
192.168.100.1
CIS LAB
172.20.0.0/16
Rivendell
192.168.100.0/23
Eth0
192.168.101.254
Arwen
Eth1
192.168.102.1
Sauron
Eth0
192.168.103.254
Mordor
192.168.102.0/23
CRIB SHEET
Elrond
Ifconfig eth0 172.20.192.133/16
Ifconfig eth1 192.168.100.1/23
route add default gw 172.20.0.1
Iptables -D FORWARD 1
echo > 1 > /proc/sys/net/ipv4/ip_forward
Arwen
Ifconfig eth0 192.168.101.254/23
Ifconfig eth1 192.168.102.1/23
route add default gw 192.168.100.1
Iptables -D FORWARD 1
echo > 1 > /proc/sys/net/ipv4/ip_forward
Sauron
Ifconfig eth0 192.168.103.254/23
route add default gw 192.168.102.1
Frodo
route add -net 192.168.100.0/23 gw 172.20.192.133
route add -net 192.168.102.0/23 gw 172.20.192.133
route add – net 192.168.100.0/22 gw 172.20.192.133

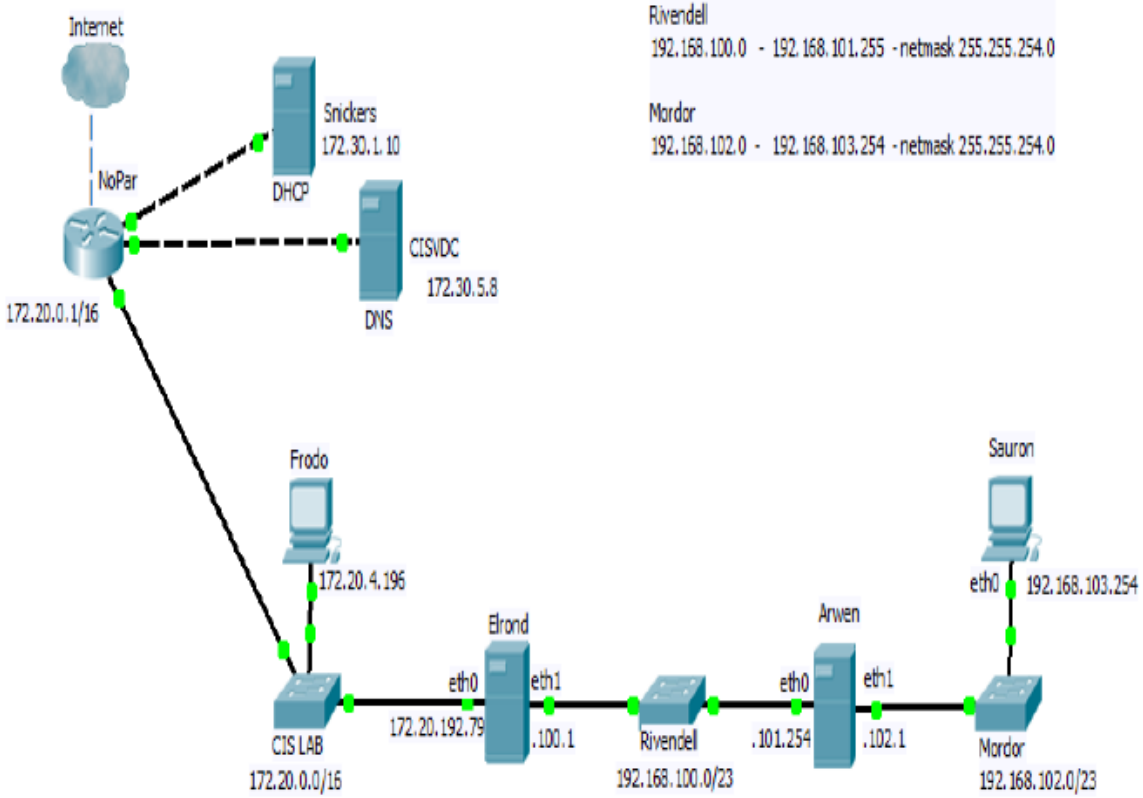

Internet
Snickers
NoPar
.0.1
CISVDC
172.30.5.8
Frodo
route add -net 192.168.100.0/23 gw 172.20.192.153
route add -net 192.168.102.0/23 gw 172.20.192.153
Elrond
ifconfig eth0 172.20.192.153/16
ifconfig eth1 192.168.100.1/23
route add default gw 172.20.0.1
route add -net 192.168.102.0/23 gw 192.168.101.254
iptables -D FORWARD 1
echo 1 > /proc/sys/net/ipv4/ip_forward
Arwen
ifconfig eth0 192.168.101.254/23
ifconfig eth1 192.168.102.1/23
route add default gw 192.168.100.1
iptables -D FORWARD 1
echo 1 > /proc/sys/net/ipv4/ip_forward
Sauron
ifconfig eth1 192.168.103.254/23
route add default gw 192.168.102.1
Frodo-p21
eth0
.20.4.46 (dhcp)
Sauron-p21
eth0
.103.254
Elrond-p21
eth0
.192.153
eth1
.100.1
Arwen-p21
eth0
.101.254
eth1
.102.1
CIS Lab
172.20.0.0/16
Rivendell-21
192.168.100.0/23
Mordor-21
192.168.102.0/23



Default gateways
Elrond, 172.20.0.1 (NoPar)
Arwen, 192.168.100.1 (Elrond)
Sauron, 192.168.102.1 (Arwe)
Frodo 172.20.0.1 (NoPar)

Static Routes
Elrond 192.168.102.0/23 192.168.101.254
Frodo 192.168.100.0/22 172.20.192.50
If commands
Elrond
ifconfig eth0 172.20.192.50/16
ifconfig eth1 192.168.100.1/23
Arwen
ifconfig eth0 192.168.101.254/23
ifconfig eth1 192.168.102.1/23
Sauron
ifconfig eth0 192.168.103.254/23
Route commands
All default gw's
route add default gw xxx.xxx.xxx.xxx
Elrond
route add -net 192.168.102.0/23 gw 192.168.101.254
Frodo
route add -net 192.168.100.0/22 gw 172.20.192.50
Firewall cmds (Elrond+Arwen)
iptables -D FORWARD 1
Packet Fwd (Elrond+Arwen)
echo 1 > /proc/sys/net/ipv4/ip_forward

Lab 3 Network Diagram



192.168.100.0/22 subnetted into two equal networks

Rivendell
192.168.100.0 - 192.168.101.255 - netmask 255.255.254.0

Mordor
192.168.102.0 - 192.168.103.254 - netmask 255.255.254.0

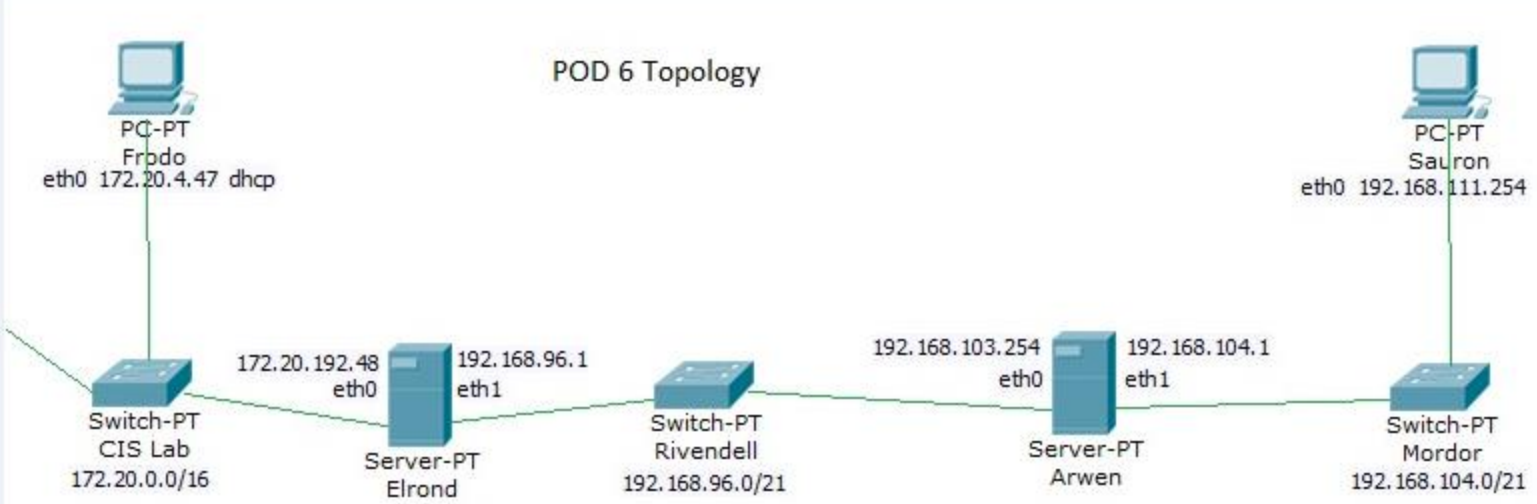
CRIB SHEET

-Elrond
ifconfig eth0 172.20.192.79/16
ifconfig eth1 192.168.100.1/23
route add default gw 172.20.0.1/16
route add -net 192.168.100.0/22 gw 192.168.101.254
iptables -D FORWARD 1
echo 1 > /proc/sys/net/ipv4/ip_forward

-Arwen
ifconfig eth0 192.168.101.254/23
ifconfig eth1 192.168.102.1/23
route add default gw 192.168.100.1
iptables -D FORWARD 1
echo 1 > /proc/sys/net/ipv4/ip_forward

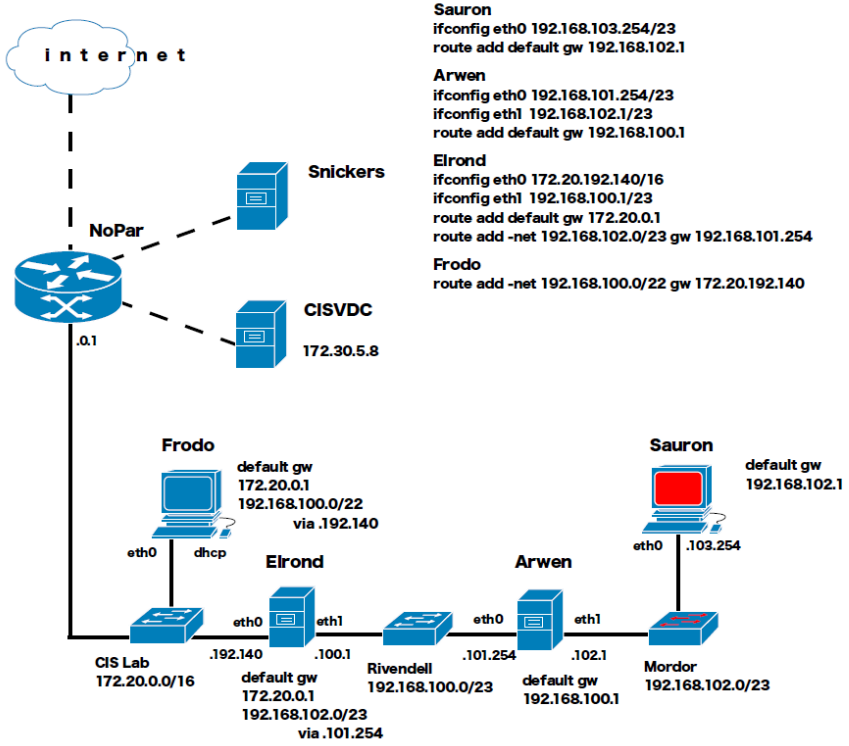
-Sauron
ifconfig eth0 192.168.103.254/23
route add default gw 192.168.102.1

-Frodo
route add -net 192.168.100.0/22 gw 172.20.192.79

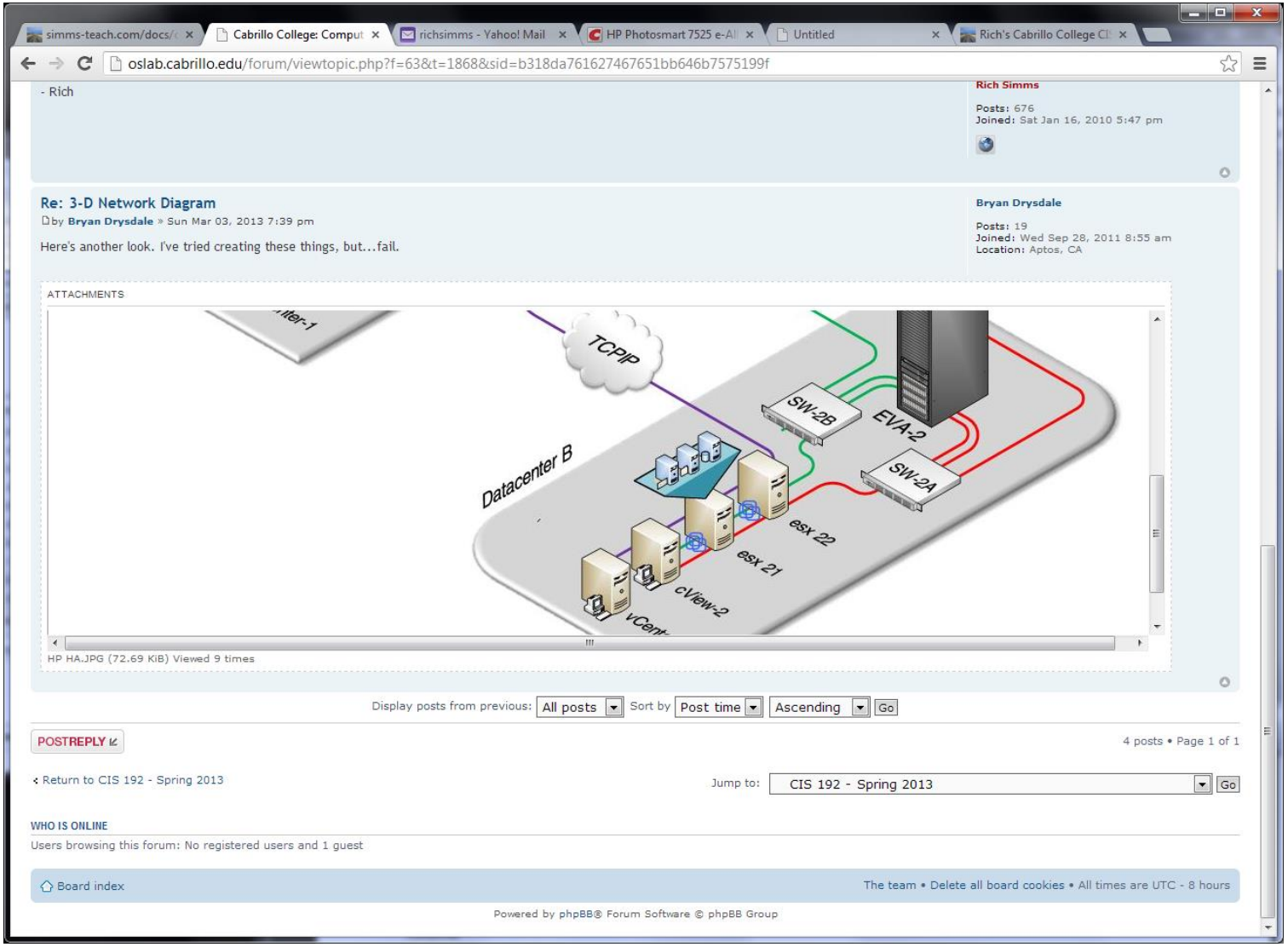


Hostname/Interface	IP Address	Subnet Mas	Default Gateway	iconfig cmd	route cmd
Frodo					
Eth0	172.20.4.47	255.255.0.0		n/a	
Elrond					
Eth0	172.20.192.48	255.255.0.0	172.20.0.1	ifconfig eth0 172.20.192.48/16	
Eth1	192.168.96.1	255.255.248.0		ifconfig eth1 192.168.96.1/21	route add -net 192.168.104.0/21 gw 192.168.103.254
Arwen					
Eth0	192.168.103.254	255.255.248.0		ifconfig eth0 192.168.103.254/21	
Eth1	192.168.104.1	255.255.248.0		ifconfig eth1 192.168.104.1/21	
Sauron					
eth0	192.168.111.254	255.255.248.0	192.168.104.1	ifconfig eth0 192.168.111.254/21	route add -net 192.168.96.0/21 gw 192.168.104.1

crib sheet.



enter mordor
at own risk.

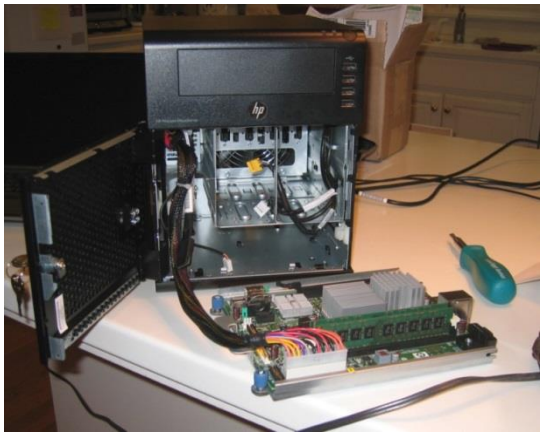

oslab.cabrillo.edu/forum/viewtopic.php?f=63&t=1868&sid=b318da761627467651bb646b7575199f
- Rich
Rich Simms
Posts: 676
Joined: Sat Jan 16, 2010 5:47 pm
Re: 3-D Network Diagram
by Bryan Drysdale » Sun Mar 03, 2013 7:39 pm
Here's another look. I've tried creating these things, but...fail.
Bryan Drysdale
Posts: 19
Joined: Wed Sep 28, 2011 8:55 am
Location: Aptos, CA
ATTACHMENTS
TCPIP
SW-2B
EVA-2
SW-2A
Datacenter B
esx 22
esx 21
cView-2
HP HA.JPG (72.69 KiB) Viewed 9 times
Display posts from previous: All posts Sort by Post time Ascending Go
POSTREPLY
4 posts • Page 1 of 1
Return to CIS 192 - Spring 2013
Jump to: CIS 192 - Spring 2013 Go
WHO IS ONLINE
Users browsing this forum: No registered users and 1 guest
Board index
The team • Delete all board cookies • All times are UTC - 8 hours
Powered by phpBB® Forum Software © phpBB Group

Class Activity

Use CCC Confer White Board

Home Labs

HP Microserver



Mikrotik Router

VMware ESXi for virtualization

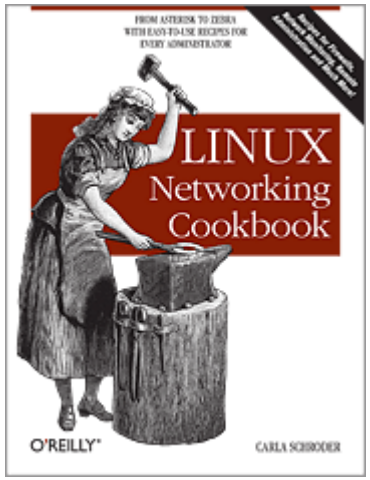
Free Microsoft software
(academic license)

Free VMware software
(academic license)

Free Linux Distros
(GNU license)

SBCs

Single Board Computers



The **Linux Networking Cookbook** by Carla Schroeder has a section on SBCs (Single Board Computers):

- Small
- Quiet
- Low power consumption
- Can run Linux OS

Examples:

- Soekris Engineering (Santa Cruz) - http://soekris.com/
- PC Engines (Switzerland) - http://www.pcengines.ch/
- MikroTik Routerboard (Latvia) - http://www.routerboard.com/
- Many more at http://www.linuxfordevices.com/

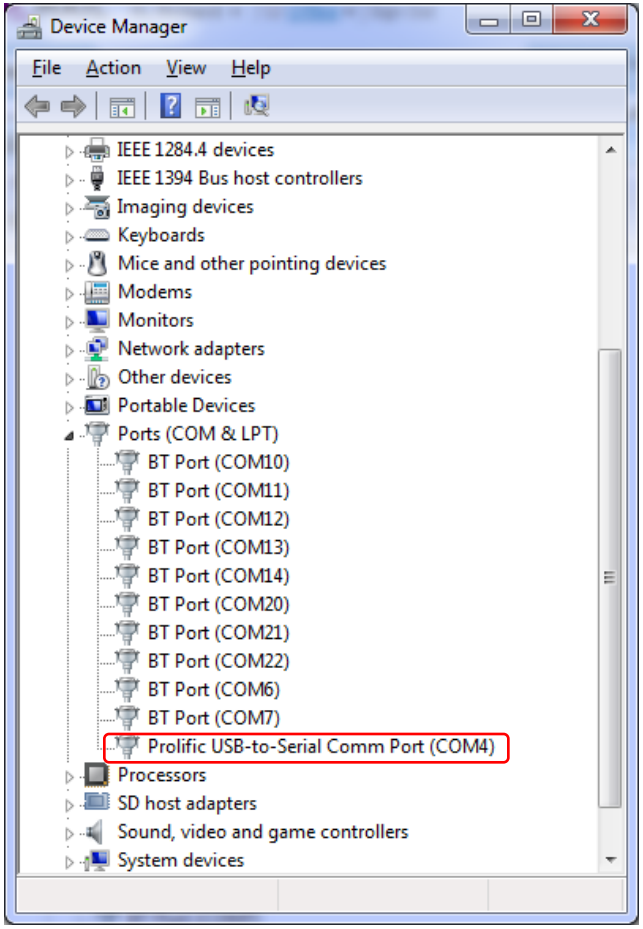
MikroTik/Routerboard - A Linux based router

Assemble your own Linux based Router. This one has five Ethernet interfaces and uses 6.4 watts of power.

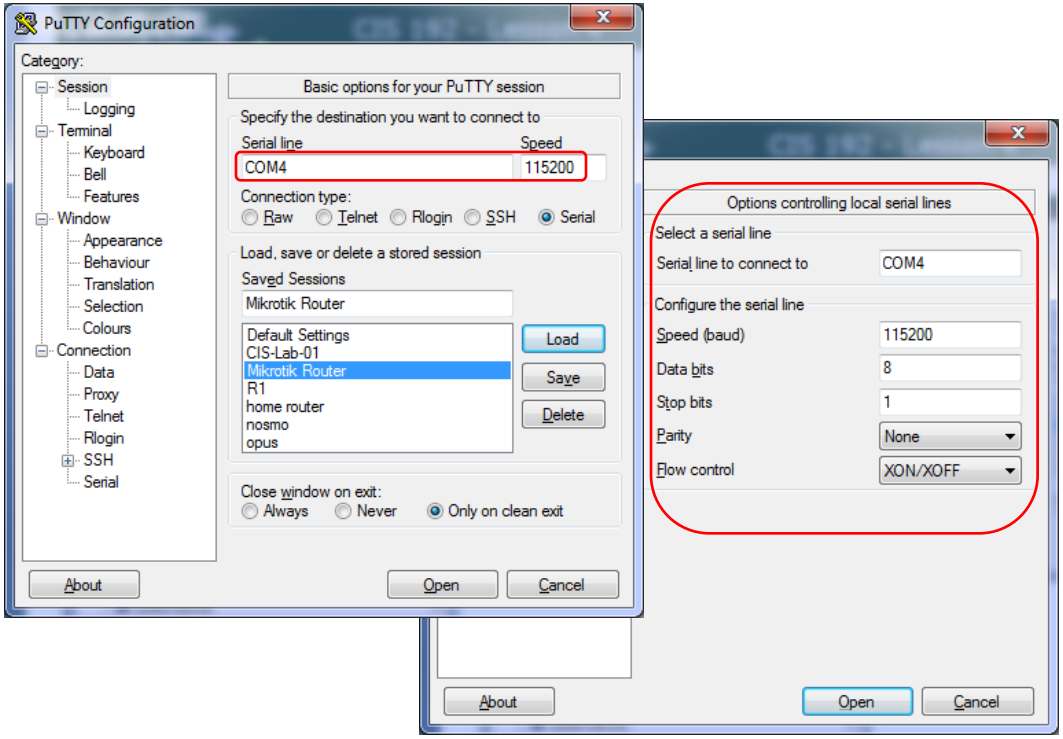
- *Eth1 is attached to the home LAN.*
- *Eth2 is attached to a 172.30.4.0/24 network.*
- *Eth3 is attached to a 172.30.1.0/24 network.*
- *The serial cable (console) can be attached to a laptop.*

• RB/450 Routerboard	$69
• CA/150 indoor case	$19
• 24HPOW power supply	$18
• SW-1301 USB-to-serial adapter	$12

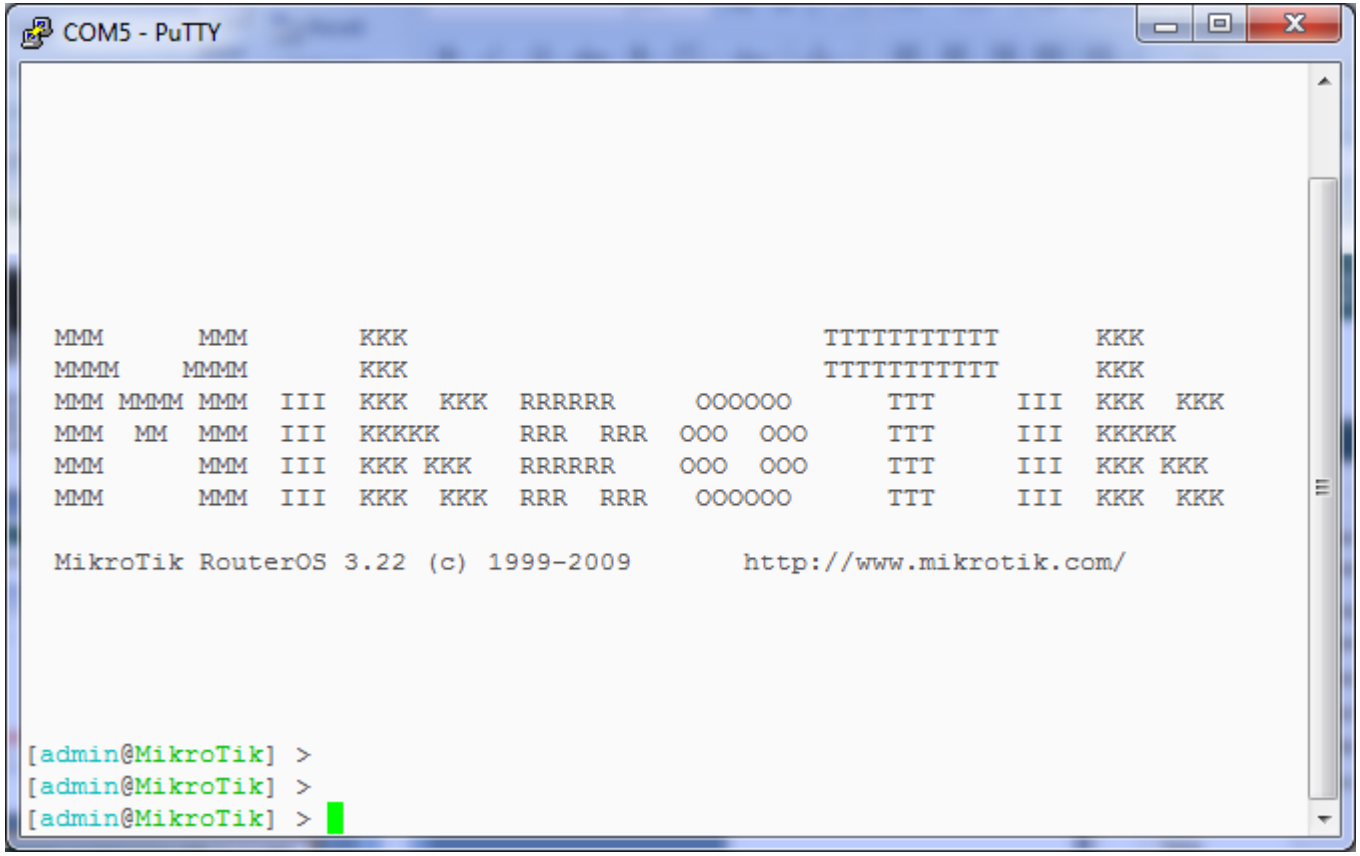
MikroTik/Routerboard - A Linux based router



With a USB-to-Serial adapter Putty can be used as the console

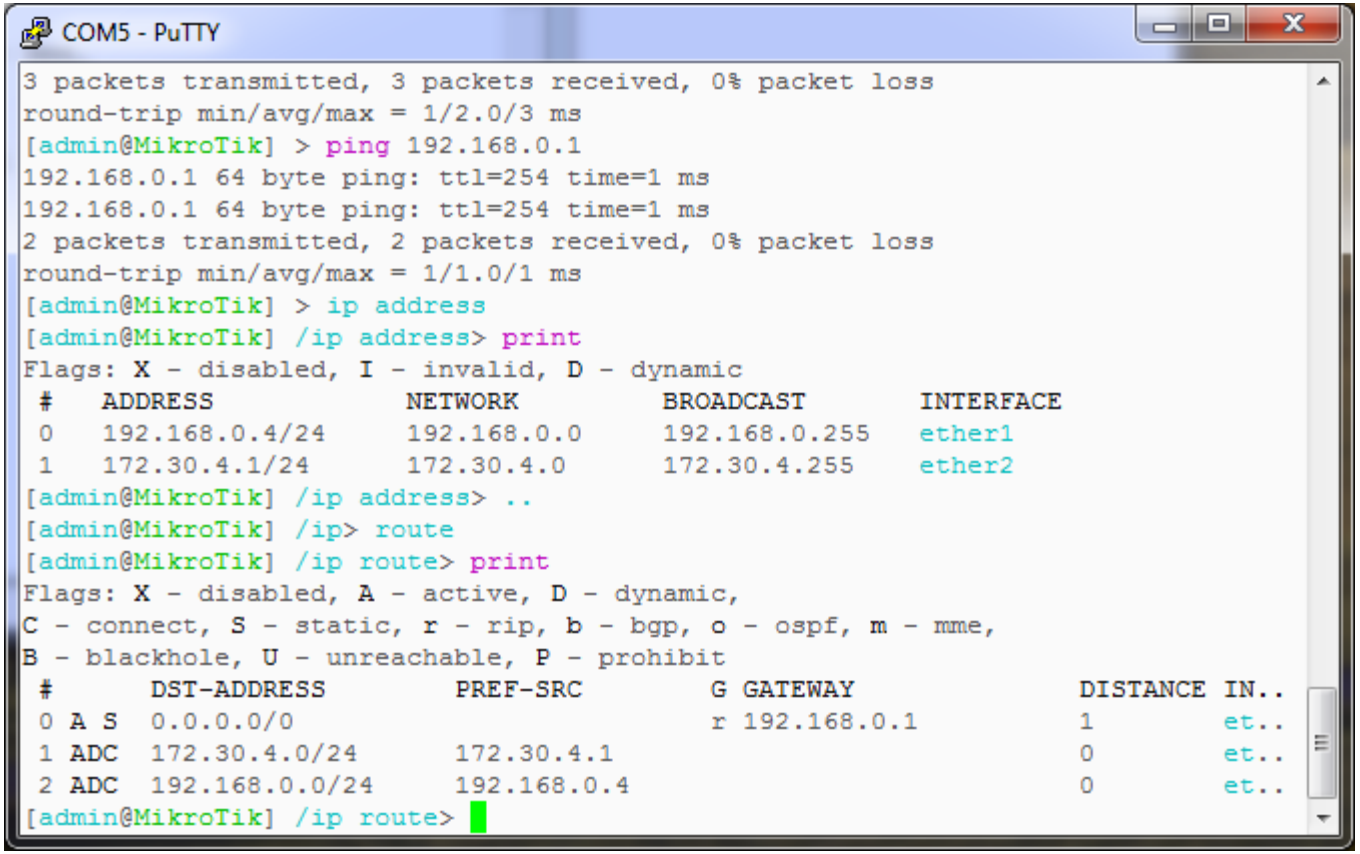


MikroTik/Routerboard - A Linux based router



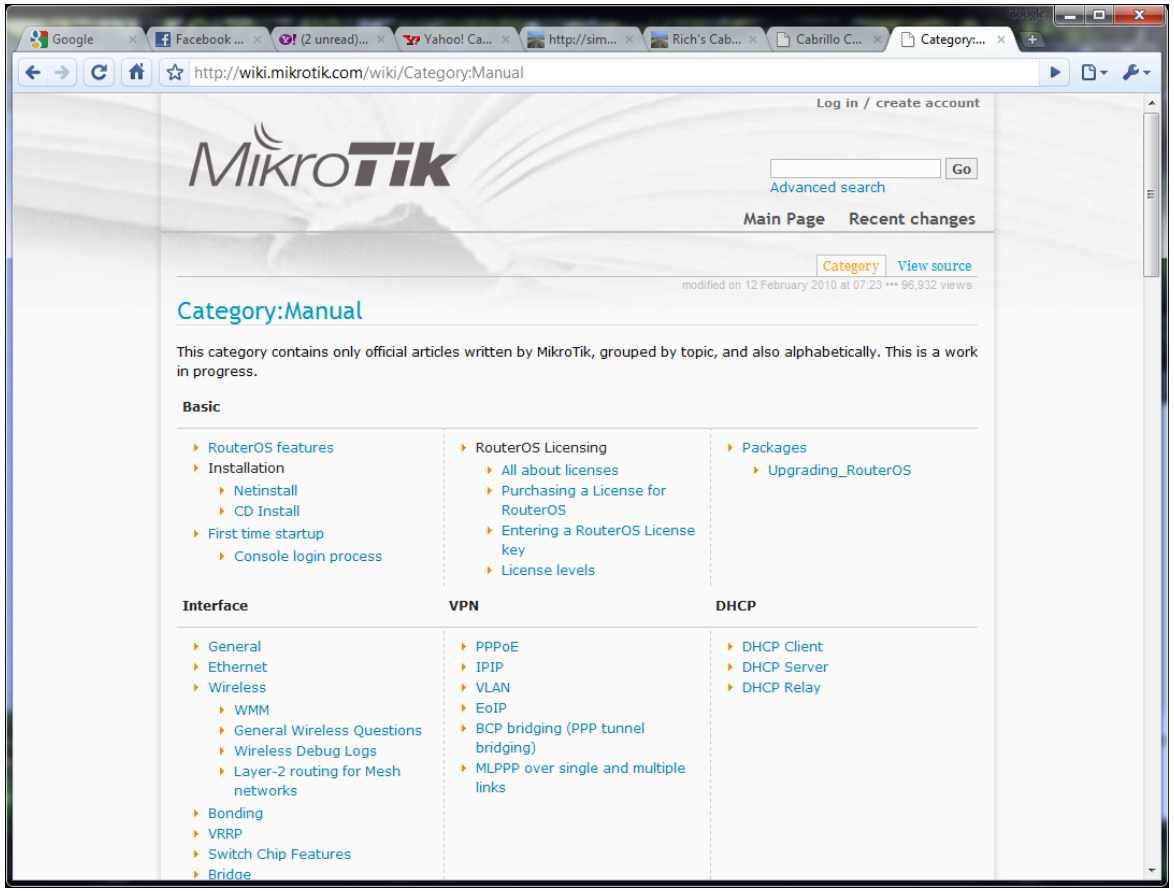
MikroTik RouterOS provides their own shell and software that runs on a Linux 2.6 kernel. The admin account is initially set with no password for first time login.

MikroTik/Routerboard - A Linux based router



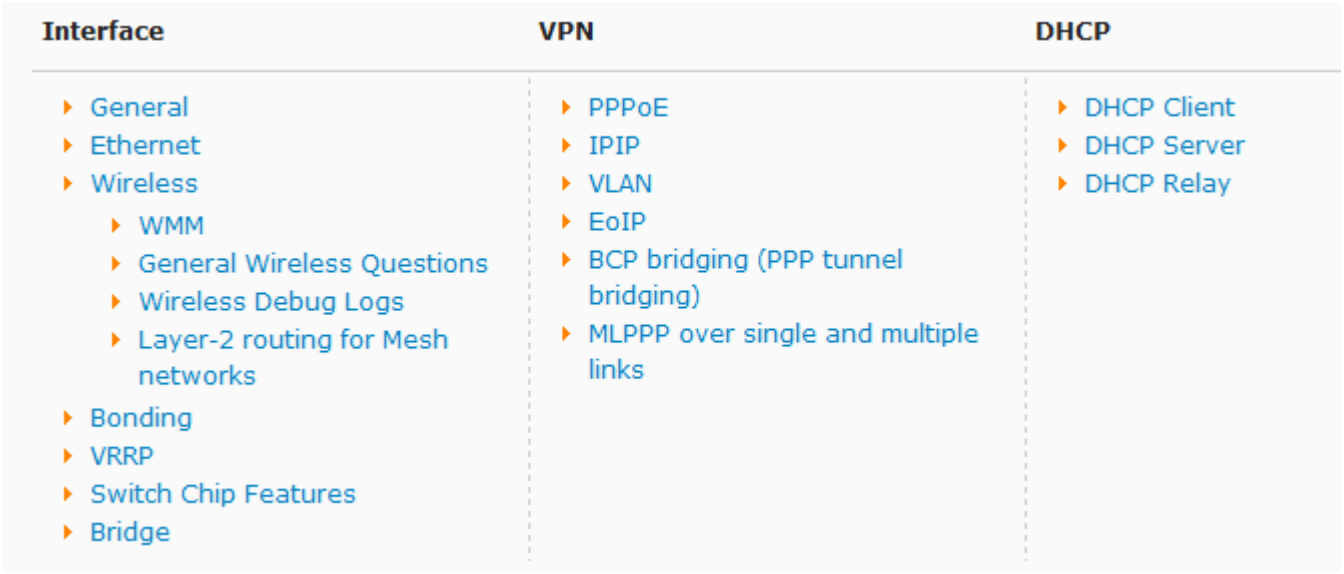
The shell let's you configure and show interfaces, routes, DHCP, etc.

MikroTik/Routerboard - A Linux based router



Online wiki documentation

MikroTik/Routerboard - A Linux based router



Interface

- General
- Ethernet
- Wireless
  - WMM
  - General Wireless Questions
  - Wireless Debug Logs
  - Layer-2 routing for Mesh networks
- Bonding
- VRRP
- Switch Chip Features
- Bridge

VPN

- PPPoE
- IPIP
- VLAN
- EoIP
- BCP bridging (PPP tunnel bridging)
- MLPPP over single and multiple links

DHCP

- DHCP Client
- DHCP Server
- DHCP Relay

Online wiki documentation areas

MikroTik/Routerboard - A Linux based router

Traffic control

- Packet Flow
- Queue
  - HTB type
  - Burst
  - Queue Size
  - PCQ type

Firewall control

- Firewall filter
- Firewall nat
- Firewall mangle
- Layer 7 matcher
- Services
- Address list
- PCC *per-connection-classifier*
- Connection Rate *connection-rate*
- UPnP

IP and Routing

- Ip address
- ARP
- Routing in general
- VRF
- Routing filters
- OSPF theory
  - OSPF-examples
  - OSPF-reference
- BGP
  - BGP based VPLS
  - BGP HowTo & FAQ
  - BGP Soft Reconfiguration
  - BGP Load Balancing
- RIP
  - Prefix list

Online wiki documentation areas

MikroTik/Routerboard - A Linux based router

Console

- Console
  - Line editor
  - Prompt
  - Scripting
    - Scripting-examples
    - Lua
  - Safe mode

User management

- Hotspot
- User Manager
- PPP AAA
- Router AAA
- RADIUS Client

Examples

- VRRP-examples
- Scripting-examples
- OSPF-examples
- A complete Layer-3 MPLS VPN example
- BGP HowTo & FAQ
- BGP Load Balancing with two interfaces
- Making a simple wireless AP
- PCQ Examples
- Load balancing multiple same subnet links

Online wiki documentation areas

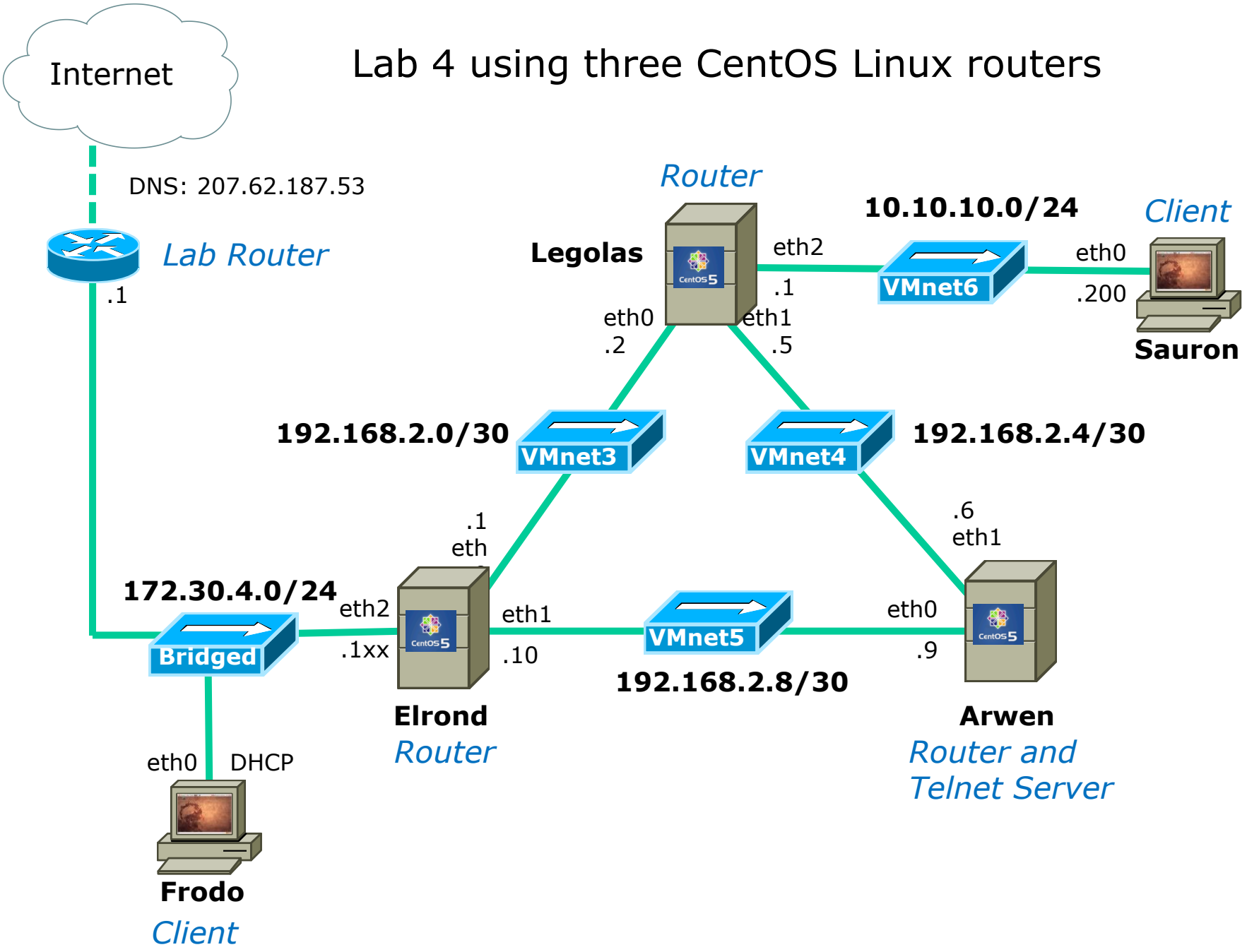
MikroTik/Routerboard - A Linux based router

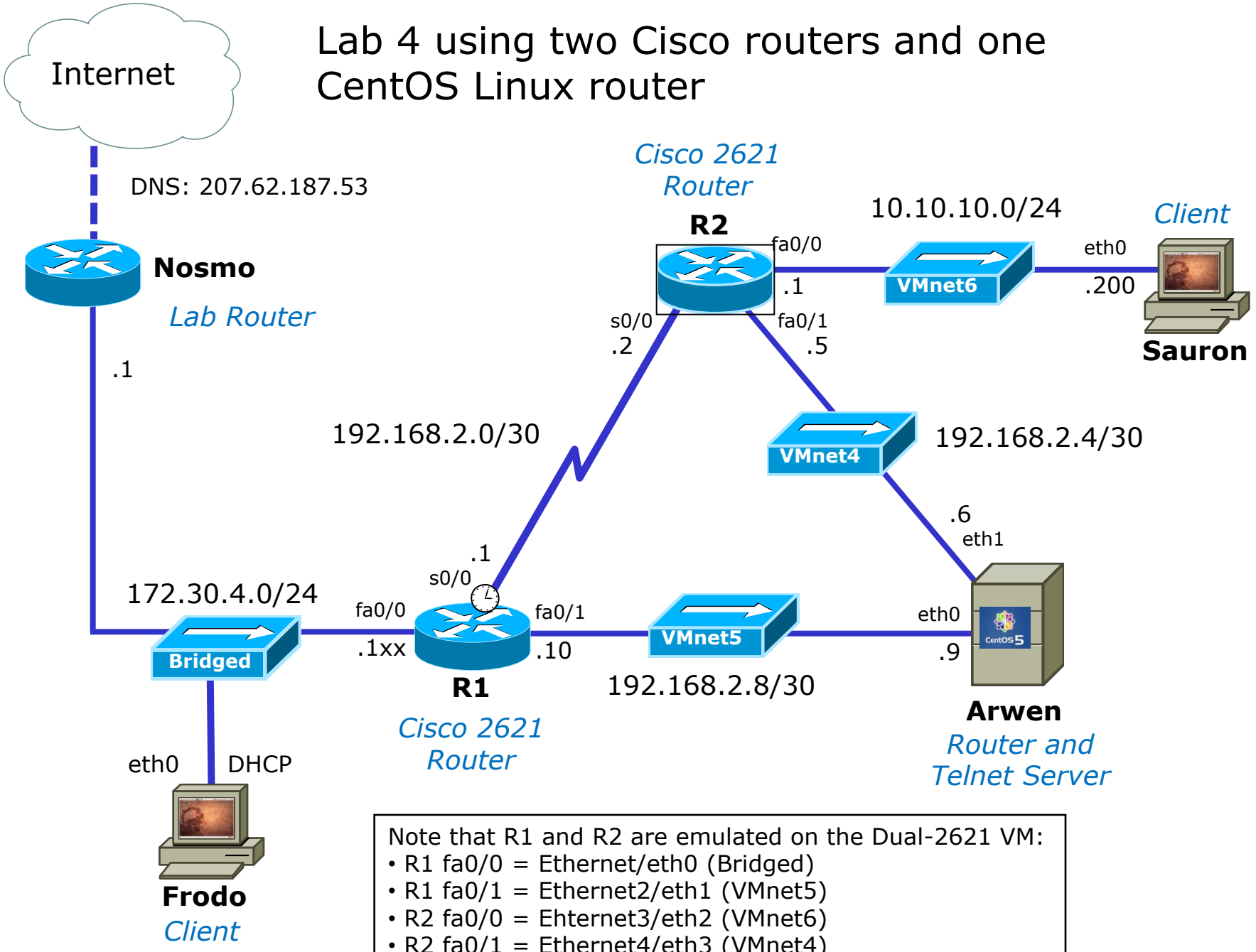
Internetworking	Hardware	Other
MPLS MPLS_Overview MPLSVPLS EXP bit behaviour BGP based VPLS Virtual Routing and Forwarding MPLS TE Tunnels Multicast routing (PIM) IGMP Proxy	Switch Chip Features MikroTik Password Recovery Maximum Transmission Unit on RouterBoards R52 diagnose	Virtualization Xen Metarouter Special_Login

Online wiki documentation areas

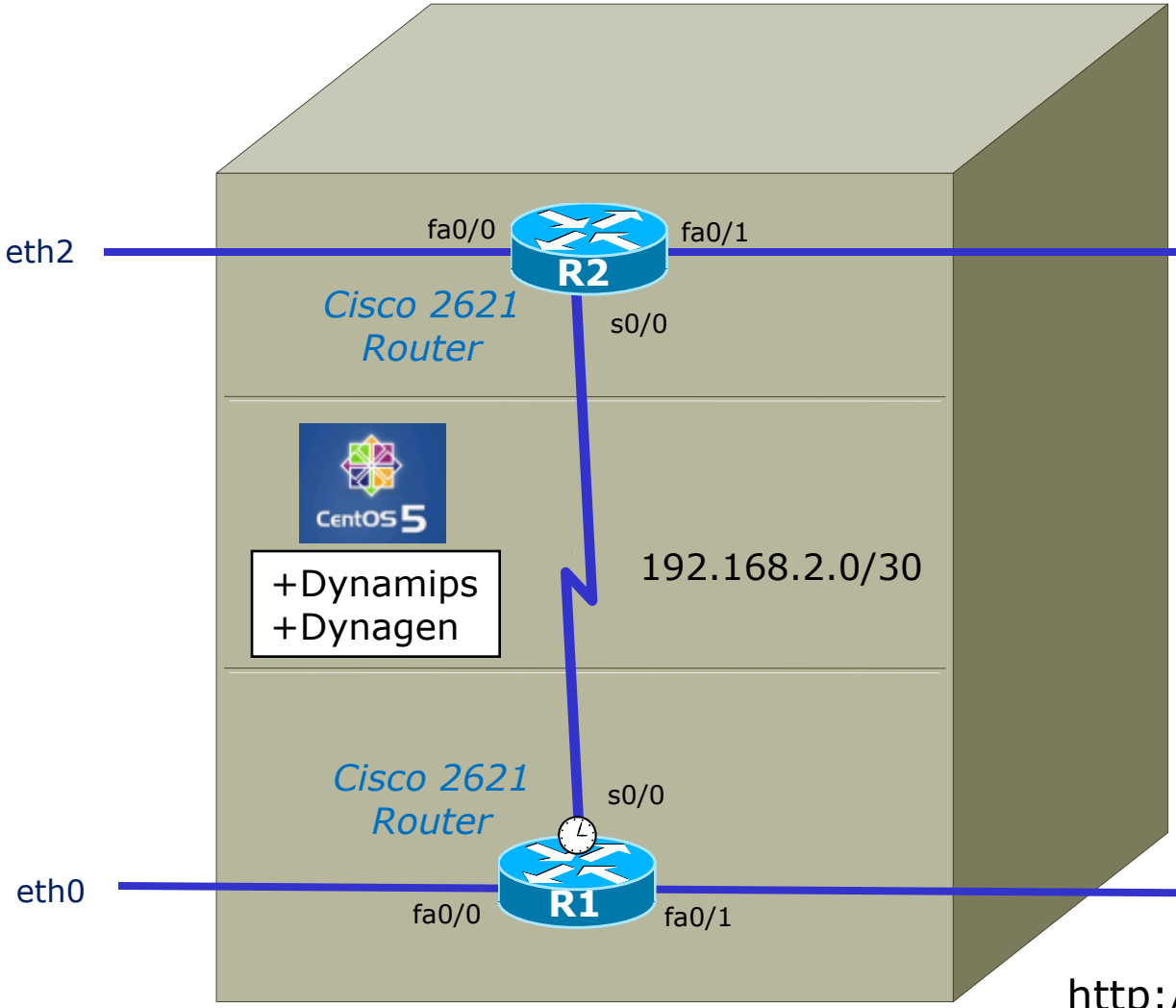
Dynamips
Dynagen

Lab 4 using three CentOS Linux routers




Lab 4 using two Cisco routers and one CentOS Linux router
Internet
DNS: 207.62.187.53
Nosmo
Lab Router
.1
Cisco 2621 Router
R2
fa0/0
.1
10.10.10.0/24
VMnet6
Client
eth0
.200
Sauron
s0/0
.2
fa0/1
.5
192.168.2.0/30
VMnet4
192.168.2.4/30
.6
eth1
.1
s0/0
172.30.4.0/24
fa0/0
.1xx
fa0/1
.10
R1
Cisco 2621 Router
VMnet5
192.168.2.8/30
eth0
.9
Arwen
Router and Telnet Server
Bridged
eth0
DHCP
Frodo
Client
Note that R1 and R2 are emulated on the Dual-2621 VM:
• R1 fa0/0 = Ethernet/eth0 (Bridged)
• R1 fa0/1 = Ethernet2/eth1 (VMnet5)
• R2 fa0/0 = Ehternet3/eth2 (VMnet6)
• R2 fa0/1 = Ethernet4/eth3 (VMnet4)

The Dual-c2621s VM



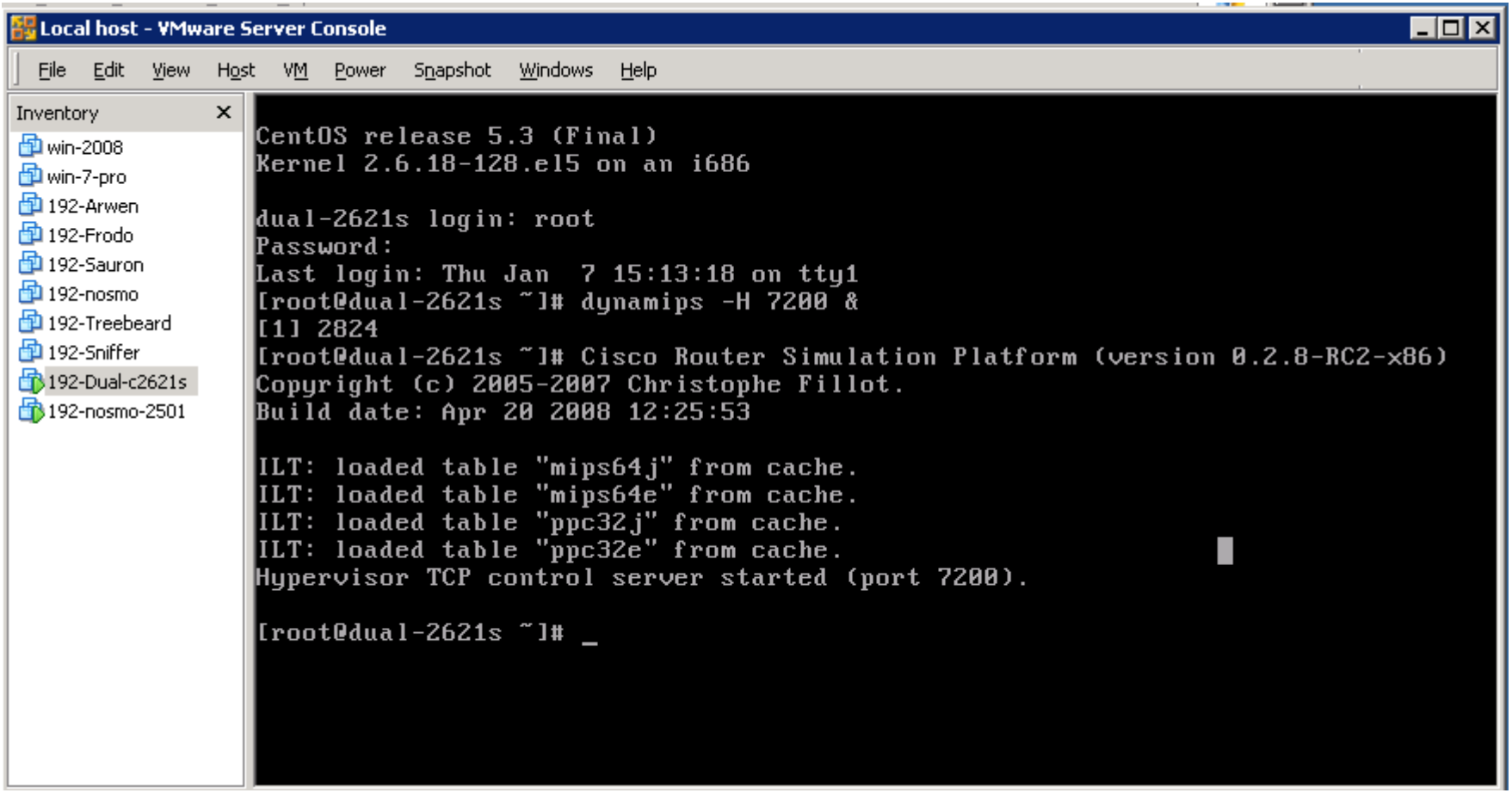
- R1 fa0/0 = Ethernet/eth0
- R1 fa0/1 = Ethernet2/eth1

- R2 fa0/0 = Ehternet3/eth2
- R2 fa0/1 = Ethernet4/eth3

A CentOS VM that runs Dynamips/Dynagen to emulate one or more Cisco routers

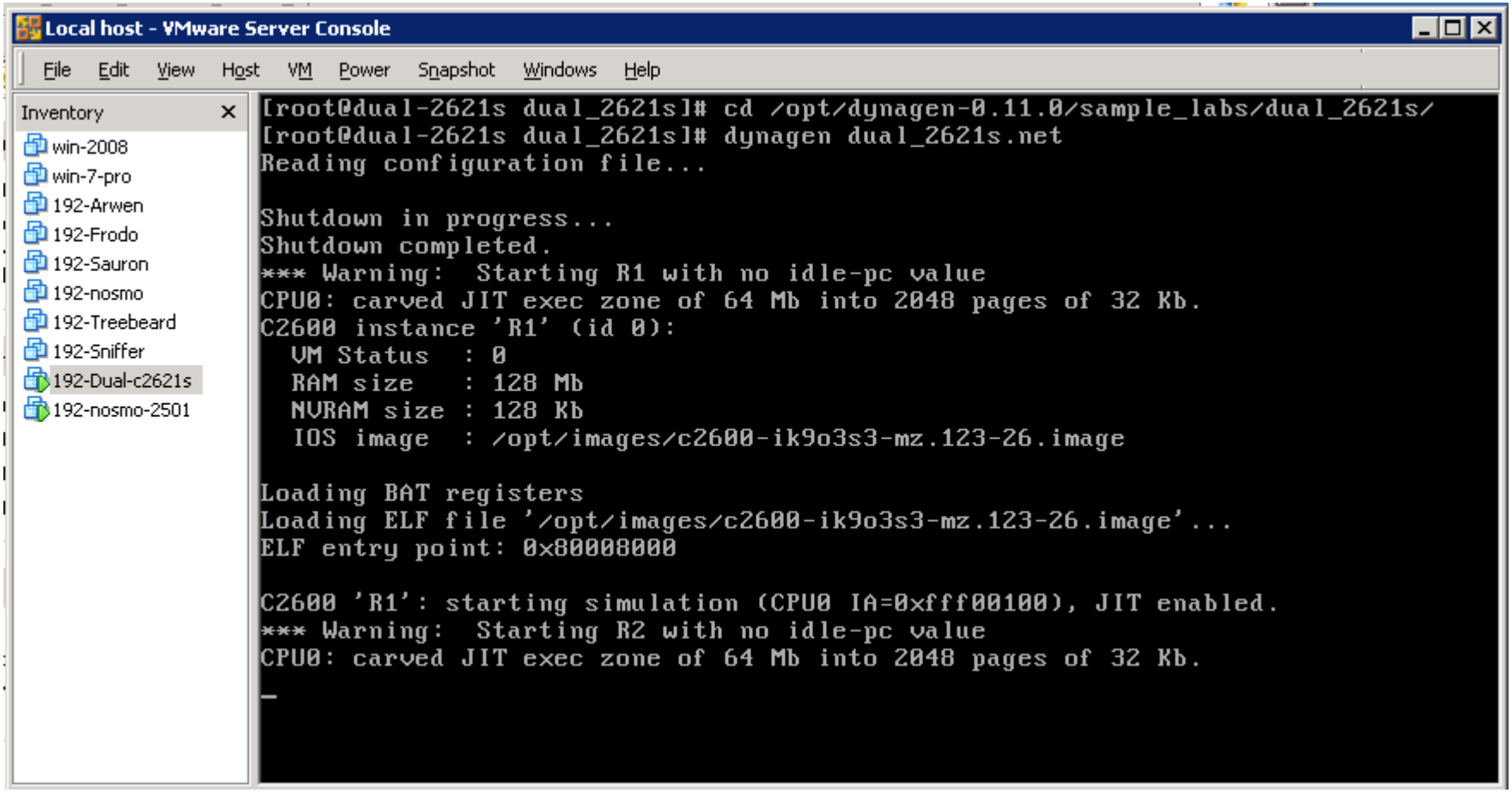
http://dynagen.org/tutorial.htm

The Dual-c2621s VM



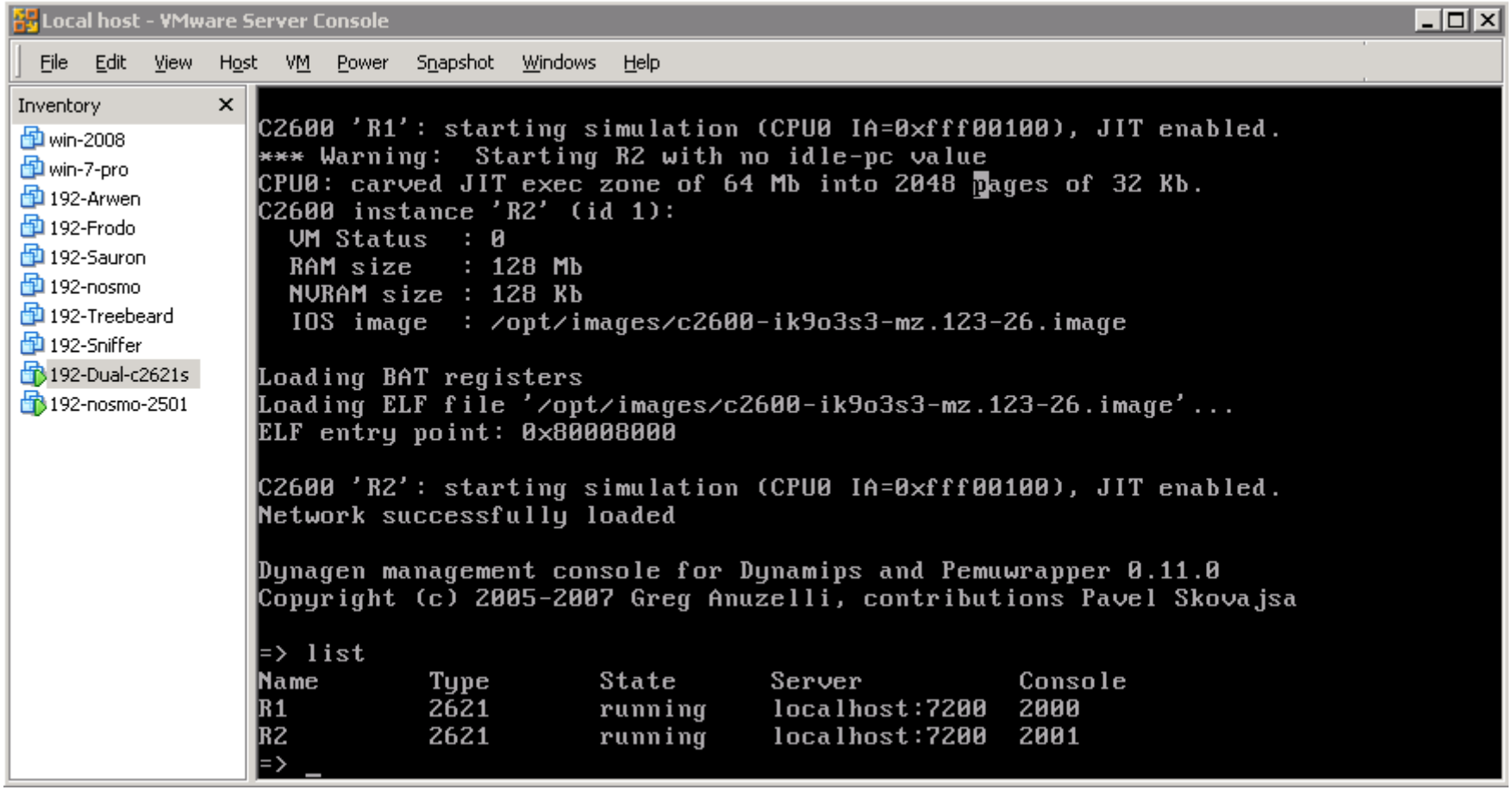
*Use **dynamips -H 7200 &** to run the Dynamips hardware emulator and listen using port 7200*

The Dual-c2621s VM



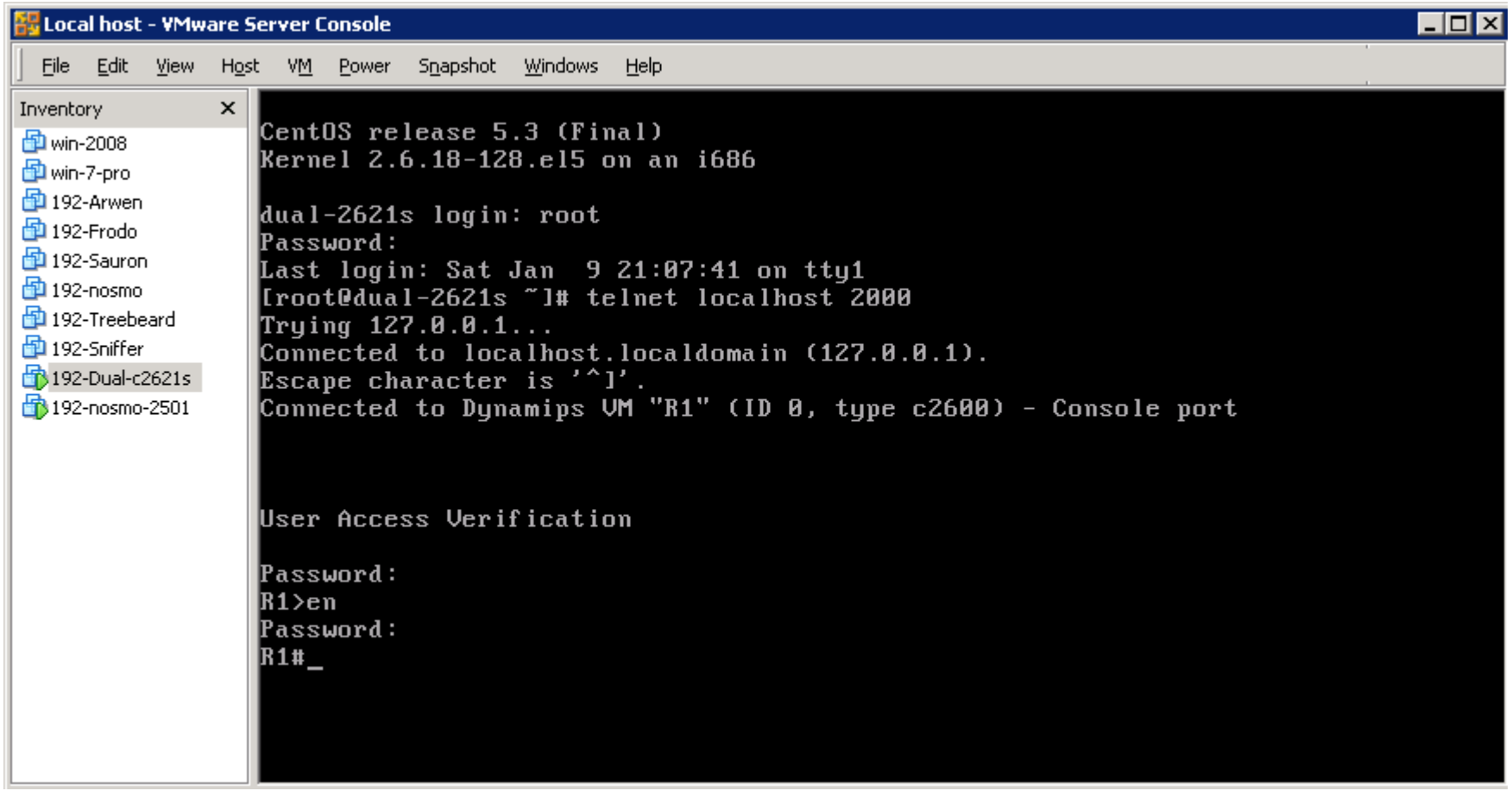
*Change directory to where the Dynagen configuration files are then use **dynagen dual-2621s.net** to start up two 2621 virtual routers*

The Dual-c2621s VM



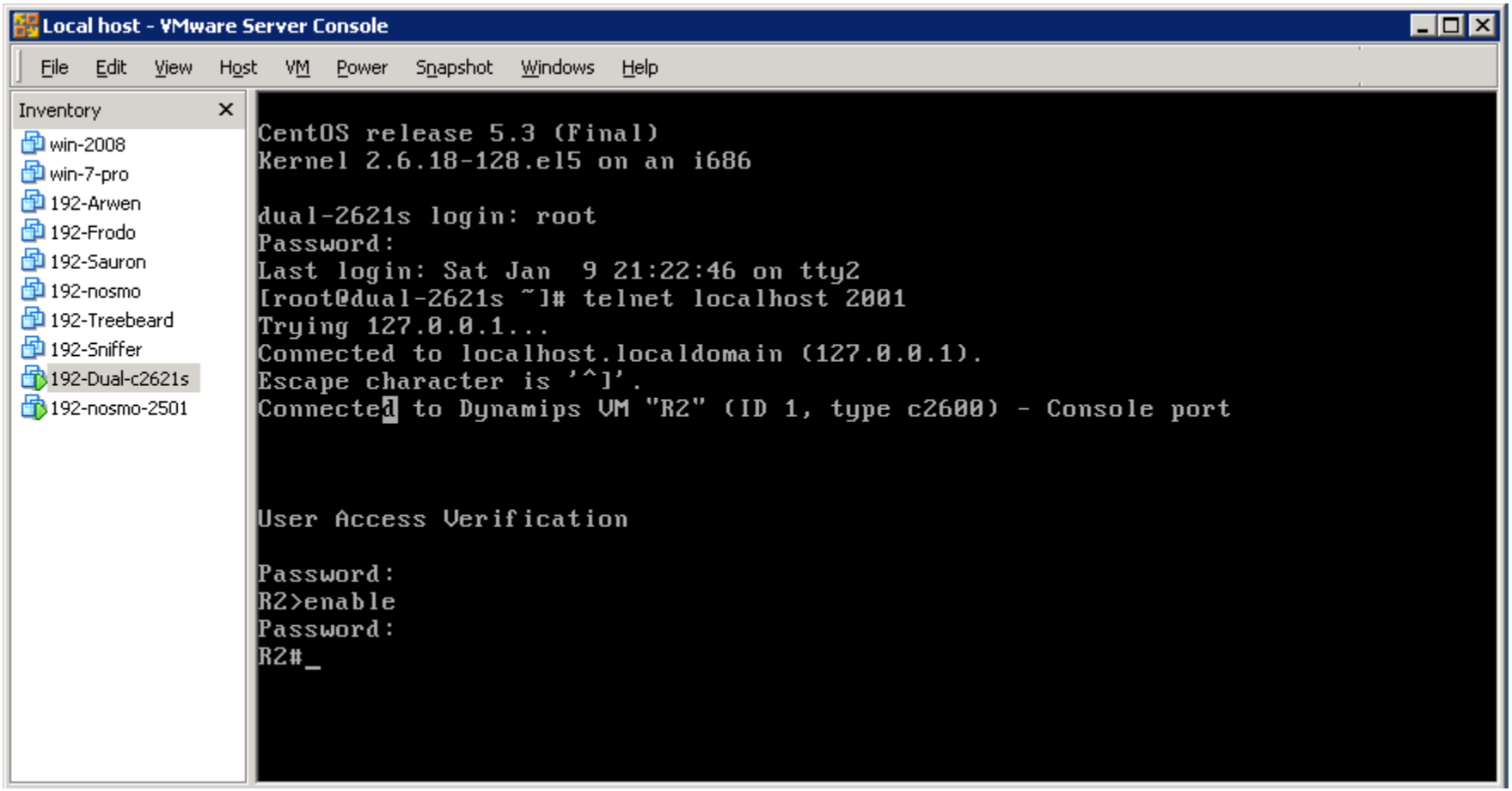
*Use **list** command to show the virtual routers and the ports they are listening on*

The Dual-c2621s VM



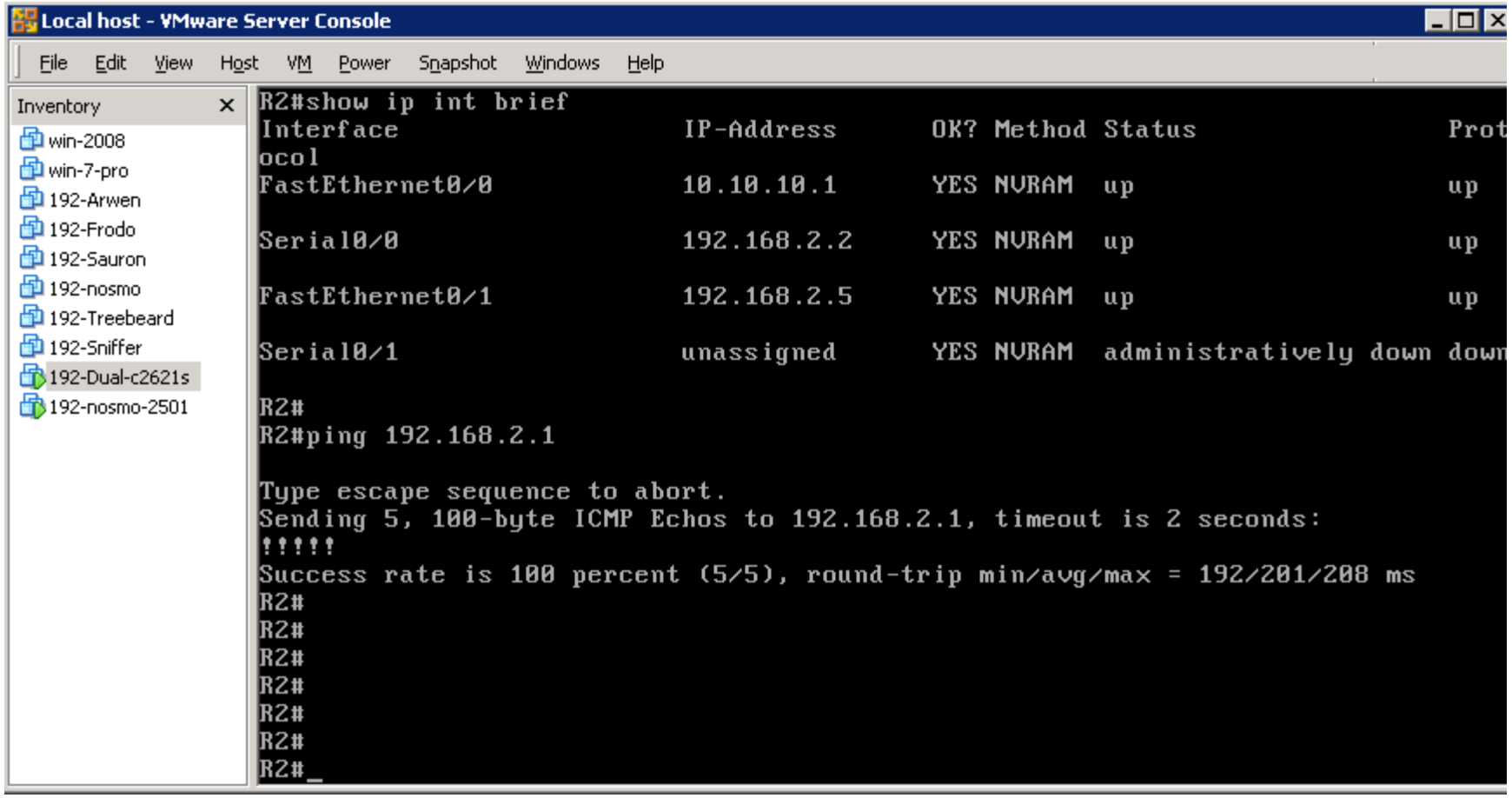
*Use **telnet localhost 2000** command to get to the R1 console (using a separate virtual terminal is handy)*

The Dual-c2621s VM



*Use **telnet localhost 2001** command to get to the R2 console (using a separate virtual terminal is handy)*

The Dual-c2621s VM



You can use the Cisco IOS commands now and the interfaces can be connected to other VMs or to your physical network!

Routing Review

Class Activity

Use CCC Confer White Board

Routing Summary

sign post

```
[root@lilly ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.10.15.48     0.0.0.0         255.255.255.240 U     0      0        0 eth1
172.30.1.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.1.1      0.0.0.0         UG    0      0        0 eth0
[root@lilly ~]#
```

routing table

- Routers operate at **layer 3** and make decisions on where to send a packet.
- Routers use the **routing table** to decide where to forward a packet.
- If there is no route for a packet's destination, the packet is dropped

Example Routing Table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Reading and understanding routing tables is absolutely ***critical***

The Routing Algorithm

(How the decision is made)

Routing Algorithm

The purpose of the Routing Algorithm is to get the packet to its destination network.

1. Compute the route destination network address for the destination IP address
2. Does the destination network match any routes to a directly connected network?
3. Does the destination network match one or more non-directly connected network routes listed in the routing table?
4. Is there a default route listed in the routing table?

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
```

The Routing Algorithm

(How the decision is made)

Routing Algorithm

The purpose of the Routing Algorithm is to get the packet to its destination network.

1. Compute the route destination network address for the destination IP address ? *Apply each **genmask** in the routing table to the destination IP address in the packet*
2. Does the destination network match any routes to a directly connected network? *If so, packet has arrived, send it out the interface for that network*
3. Does the destination network match one or more non-directly connected network routes listed in the routing table? *If so, packet has not yet arrived at its destination, send it to the next hop **gateway** using the appropriate interface. If more than one route matches, select the best match (largest **genmask**).*
4. Is there a default route listed in the routing table? *If so, use that **gateway.** Otherwise, drop the packet - "network is unreachable"*

Compute the route destination network address

The destination network is obtained by applying the genmask to the IP destination address in the packet.

Example: Destination IP=192.168.3.200 and genmask=255.255.255.0

- By hand

  128 64 32 16 8 4 2 1

```
110000 10101000 00000011 11001000  192.168.3.200
111111 11111111 11111111 00000000  255.255.255.0
110000 10101000 00000011 00000000  192.168.3.0
```

- With ipcalc

```
[root@elrond ~]# ipcalc -n 192.168.3.200 255.255.255.0
NETWORK=192.168.3.0
```

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
```

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

*These routes are to **directly connected networks**. Note there is no need for a **gateway**, aka next hop router, to get to these networks.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

*These routes are **NOT directly connected networks**. Packets must travel via a **gateway,** aka next hop router, to get to the destination network.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Translation:

- *Packets going to a destination host network of **172.30.4.0/24** have arrived!*
- *Proceed out the door labeled **eth0** and locate the destination host on that directly attached network.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Translation:

- *Packets going to a destination host network of **192.168.3.0/24** have **NOT** arrived!*
- *Proceed out the door labeled **eth1**, locate the next hop router at **192.168.2.123** and ask for more routing instructions there.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Translation:

- *Packets going to a destination host network of **192.168.2.0/24** have arrived!*
- *Proceed out the door labeled **eth1** and locate the destination host on that directly attached network.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Translation:

- *Packets going to a destination host network of **169.254.0.0/16** have arrived!*
- *Proceed out the door labeled **eth1** and locate the destination host on that directly attached network.*

Reading the routing table

Routing Table

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

Translation:

- *Packets going to a destination for **any other networks** have **NOT** arrived!*
- *Proceed out the door labeled **eth0**, locate the next hop router at **172.30.4.1** and ask for more routing directions there.*

Configuring the Routing Table

- Directly connected networks are automatically added to the routing table.
- APIPA routes are automatically added to the routing table.
- Default gateways can be **manually** added using the route command or added to a configuration file used by the network service. *(Lab 3)*
- Static routes can be **manually** added using the route command or added to a configuration file used by the network service. *(Lab 3)*
- Dynamic routing services that use routing protocols like RIP and OSPF can add **automatically** add routes to the routing table. *(Lab 4)*

The Routing Table
Supernetting

Routing Table

```
root@frodo:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.3.0     172.30.1.125    255.255.255.0   UG    0      0        0 eth0
172.30.1.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.2.0     172.30.1.125    255.255.255.0   UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 eth0
0.0.0.0         172.30.1.1      0.0.0.0         UG    100    0        0 eth0
root@frodo:~#
```

*Note: these two routes could be replaced with a single route for **192.168.2.0/23** or a broader **192.168.0.0/16** for Lab 3. This is supernetting (the reverse of subnetting)*

route command -n option

show route table with names

```
[root@elrond ~]# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      *               255.255.255.0   U     0      0        0 eth0
192.168.3.0     legolas         255.255.255.0   UG    0      0        0 eth1
192.168.2.0     *               255.255.255.0   U     0      0        0 eth1
169.254.0.0     *               255.255.0.0     U     0      0        0 eth1
default         nosmo           0.0.0.0         UG    0      0        0 eth0
```

show route table with IP addresses

```
[root@elrond ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.30.4.0      0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.3.0     192.168.2.123   255.255.255.0   UG    0      0        0 eth1
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         172.30.4.1      0.0.0.0         UG    0      0        0 eth0
[root@elrond ~]#
```

route command for viewing cache

show route table cache with names

```
[root@elrond ~]# route -C
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
192.168.2.125   sauron          legolas                0      0        0 eth1
172.30.4.125    nosmo           nosmo                  0      0        0 eth0
172.30.4.125    nosmo           nosmo                  0      0        6 eth0
sauron          192.168.2.125   192.168.2.125   l      0      0        1 lo
frodo           172.30.4.125    172.30.4.125    il     0      0        1 lo
172.30.4.108    172.30.4.255    172.30.4.255    ibl    0      0        0 lo
172.30.4.103    172.30.4.125    172.30.4.125    il     0      0      105 lo
nosmo           172.30.4.125    172.30.4.125    il     0      0        5 lo
172.30.4.125    172.30.4.103    172.30.4.103           0      1        0 eth0
legolas         192.168.2.125   192.168.2.125   il     0      0        0 lo
172.30.4.125    frodo           frodo                  0      0        0 eth0
172.30.4.125    frodo           frodo                  0      0        1 eth0
172.30.4.10     172.30.4.255    172.30.4.255    ibl    0      0       10 lo
192.168.2.125   sauron          legolas                0      0        2 eth1
172.30.4.12     255.255.255.255 255.255.255.255 ibl    0      0        3 lo
172.30.4.10     172.30.4.255    172.30.4.255    ibl    0      0       10 lo
192.168.2.125   sauron          legolas                0      0        2 eth1
172.30.4.12     255.255.255.255 255.255.255.255 ibl    0      0        3 lo
[root@elrond ~]#
```

route command for viewing cache

show route table cache with IP addresses

```
[root@elrond ~]# route -Cn
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
192.168.2.125   192.168.3.200   192.168.2.123           0      0        0 eth1
172.30.4.125    172.30.4.1      172.30.4.1              0      0        0 eth0
172.30.4.125    172.30.4.1      172.30.4.1              0      0        6 eth0
192.168.3.200   192.168.2.125   192.168.2.125   l       0      0        1 lo
172.30.4.150    172.30.4.125    172.30.4.125    il      0      0        1 lo
172.30.4.108    172.30.4.255    172.30.4.255    ibl     0      0        0 lo
172.30.4.103    172.30.4.125    172.30.4.125    il      0      0      119 lo
172.30.4.125    207.62.187.53   172.30.4.1              0      0        7 eth0
172.30.4.1      172.30.4.125    172.30.4.125    il      0      0        5 lo
172.30.4.106    172.30.4.255    172.30.4.255    ibl     0      0        0 lo
172.30.4.110    172.30.4.255    172.30.4.255    ibl     0      0        0 lo
207.62.187.53   172.30.4.125    172.30.4.125    l       0      0        7 lo
172.30.4.125    172.30.4.103    172.30.4.103            0      1        0 eth0
192.168.2.123   192.168.2.125   192.168.2.125   il      0      0        0 lo
172.30.4.125    172.30.4.150    172.30.4.150            0      0        0 eth0
172.30.4.125    207.62.187.53   172.30.4.1              0      0        7 eth0
172.30.4.125    172.30.4.150    172.30.4.150            0      0        1 eth0
172.30.4.10     172.30.4.255    172.30.4.255    ibl     0      0       14 lo
192.168.2.125   192.168.3.200   192.168.2.123           0      0        2 eth1
172.30.4.12     255.255.255.255 255.255.255.255 ibl     0      0        5 lo
[root@elrond ~]#
```

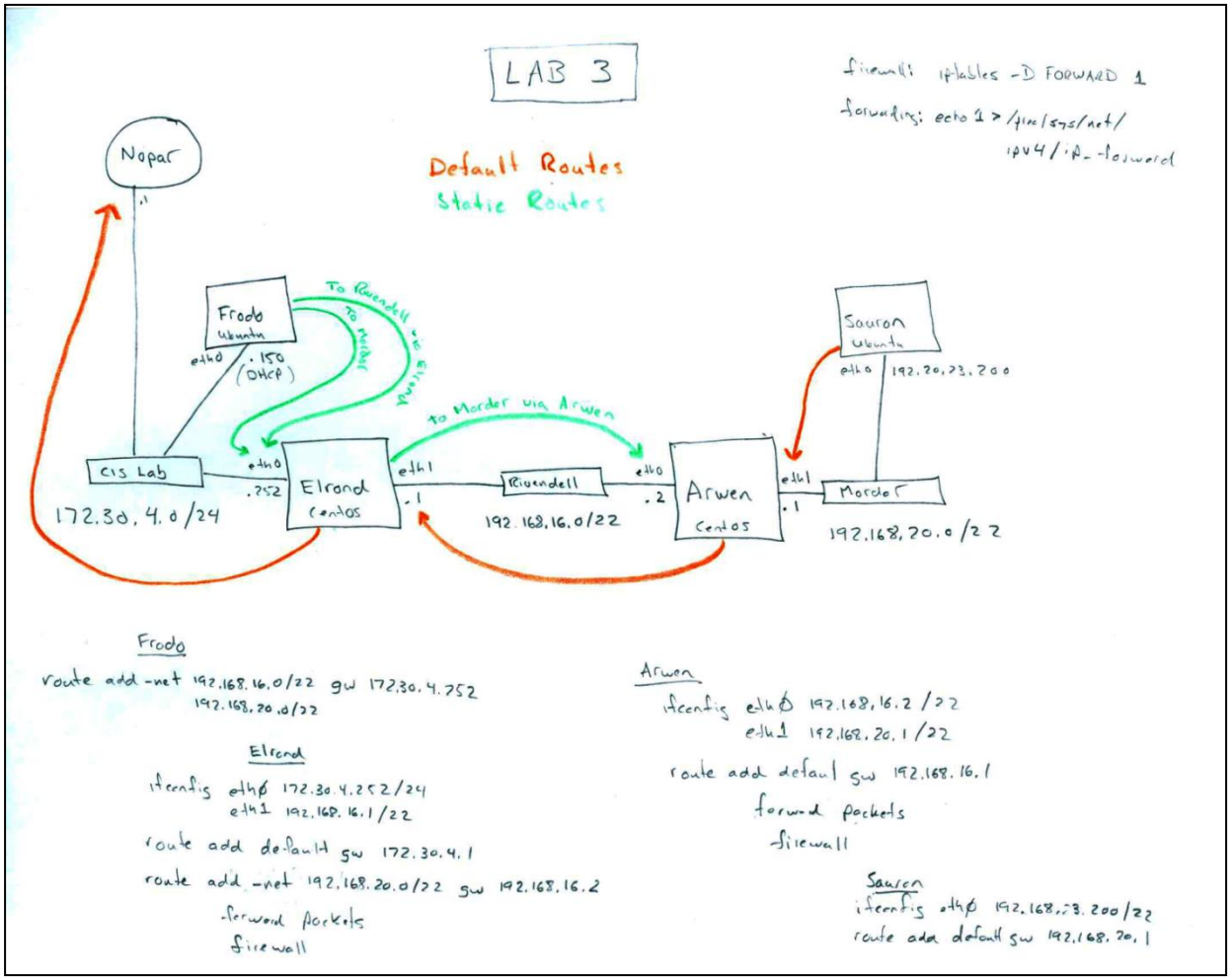
route command
flushing the cache

Flush the route cache

```
[root@elrond ~]# ip route flush cache
[root@elrond ~]# route -C
Kernel IP routing cache
Source          Destination     Gateway         Flags Metric Ref    Use Iface
172.30.4.103    172.30.4.125    172.30.4.125    il    0      0        3 lo
172.30.4.125    172.30.4.103    172.30.4.103          0      1        0 eth0
buttercup.cabri 172.30.4.125    172.30.4.125    l     0      0        1 lo
172.30.4.103    172.30.4.125    172.30.4.125    il    0      0        4 lo
172.30.4.125    172.30.4.103    172.30.4.103          0      1        0 eth0
[root@elrond ~]#
```

*Note: Use **route -CF** on Red Hat 9*

Default and Static Routes Configuration



The orange default routes led traffic towards the Internet because we don't wan to have to create a static route for every network in the world!

Green static routes direct traffic towards our private networks.

*If this was a larger and more complex network manually adding routes would get very **tedious** and **problematic**!*

Class Activity

Use CCC Confer White Board

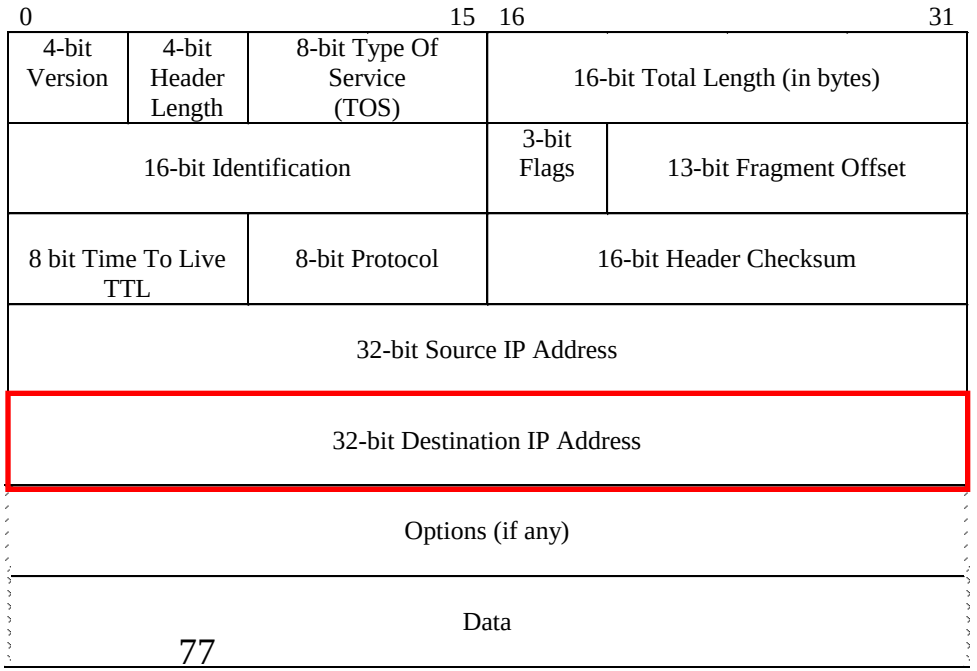
Dynamic Routing Protocols

Routed Protocol

- IP is a routed protocol
- A routed protocol is a layer 3 protocol that contains network addressing information.
- This network addressing information is used by routers to determine the which interface, which next router, to forward this packet.

Note that the subnet mask does not travel with the packet.

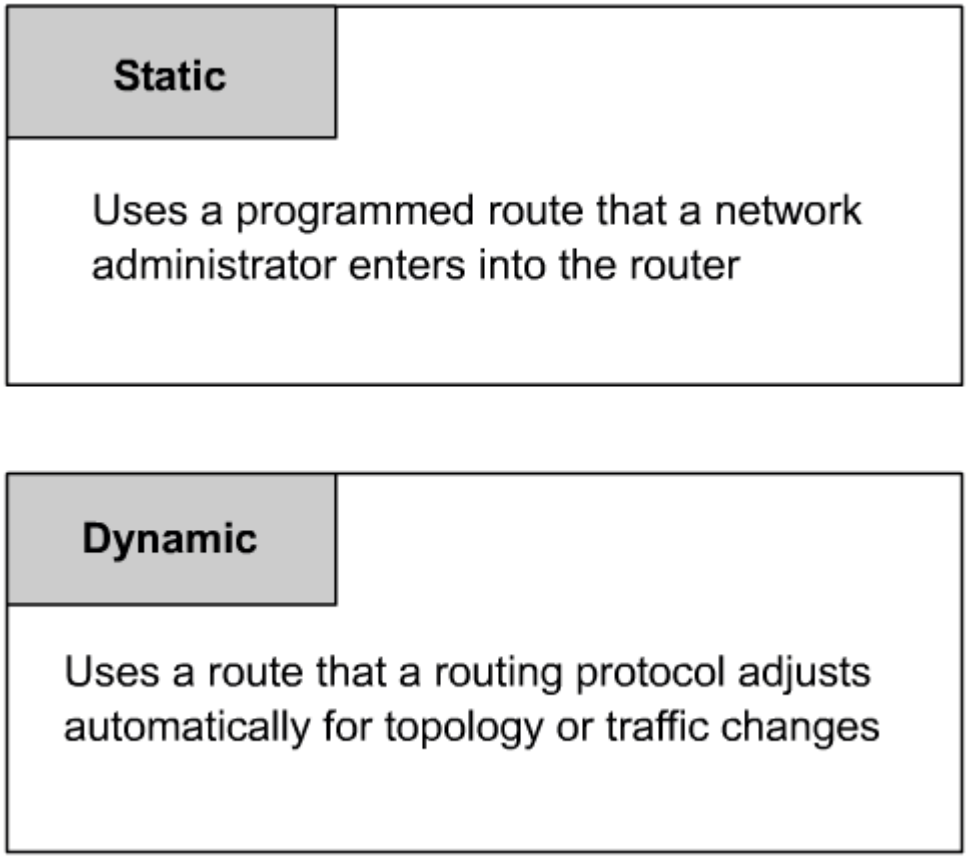
IP Header



Rick Graziani
graziani@cabrillo.edu

Routing Types

- A router must learn about non-directly connected networks either statically or dynamically.
- **Directly connected networks** are networks that the router is connected to, has an IP address/mask.
- **Non-directly connected networks** are remote networks connected to other routers.

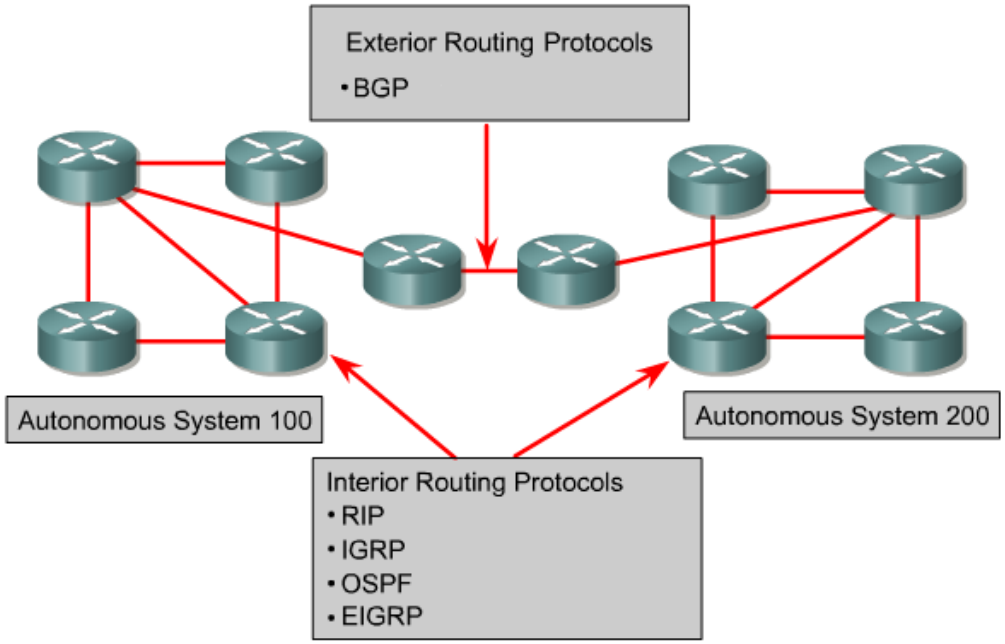


Note, for Lab 3 we had to add static routes manually on the CIS Lab hosts so that they could reach the non-directly connected Rivendell and Mordor networks.

Dynamic vs static routing

- For very small networks, static routes provide a quick and easy method to set up the routing tables.
- In Lab 3, static routes were used to reach the two inner private networks from the CIS Lab hosts.
- As the number of networks grow and change, it becomes increasingly difficult to maintain routing tables using only static routes. With 10's or 100's of routers the setup and ongoing administration can quickly become a nightmare.
- At a certain point the investment in setting up dynamic routing becomes very attractive.
- We will set up dynamic routing in Lab 4.

Routing Protocols



"An AS is a connected group of one or more IP prefixes run by one or more network operators which has a SINGLE and CLEARLY DEFINED routing policy." (RFC 1930)

ISPs and large organizations are assigned a unique ASN (Autonomous System Number) for use with BGP routing.

- **RIP** - A distance vector interior routing protocol
- **IGRP** - Cisco's distance vector interior routing protocol
- **OSPF and IS-IS** - A link-state interior routing protocol
- **EIGRP** - Cisco's advanced distance vector interior routing protocol
- **BGP** - A distance vector exterior routing protocol

Routing Protocols

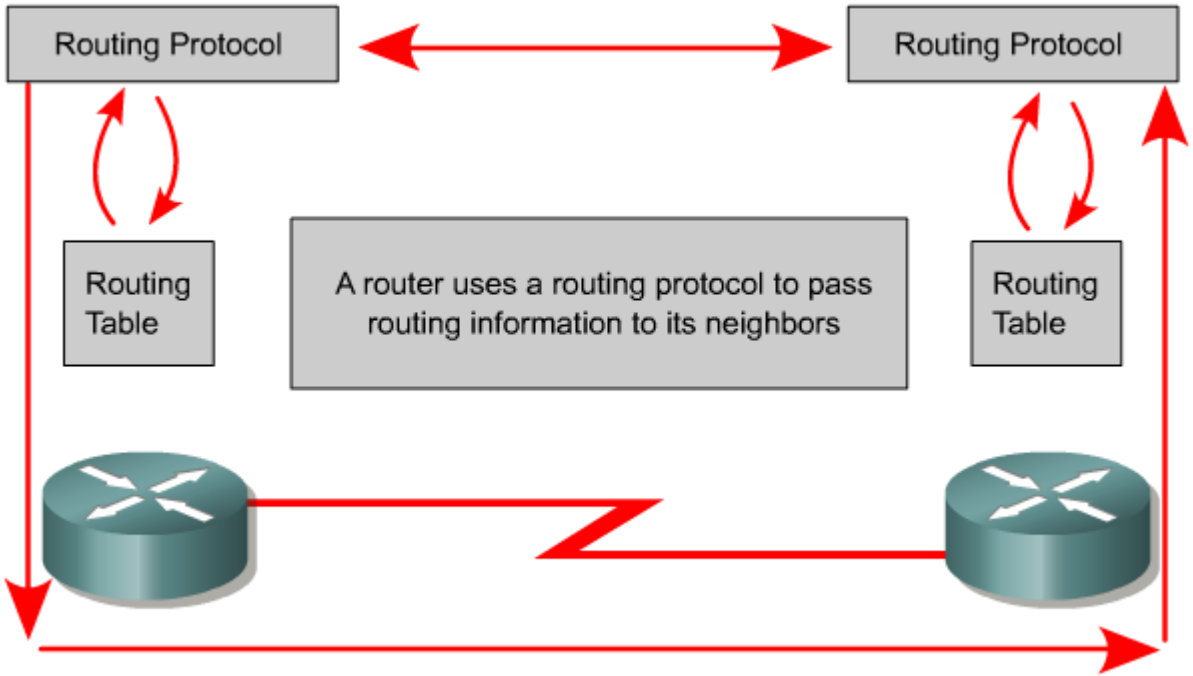
*After doing lab 3 can you imagine **manually** setting up and maintaining static routes on dozens or evens hundreds of routers!*

- Protocols used by routers to build routing tables.
- Routing tables are used by routers to forward packets.
  - **RIP**
  - **IGRP** and **EIGRP**
  - **OSPF**
  - **IS-IS**
  - **BGP**

These are major routing protocols you will learn about in the Cabrillo Cisco networking classes.

*These protocols allow routers to talk to each other and **automatically** configure the routing tables with remote network routes*

Routing Protocols - CIS 82 / CST 312



*The whole idea is to automate making correct **routing tables** without the need to manually set static routes on multiple routers.*

- The goal of a routing protocol is to build and maintain the routing table.
- This table contains the learned networks and associated ports for those networks.
- Routers use routing protocols to manage information received from other routers, information learned from the configuration of its own interfaces, along with manually configured routes.

Linux Implementations

Some dynamic routing software options

- **routed** - an early and widespread RIPv1 implementation
- **gated** - multiple routing protocols (no longer open source)
- **zebra** - GNU licensed (BGP-4, RIPv1, RIPv2, OSPFv2)
- **quagga** - Fork of zebra (BGPv4+, RIPv1, RIPv2, RIPng, OSPFv2, OSPFv3)

Software Installation Tip for Labs

Installing Software on a VM that is not connected to the Internet

Just cable it temporarily to the CIS Lab network and use dhclient to get an IP address

1. Use **ifconfig eth0 down**
2. Re-cable eth0 to the CIS Lab network.
3. Use **dhclient -v eth0** to join the CIS Lab network[1].
4. Use **yum install** *whatever*
5. Use **dhclient -r eth0** to release DHCP address.
6. Use **ifconfig eth0 down**
7. Re-cable eth0 back to the previous network.
8. Use **service network restart** to restore static IP settings again.

[1] I've noticed that **dhclient** on the newer CentOS distros will ignore the default gateway from the DHCP server if a different one is specified in /etc/sysconfig/networks. If this happens use **route add default gw 172.20.0.1** to add it manually

Installing Software on a VM that is not connected to the Internet

- *Bringing down the currently configured interface*
- *Re-cable the interface to the CIS Lab network*
- *Using DHCP to get an IP address*

```
[root@legolas ~]# ifconfig eth0 down
[root@legolas ~]# dhclient eth0
Internet Systems Consortium DHCP Client V3.0.5-RedHat
Copyright 2004-2006 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

Listening on LPF/eth0/00:0c:29:f9:1c:9c
Sending on   LPF/eth0/00:0c:29:f9:1c:9c
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 3
DHCPOFFER from 172.30.4.10
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 172.30.4.10
cp: cannot stat `/etc/resolv.conf': No such file or directory
bound to 172.30.4.155 -- renewal in 2804 seconds.
[root@legolas ~]# _
```

Installing Software on a VM that is not connected to the Internet

- *Release DHCP address with **dhclient -r***

```
[root@legolas ~]# dhclient -r
Internet Systems Consortium DHCP Client V3.0.5-RedHat
Copyright 2004-2006 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

Listening on LPF/eth1/00:0c:29:f9:1c:a6
Sending on   LPF/eth1/00:0c:29:f9:1c:a6
Listening on LPF/eth0/00:0c:29:f9:1c:9c
Sending on   LPF/eth0/00:0c:29:f9:1c:9c
Sending on   Socket/fallback
DHCPRELEASE on eth0 to 172.30.4.10 port 67
[root@legolas ~]# _
```

- *Re-cable VM back into your lab network*
- *Use **service network restart** to restore previous "permanent" static settings or redo manually if done using temporary method*

10 Steps for installing Network Service

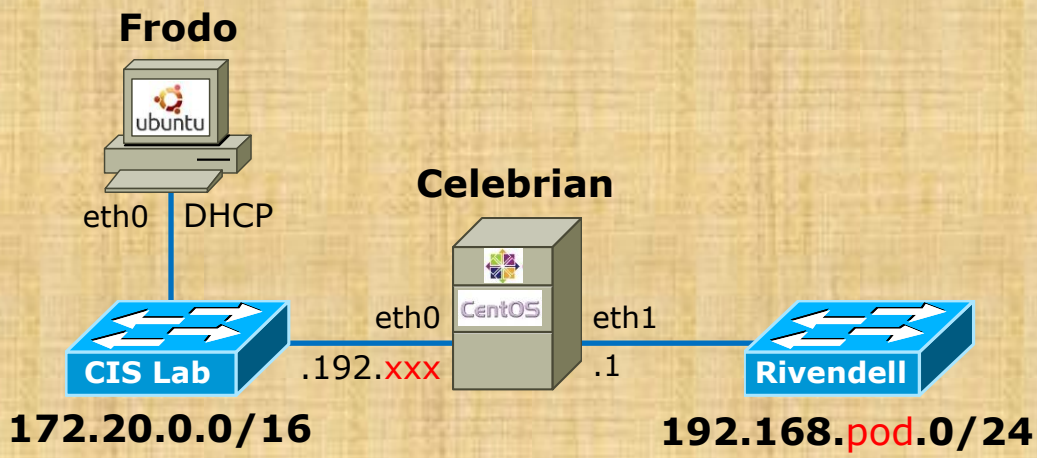
Service Applications

Steps to installing network services

1. Install software package using **yum**, **rpm** or build from source code
2. Customize service's configuration file
3. Modify the firewall to allow access to the service
4. Customize SELinux context settings to allow use
5. Start the service
6. Configure service to automatically start when system boots
7. Monitor and verify service is running
8. Troubleshoot as necessary
9. Monitor log files as appropriate
10. Configure additional security

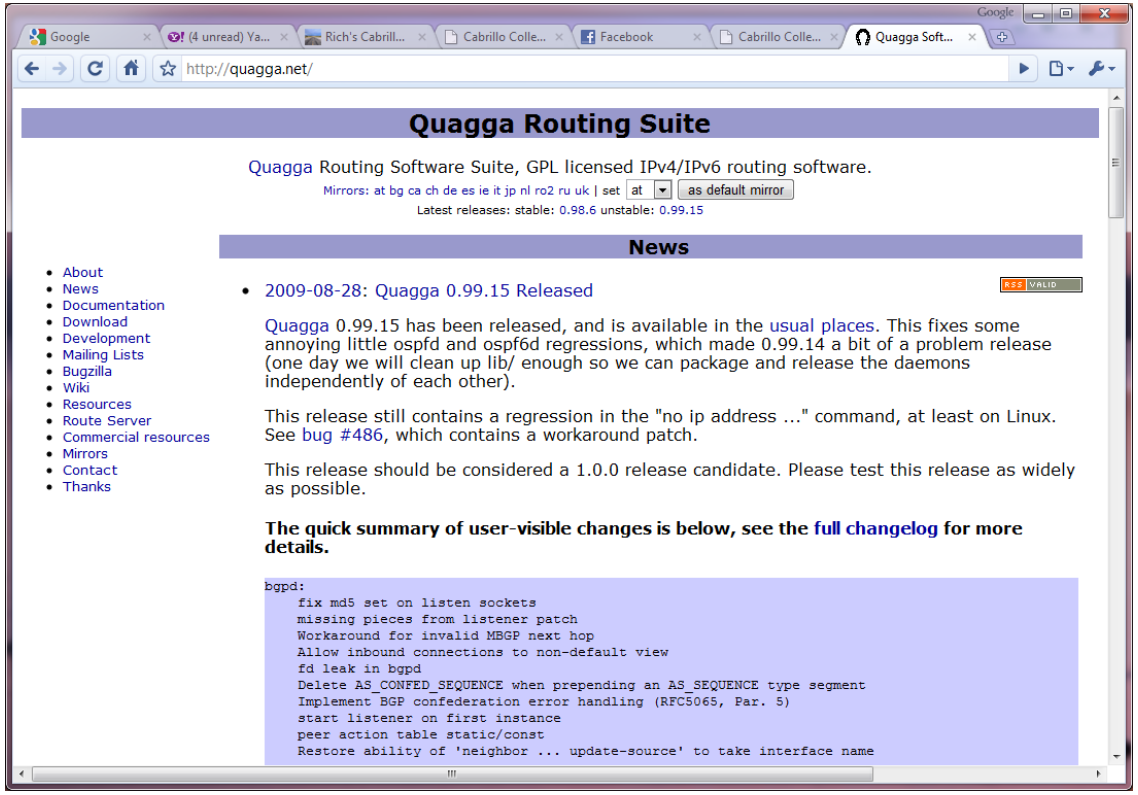
Class Activity

Build this please!



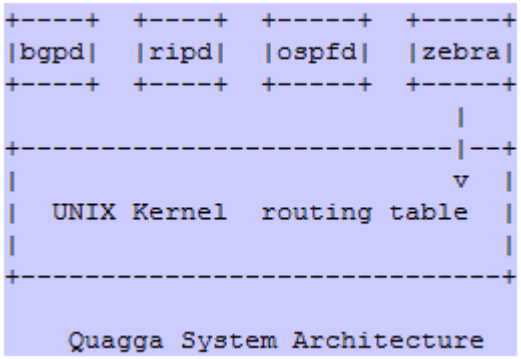
Installing Quagga

Quagga - A fork of GNU Zebra
http://quagga.net/



The CLI is remarkably *similar to some other routing software we study here at Cabrillo!*

Note: There are a number of recipes for using Quagga in the LINUX Networking Cookbook by Carla Schroeder (O'Reilly)



Quagga - Overview

- yum installable
- Quagga has multiple daemons (services).
- They can be used like typical Linux services where you edit the configuration files in /etc and then use the **service** and **chkconfig** commands to control running the services.
- Each Quagga daemon or service (like zebra and ripd) also have individual UI shells.
- You can also use vtysh as an integrated shell for all the daemons.

With some initial testing using the Dual-2621's VM both Cisco and Quagga implementations of OSPF talk to each other - the beauty of standards!

Installing Quagga

Step 1 *Install software*

```
[root@celebrian ~]# rpm -qa | grep quagga
[root@celebrian ~]#
```

The server package "quagga" has not yet been installed.

Installing Quagga

Step 1 *Install software with yum*

```
[root@celebrian ~]# yum install quagga
Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: mirrors.versaweb.com
 * extras: mirrors.usc.edu
 * updates: ftp.osuosl.org
base                                                     | 3.7 kB     00:00
extras                                                   | 3.0 kB     00:00
updates                                                  | 3.5 kB     00:00
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package quagga.i686 0:0.99.15-5.el6_0.2 set to be updated
--> Processing Dependency: libnetsnmp.so.20 for package: quagga-0.99.15-5.el6_0.2.i686
--> Processing Dependency: net-snmp for package: quagga-0.99.15-5.el6_0.2.i686
--> Running transaction check
---> Package net-snmp.i686 1:5.5-27.el6_0.1 set to be updated
--> Processing Dependency: libsensors.so.4 for package: 1:net-snmp-5.5-27.el6_0.1.i686
---> Package net-snmp-libs.i686 1:5.5-27.el6_0.1 set to be updated
--> Running transaction check
---> Package lm_sensors-libs.i686 0:3.1.1-10.el6 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

Installing Quagga

```
Dependencies Resolved

================================================================================
 Package                Arch       Version                    Repository     Size
================================================================================
Installing:
 quagga                 i686       0.99.15-5.el6_0.2          updates       1.0 M
Installing for dependencies:
 lm_sensors-libs        i686       3.1.1-10.el6               base           36 k
 net-snmp               i686       1:5.5-27.el6_0.1           updates       297 k
 net-snmp-libs          i686       1:5.5-27.el6_0.1           updates       1.5 M

Transaction Summary
================================================================================
Install       4 Package(s)
Upgrade       0 Package(s)

Total download size: 2.8 M
Installed size: 11 M
Is this ok [y/N]: y
```

Installing Quagga

```
Is this ok [y/N]: y
Downloading Packages:
(1/4): lm_sensors-libs-3.1.1-10.el6.i686.rpm              |  36 kB     00:00
(2/4): net-snmp-5.5-27.el6_0.1.i686.rpm                   | 297 kB     00:00
(3/4): net-snmp-libs-5.5-27.el6_0.1.i686.rpm              | 1.5 MB     00:00
(4/4): quagga-0.99.15-5.el6_0.2.i686.rpm                  | 1.0 MB     00:00
--------------------------------------------------------------------------------
Total                                           2.2 MB/s | 2.8 MB     00:01
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing     : lm_sensors-libs-3.1.1-10.el6.i686                      1/4
  Installing     : 1:net-snmp-libs-5.5-27.el6_0.1.i686                    2/4
  Installing     : 1:net-snmp-5.5-27.el6_0.1.i686                         3/4
  Installing     : quagga-0.99.15-5.el6_0.2.i686                          4/4

Installed:
  quagga.i686 0:0.99.15-5.el6_0.2

Dependency Installed:
  lm_sensors-libs.i686 0:3.1.1-10.el6       net-snmp.i686 1:5.5-27.el6_0.1
  net-snmp-libs.i686 1:5.5-27.el6_0.1

Complete!
[root@celebrian ~]#
```

Installing Quagga

```
[root@celebrian ~]# rpm -qa | grep quagga
quagga-0.99.15-5.el6_0.2.i686
[root@celebrian ~]#
```

Quagga has been installed

Note, you can use **yum** command to only download rpms (and not install them) with the downloadonly option. Useful for doing installations on systems with no Internet access.

yum install yum-downloadonly
yum install quagga --downloadonly

The downloaded rpms will be found in /var/cache/yum//packages*

Installing Quagga

```
[root@celebrian ~]# rpm -qi quagga
Name        : quagga                       Relocations: (not relocatable)
Version     : 0.99.15                           Vendor: CentOS
Release     : 5.el6_0.2                     Build Date: Sat 25 Jun 2011 05:15:32 AM PDT
Install Date: Tue 15 Nov 2011 06:40:56 AM PST      Build Host: c6b5.bsys.dev.centos.org
Group       : System Environment/Daemons    Source RPM: quagga-0.99.15-5.el6_0.2.src.rpm
Size        : 4431645                          License: GPLv2+
Signature   : RSA/8, Tue 05 Jul 2011 06:45:16 PM PDT, Key ID 0946fca2c105b9de
Packager    : CentOS BuildSystem <http://bugs.centos.org>
URL         : http://www.quagga.net
Summary     : Routing daemon
Description :
Quagga is a free software that manages TCP/IP based routing
protocol. It takes multi-server and multi-thread approach to resolve
the current complexity of the Internet.

Quagga supports BGP4, BGP4+, OSPFv2, OSPFv3, RIPv1, RIPv2, and RIPng.

Quagga is intended to be used as a Route Server and a Route Reflector. It is
not a toolkit, it provides full routing power under a new architecture.
Quagga by design has a process for each protocol.

Quagga is a fork of GNU Zebra.
[root@celebrian ~]#
```

The -qi option on rpm gives you a summary of the package

Celebrian

Class Activity

Install quagga and telnet client

- Check if packages have already been installed with
  rpm -qa | grep quagga
  rpm -qa | grep telnet

- Install the quagga and telnet client package with
  yum install quagga telnet

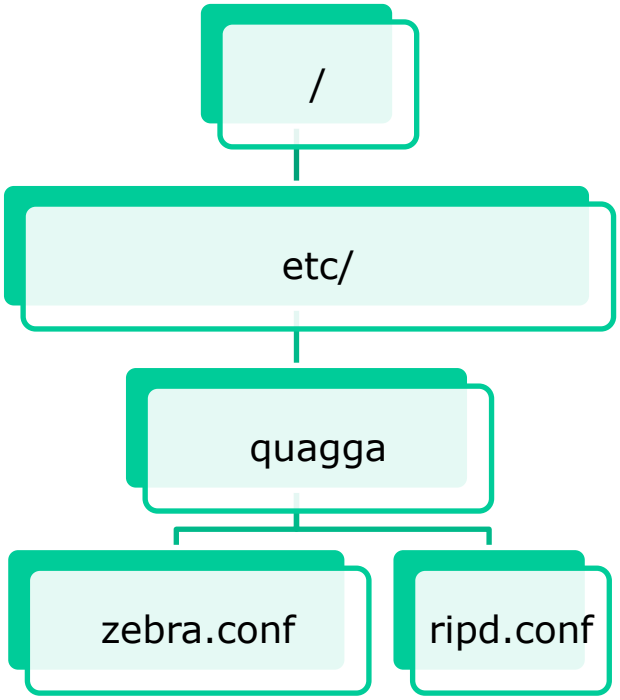
- Check if packages have been installed with
  rpm -qa | grep quagga
  rpm -qa | grep telnet

- To learn more about a package use
  rpm -qi quagga
  rpm -qi telnet

/etc/quagga/zebra.conf and /etc/quagga/ripd.conf

Step 2 *Customize the configuration files*

main configuration files for Quagga when implementing RIPv2



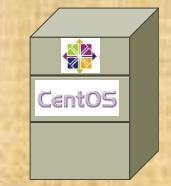
/etc/quagga/zebra.conf and /etc/quagga/ripd.conf

```
[root@celebrian ~]# cat /etc/quagga/zebra.conf
hostname pxx-celebrian
!
password quagga
enable password quagga
!
log file /var/log/quagga/zebra.log
```

```
[root@celebrian ~]# cat /etc/quagga/ripd.conf
hostname pxx-celebrian
log file /var/log/quagga/ripd.log
!
router rip
 network eth1
 redistribute connected
!
line vty
  password quagga
!
```

Class Activity

Configure Quagga



Celebrian

```
[root@celebrian ~]# cat /etc/quagga/zebra.conf
hostname pxx-celebrian
!
password quagga
enable password quagga
!
log file /var/log/quagga/zebra.log
```

```
[root@celebrian ~]# cat /etc/quagga/ripd.conf
hostname pxx-celebrian
log file /var/log/quagga/ripd.log
!
router rip
 network eth1
 redistribute connected
!
line vty
  password quagga
!
```

/etc/quagga/zebra.conf and /etc/quagga/ripd.conf

Set ownership of configuration files

```
[root@celebrian ~]# cd /etc/quagga
[root@celebrian quagga]# chown quagga:quagga ripd.conf zebra.conf

[root@celebrian quagga]# ls -l ripd.conf zebra.conf
-rw-r--r--. 1 quagga quagga 144 Nov 15 07:48 ripd.conf
-rw-r-----. 1 quagga quagga 106 Nov 15 07:47 zebra.conf
```

No Longer Needed :)

Quagga and the Firewall

Step 3 *Modify the firewall*

Firewall ports used for implementing Quagga RIPv2

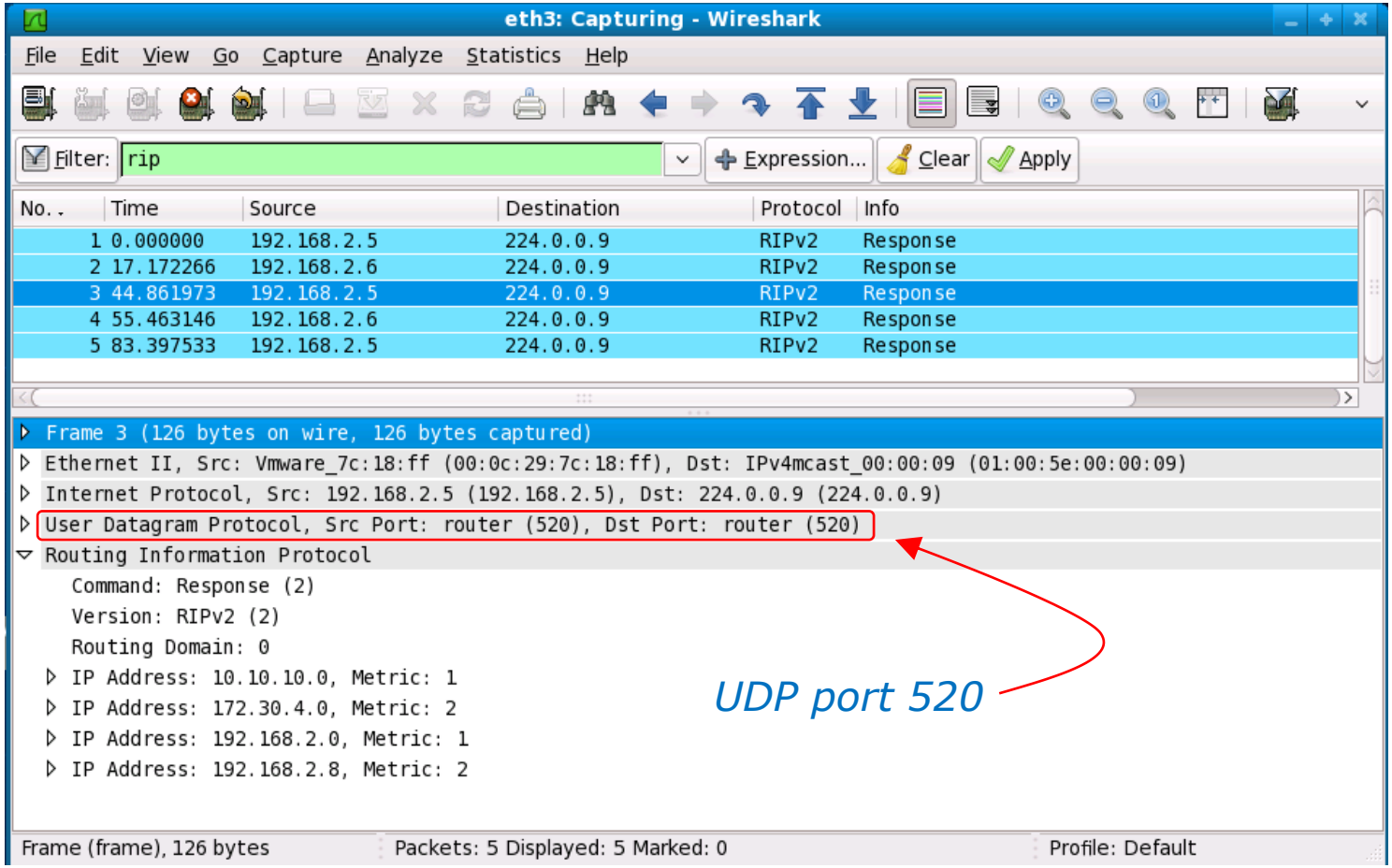
UDP 520 *RIP advertisements*

Other Firewall changes needed for Quagga RIPv2

- Routers should be forwarding packets and not filtering them out.
- In particular the UDP RIP packets must be allowed to pass through the router so they can get to the other routers.

Quagga and the Firewall

We would like RIP updates to be passed between the routers



Quagga and the Firewall

Default firewall (in memory)

```
[root@celebrian ~]# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere             anywhere
3    ACCEPT     all  --  anywhere             anywhere
4    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
5    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
[root@celebrian ~]#
```

- *There is no rule on the INPUT chain to accept incoming RIP packets (UDP port 520) so they will be rejected.*
- *All packets going through the FORWARD chain get rejected.*

Quagga and the Firewall

Default firewall (in configuration file)

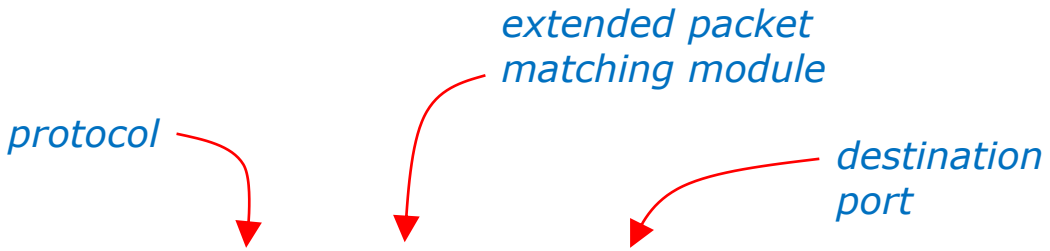
```
[root@celebrian ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@celebrian ~]#
```

- *There is no rule on the INPUT chain to accept incoming RIP packets (UDP port 520) so they will be rejected.*
- *All packets going through the FORWARD chain get rejected.*

Quagga and the Firewall

```
[root@celebrian ~]# iptables -D FORWARD 1
```

Delete the first rule on the FORWARD chain allowing all packets to be forwarded




```
[root@celebrian ~]# iptables -I INPUT 4 -p udp -m udp --dport 520 -j ACCEPT
```

Insert a rule above rule 4 on the INPUT chain to accept incoming packets to UDP port 520

```
[root@celebrian ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]
```

Save the rules in memory to the configuration file

Modifying the Firewall
(Centos)

Modified firewall (in memory)

```
[root@celebrian ~]# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere             anywhere
3    ACCEPT     all  --  anywhere             anywhere
4    ACCEPT     udp  --  anywhere             anywhere            udp dpt:router
5    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
6    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
[root@celebrian ~]#
```

RIP port open

No filtering now on forwarded packets

Quagga and the Firewall

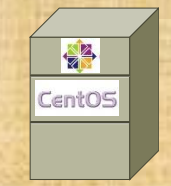
Modified firewall (in configuration file)

```
[root@celebrian ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.4.7 on Tue Nov 15 00:41:40 2011
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [11:1740]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p udp -m udp --dport 520 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
# Completed on Tue Nov 15 00:41:40 2011
[root@celebrian ~]#
```

No filtering now on forwarded packets

RIP port open

Activity



Celebrian

Modify Firewall

```
iptables -L
cat /etc/sysconfig/iptables
```

```
iptables -I INPUT 4 -p udp -m udp --dport 520 -j ACCEPT
iptables -D FORWARD 1
service iptables save
```

```
iptables -L
cat /etc/sysconfig/iptables
```

SELinux

Step 4 *Configure SELinux*

Overview

SELinux is like an internal firewall where you can define what subjects (users, programs) can access which objects (files, devices)

- Originally created by the NSA (National Security Agency)
- Based on the MAC (Mandatory Access Control) concept where administrators control all interactions between programs.
- Programs and users start with no rights. Any rights must be granted by the administrator as part of the security policy for the system.
- Standard UNIX permissions are checked first then SELinux rules are applied if necessary.

SELinux

Security Contexts

Security context have three components: a **user identity**, a **role**, and a **type** (also known as a domain).

```
[root@celebrian quagga]# ls -lZ /etc/quagga/[rz]*.conf
-rw-r--r--. quagga quagga unconfined_u:object_r:zebra_conf_t:s0 /etc/quagga/ripd.conf
-rw-r-----. quagga quagga unconfined_u:object_r:zebra_conf_t:s0 /etc/quagga/zebra.conf
```

This context type above is already correct for quagga configuration files, if you did need to reset it use:

```
cd /etc/quagga
chcon -v --type=zebra_conf_t ripd.conf zebra.conf
```

Managing Quagga Services (CentOS)

Step 5 *Start the service*

```
[root@celebrian ~]# service zebra start
Starting zebra:                                            [  OK  ]
[root@celebrian ~]# service ripd start
Starting ripd:                                             [  OK  ]
[  OK  ]
```

Step 6 *Start the service automatically during system startup*

```
[root@celebrian ~]# chkconfig zebra on
[root@celebrian ~]# chkconfig ripd on


[root@celebrian ~]# chkconfig --list  zebra
zebra           0:off   1:off   2:on    3:on    4:on    5:on    6:off
[root@celebrian ~]# chkconfig --list  ripd
ripd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

Managing Quagga Services (CentOS)

Step 7 *Monitor and verify service is running*

```
[root@celebrian ~]# service zebra status
zebra (pid 6823) is running...

[root@celebrian ~]# service ripd status
ripd (pid 6836) is running...
```

```
[root@celebrian ~]# ps -ef | grep quagga
quagga    6823     1  0 08:19 ?        00:00:00 zebra -d -A 127.0.0.1 -f /etc/quagga/zebra.conf
quagga    6836     1  0 08:19 ?        00:00:00 ripd -d -A 127.0.0.1 -f /etc/quagga/ripd.conf
root      6862  1856  0 08:20 pts/0    00:00:00 grep quagga
```

Before quagga services were started (Lab 4)

```
[root@celebrian ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.8     0.0.0.0         255.255.255.252 U     0      0        0 eth0
192.168.2.4     0.0.0.0         255.255.255.252 U     0      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1003   0        0 eth1
[root@celebrian ~]#
```

After quagga services were started (Lab 4)

```
[root@celebrian ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.8     0.0.0.0         255.255.255.252 U     0      0        0 eth0
192.168.2.0     192.168.2.5     255.255.255.252 UG    2      0        0 eth1
192.168.2.4     0.0.0.0         255.255.255.252 U     0      0        0 eth1
172.30.4.0      192.168.2.10    255.255.255.0   UG    2      0        0 eth0
10.10.10.0      192.168.2.5     255.255.255.0   UG    2      0        0 eth1
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1003   0        0 eth1
0.0.0.0         192.168.2.10    0.0.0.0         UG    2      0        0 eth0
[root@celebrian ~]#
```

Quagga

Step 7 *Monitor and verify service is running*

```
[root@celebrian ~]# netstat -uln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 0.0.0.0:520             0.0.0.0:*
```

UDP port 520 is used for RIP advertisements

Quagga

Step 7 *Monitor and verify service is running*

```
[root@celebrian ~]# netstat -tlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address     State    PID/Program name
tcp        0      0 localhost:discp-client     *:*                 LISTEN   6823/zebra
tcp        0      0 localhost:discp-server     *:*                 LISTEN   6836/ripd
tcp        0      0 *:ssh                      *:*                 LISTEN   1327/sshd
tcp        0      0 localhost:smtp             *:*                 LISTEN   1403/master
tcp        0      0 *:ssh                      *:*                 LISTEN   1327/sshd
tcp        0      0 localhost:smtp             *:*                 LISTEN   1403/master
[root@celebrian ~]#
```

```
[root@celebrian ~]# netstat -tlnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address     State    PID/Program name
tcp        0      0 127.0.0.1:2601             0.0.0.0:*           LISTEN   6823/zebra
tcp        0      0 127.0.0.1:2602             0.0.0.0:*           LISTEN   6836/ripd
tcp        0      0 0.0.0.0:22                 0.0.0.0:*           LISTEN   1327/sshd
tcp        0      0 127.0.0.1:25               0.0.0.0:*           LISTEN   1403/master
tcp        0      0 :::22                      :::*                LISTEN   1327/sshd
tcp        0      0 ::1:25                     :::*                LISTEN   1403/master
[root@celebrian ~]#
```

zebra and ripd daemons are using TCP ports 2601 and 2602

Quagga

Step 8 *Troubleshoot*

If the Quagga shell write command fails in updating the configuration files:

1. Check config files are owned by quagga

2. Check SELinux context type is zebra_conf_t

Step 8 *Troubleshoot*

Quagga

```
legolas(ripd)# debug rip zebra
legolas(ripd)# debug rip event
```

Enable debugging to log RIP events in log file

```
[root@legolas ~]# tail -f /var/log/quagga/ripd.log
2009/02/26 09:12:56 RIP: RECV packet from 192.168.2.1 port 520 on eth0
2009/02/26 09:13:04 RIP: update timer fire!
2009/02/26 09:13:04 RIP: SEND UPDATE to eth0 ifindex 2
2009/02/26 09:13:04 RIP: multicast announce on eth0
2009/02/26 09:13:04 RIP: update routes on interface eth0 ifindex 2
2009/02/26 09:13:04 RIP: SEND to  224.0.0.9.520
2009/02/26 09:13:04 RIP: SEND UPDATE to eth1 ifindex 3
2009/02/26 09:13:04 RIP: multicast announce on eth1
2009/02/26 09:13:04 RIP: update routes on interface eth1 ifindex 3
2009/02/26 09:13:04 RIP: SEND to  224.0.0.9.520
2009/02/26 09:13:24 RIP: RECV packet from 192.168.2.6 port 520 on eth1
2009/02/26 09:13:30 RIP: update timer fire!
2009/02/26 09:13:30 RIP: SEND UPDATE to eth0 ifindex 2
2009/02/26 09:13:30 RIP: multicast announce on eth0
2009/02/26 09:13:30 RIP: update routes on interface eth0 ifindex 2
```

< snipped >

-f option on the tail command shows real-time additions to the log. Use Ctrl-C to end

Quagga

Step 9 *Monitor log files*

```
[root@celebrian ~]# tail /var/log/quagga/zebra.log
2011/11/15 08:19:13 ZEBRA: Zebra 0.99.15 starting: vty@2601
[root@celebrian ~]#
```

```
[root@celebrian ~]# tail /var/log/quagga/ripd.log
2011/11/15 08:38:24 RIP: update timer fire!
2011/11/15 08:38:24 RIP: SEND UPDATE to eth0 ifindex 2
2011/11/15 08:38:24 RIP: multicast announce on eth0
2011/11/15 08:38:24 RIP: update routes on interface eth0 ifindex 2
2011/11/15 08:38:24 RIP: SEND to  224.0.0.9.520
2011/11/15 08:38:24 RIP: SEND UPDATE to eth1 ifindex 3
2011/11/15 08:38:24 RIP: multicast announce on eth1
2011/11/15 08:38:24 RIP: update routes on interface eth1 ifindex 3
2011/11/15 08:38:24 RIP: SEND to  224.0.0.9.520
2011/11/15 08:38:30 RIP: RECV packet from 192.168.2.10 port 520 on eth0
[root@celebrian ~]#
```

Quagga

Step 10 *Configure additional security*

Using Quagga

Quagga - individual routing daemon shells

To use: telnet to localhost port 2601 for zebra or 2602 for ripd.

```
[root@legolas ~]# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
legolas> en
legolas#
```

zebra service

Logging in to the shell

Enable privileged mode

Privileged mode prompt

Quagga - vtysh as an integrated Shell

Or use vtysh for an integrated shell

Show eth0 information

```
[root@legolas quagga]# vtysh

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

legolas.localdomain# sh int eth0
Interface eth0 is up, line protocol detection is disabled
  index 2 metric 1 mtu 1500 <UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:0c:29:7c:18:f5
  inet 192.168.2.2/30 broadcast 192.168.2.3
  inet6 fe80::20c:29ff:fe7c:18f5/64
    input packets 10923, bytes 1096902, dropped 0, multicast packets 0
    input errors 0, length 0, overrun 0, CRC 0, frame 0, fifo 0, missed 0
    output packets 8480, bytes 950760, dropped 0
    output errors 0, aborted 0, carrier 0, fifo 0, heartbeat 0, window 0
    collisions 0
legolas.localdomain#
```

Quagga - A fork of GNU Zebra

```
[root@legolas ~]# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
legolas> en
legolas# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

K>* 0.0.0.0/0 via 192.168.2.1, eth0
C>* 10.10.10.0/24 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
K>* 169.254.0.0/16 is directly connected, eth0
R>* 172.30.4.0/24 [120/2] via 192.168.2.1, eth0, 03:24:42
C>* 192.168.2.0/30 is directly connected, eth0
C>* 192.168.2.4/30 is directly connected, eth1
R>* 192.168.2.8/30 [120/2] via 192.168.2.1, eth0, 03:24:42
legolas#
```

zebra service

Show the routing table

The default gateway shows as a kernel route, each NIC is shown as directly connected, and the other routes were added using RIPv2

Quagga

Show the running configuration in the vtysh or cat the configuration file

Quagga shell

```
celebrian.localdomain# sh run
Building configuration...

Current configuration:
!
hostname arwen.localdomain
log file /var/log/quagga/zebra.log
hostname celebrian.localdomain
log file /var/log/quagga/ripd.log
!
debug rip events
debug rip zebra
!
password quagga
enable password quagga
!
interface eth0
 ipv6 nd suppress-ra
!
interface eth1
 ipv6 nd suppress-ra
!
interface lo
!
interface sit0
 ipv6 nd suppress-ra
end
celebrian.localdomain#
```

Linux shell

```
[root@celebrian ~]# cat /etc/quagga/ripd.conf
!
! Zebra configuration saved from vty
!   2011/11/15 08:52:47
!
hostname celebrian.localdomain
log file /var/log/quagga/ripd.log
!
debug rip events
debug rip zebra
!
router rip
 redistribute connected
 network eth0
 network eth1
!
line vty
!
[root@celebrian ~]#
```

Quagga - A fork of GNU Zebra

Configuration command completion and ? help is similar to other routing software we study at Cabrillo

Enter configuration mode (note that commands and arguments may be abbreviated

Use ? to see what could come next on the command

Command completion with tab

```
legolas# conf t
legolas(config)# hostname R1
R1(config)# hostname legolas
legolas(config)# ip
  forwarding   Turn on IP forwarding
  prefix-list  Build a prefix list
  protocol     Apply route map to PROTO
  route        Establish static routes
legolas(config)# ip forw
legolas(config)# ip forwarding
  <cr>
llegolas(config)# exit
legolas# wr
Building Configuration...
Configuration saved to /etc/quagga/zebra.conf
Configuration saved to /etc/quagga/ripd.conf
[OK]
legolas#
```

Quagga - A fork of GNU Zebra

ripd service

```
[root@legolas ~]# telnet localhost 2602
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
legolas(ripd)> enable
legolas(ripd)#
legolas(ripd)# show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
C(r) 10.10.10.0/24      0.0.0.0               1 self              0
R(n) 172.30.4.0/24      192.168.2.1           2 192.168.2.1       0 02:31
C(i) 192.168.2.0/30     0.0.0.0               1 self              0
C(i) 192.168.2.4/30     0.0.0.0               1 self              0
R(n) 192.168.2.8/30     192.168.2.1           2 192.168.2.1       0 02:31
legolas(ripd)#
```

Using the ripd shell to check RIP information

Show routing table

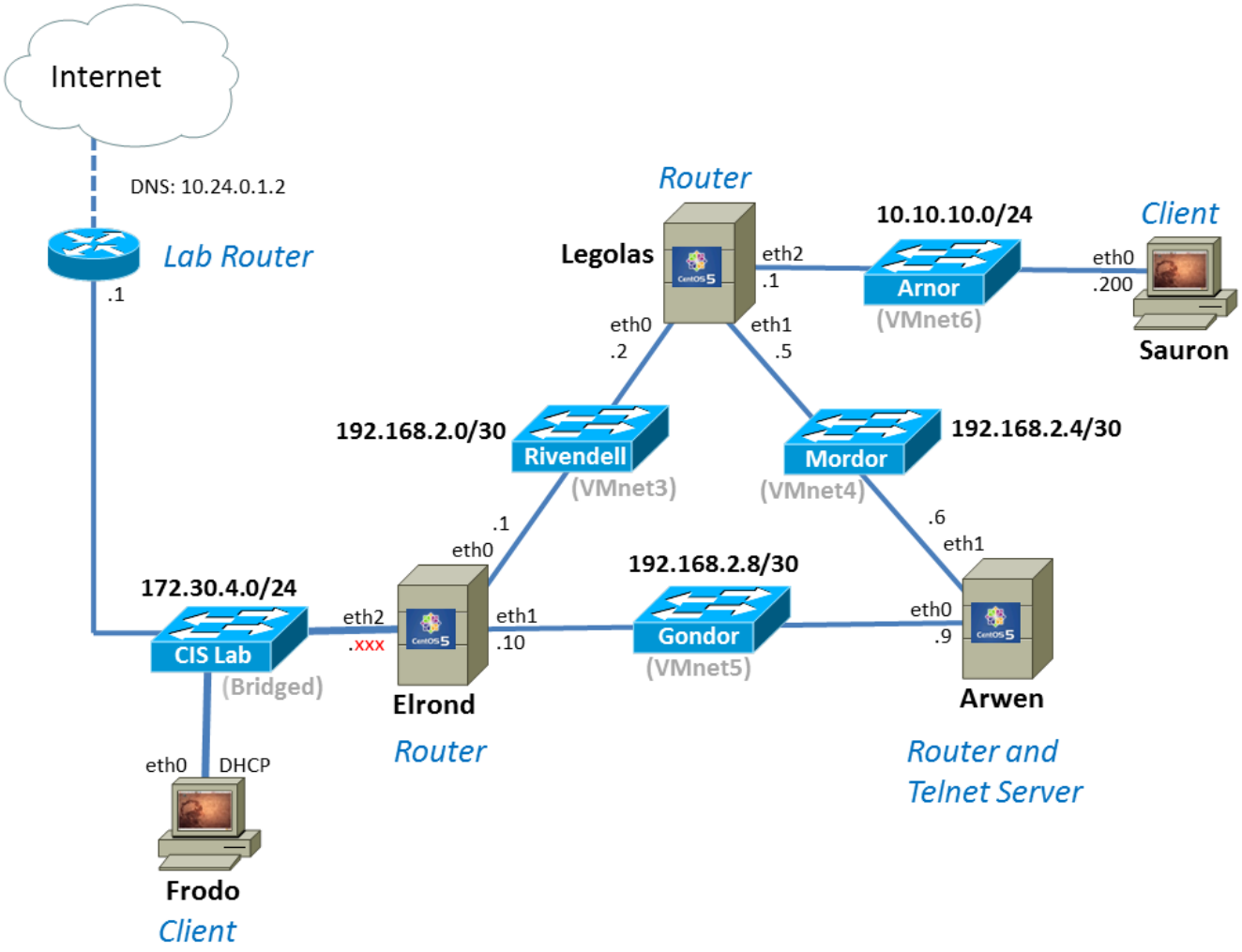
Seeing RIP routes indicates RIP is working between routers

Lab 4 Skills

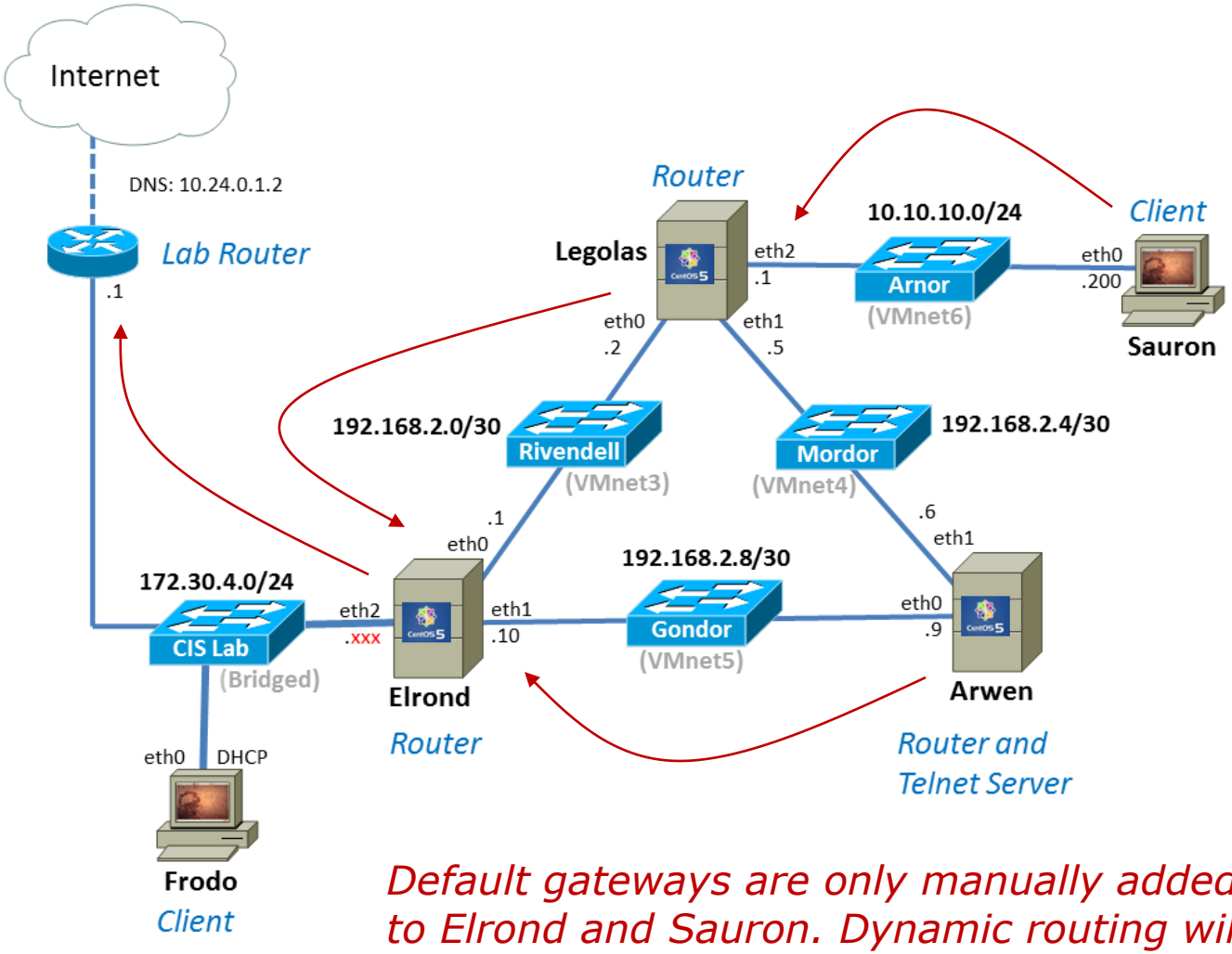
Skills needed for Lab 4!

- Adding NICs
- Changing VMware host memory usage
- Cabling NICs
- Getting the graphical desktop
- Modifying the firewall
- Changing SELinux mode
- Installing software
- Managing daemons
- Using Sniffer s for troublshooting

The network used for Lab 4

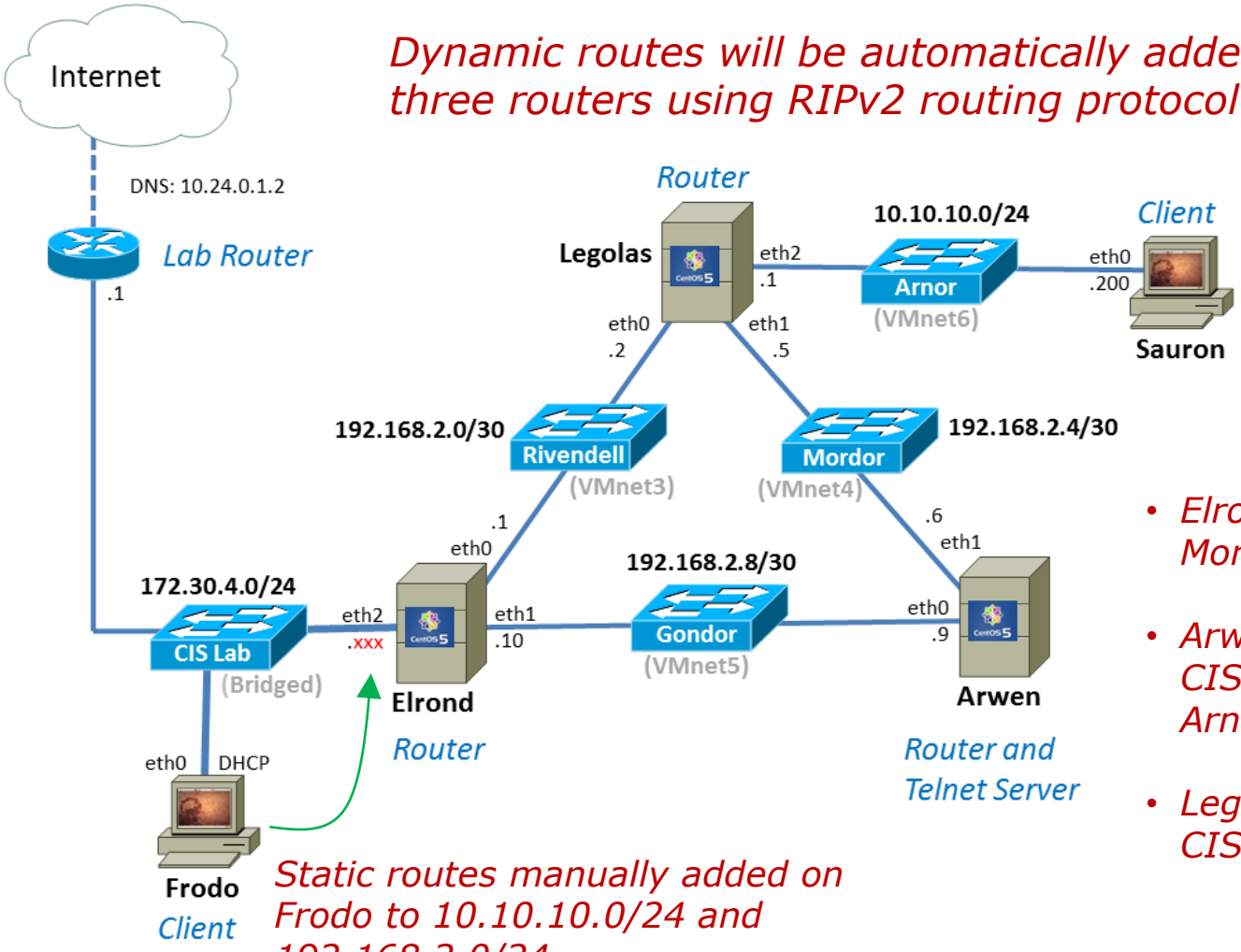


The network used for Lab 4



Default gateways are only manually added to Elrond and Sauron. Dynamic routing will handle adding them to Arwen and Logolas.

The network used for Lab 4



- *Elrond needs routes to Mordor and Arnor*
- *Arwen needs routes to CIS Lab, Rivendell and Arnor*
- *Legolas need routes to CIS Lab and Gondor*

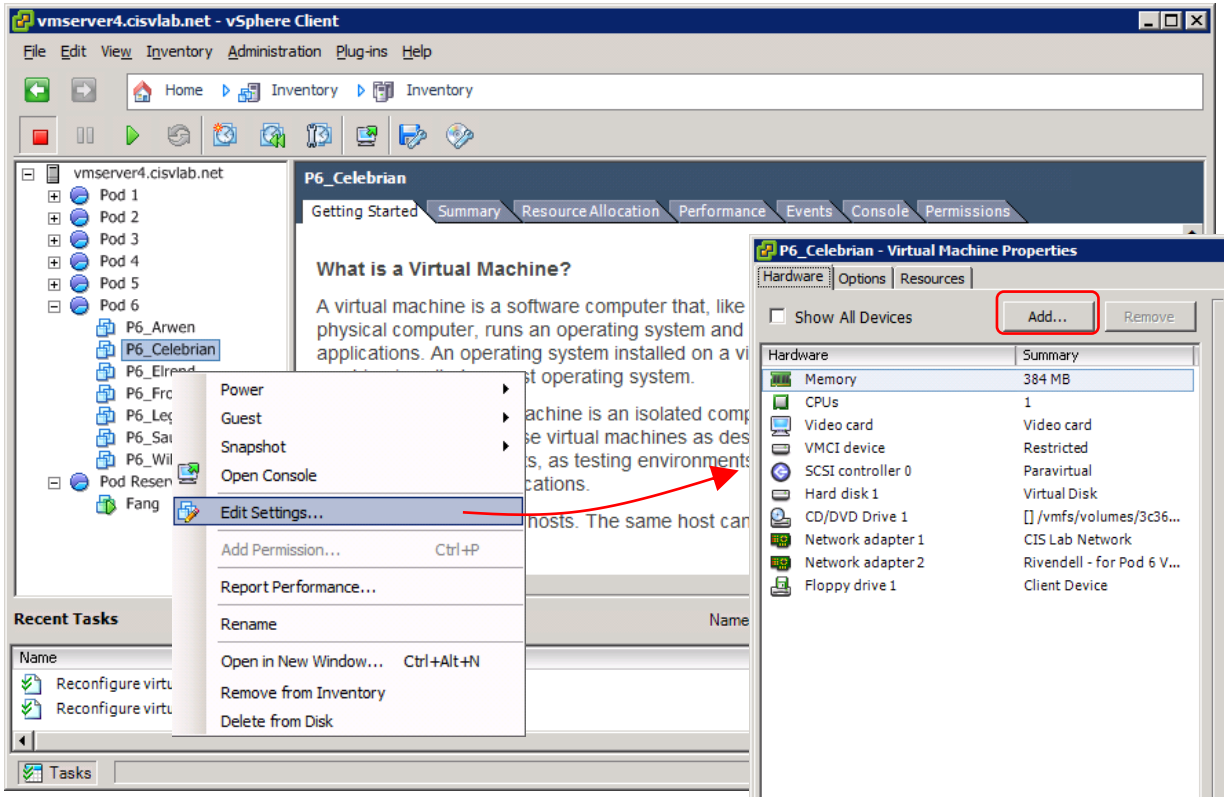
Adding Hardware

Adding another NIC
(Without going to Fry's)

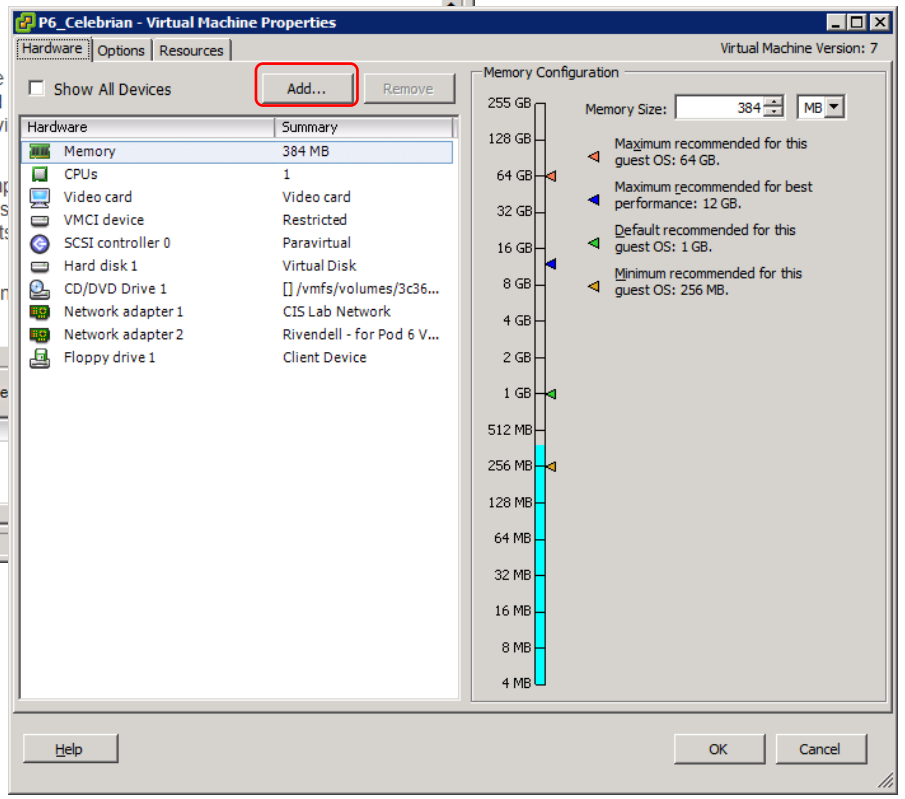
- The VM needs to be powered off
- Start with **(Edit) Settings…**
- Click Add... button to get to the **Add Hardware (Wizard)**
- Add a **(Ethernet) Network Adapter** and keep hitting Next button till added.

Adding another NIC
(VMware ESXi/vSphere)

Right click > Edit Settings...

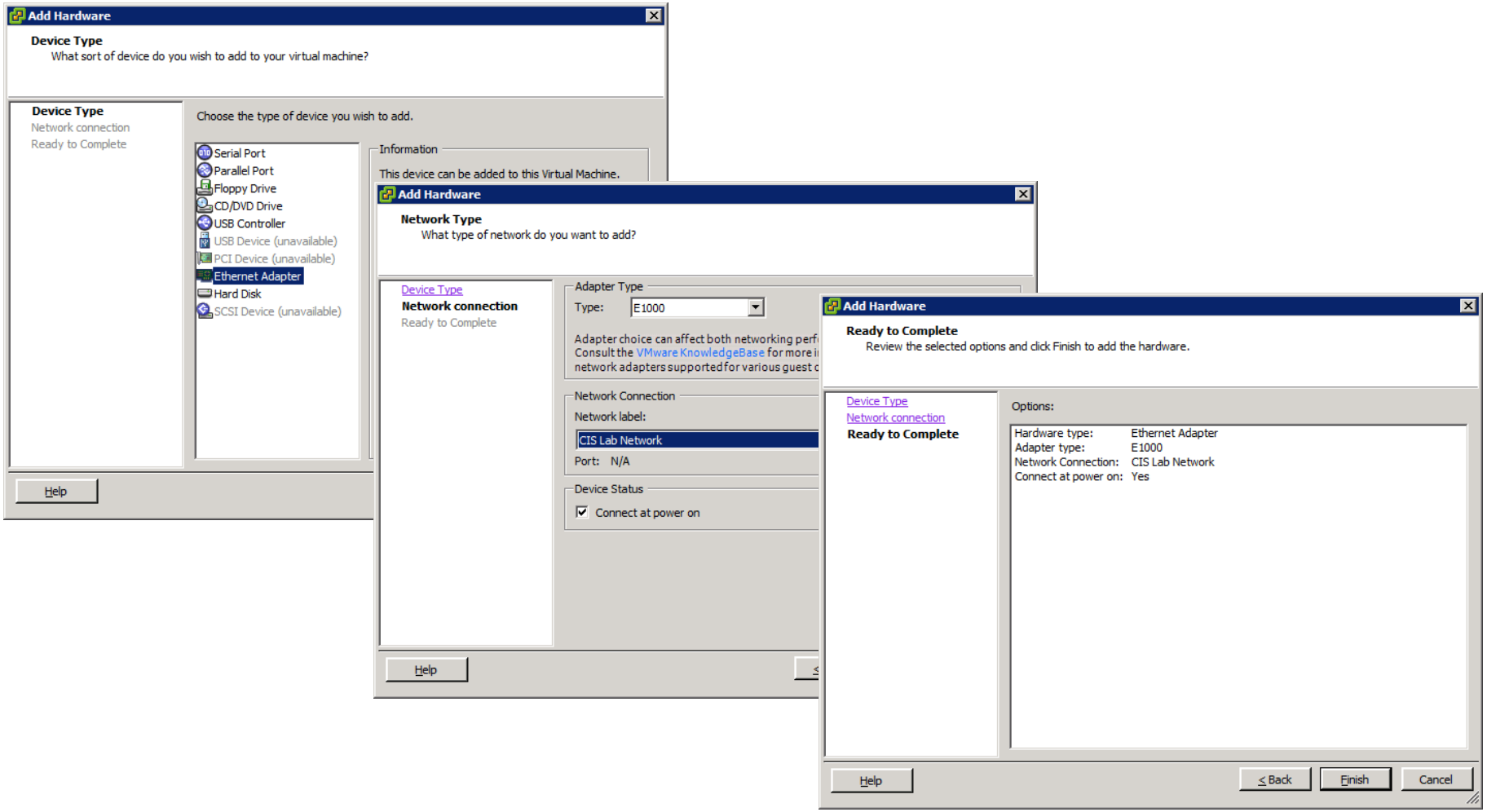


Add...



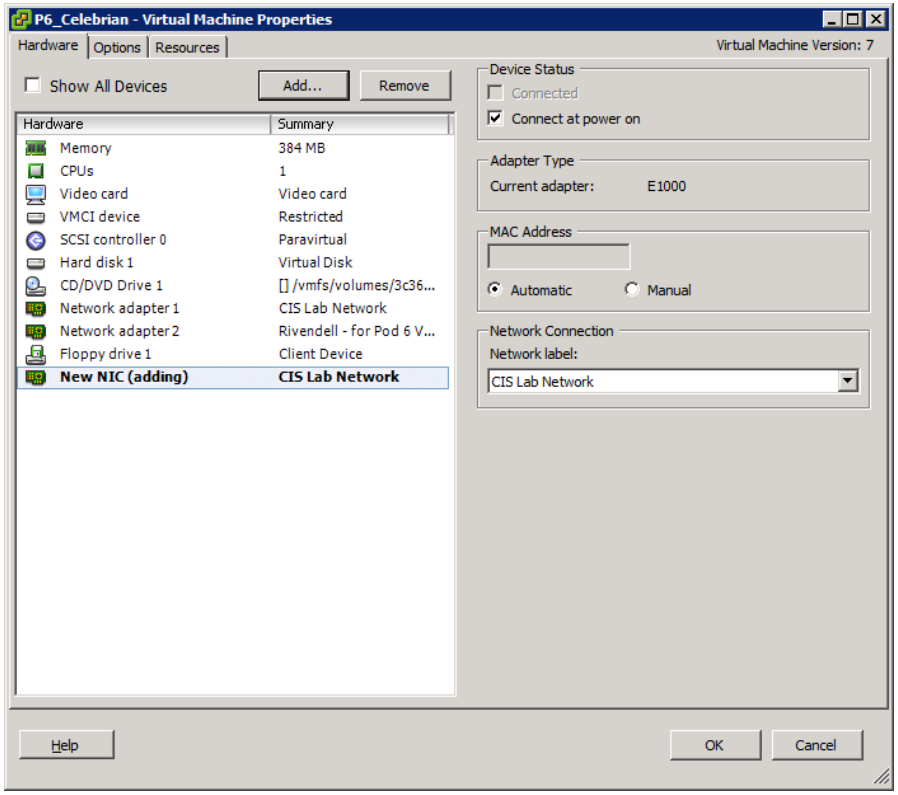
Adding another NIC
(VMware ESXi/vSphere)

Add hardware wizard

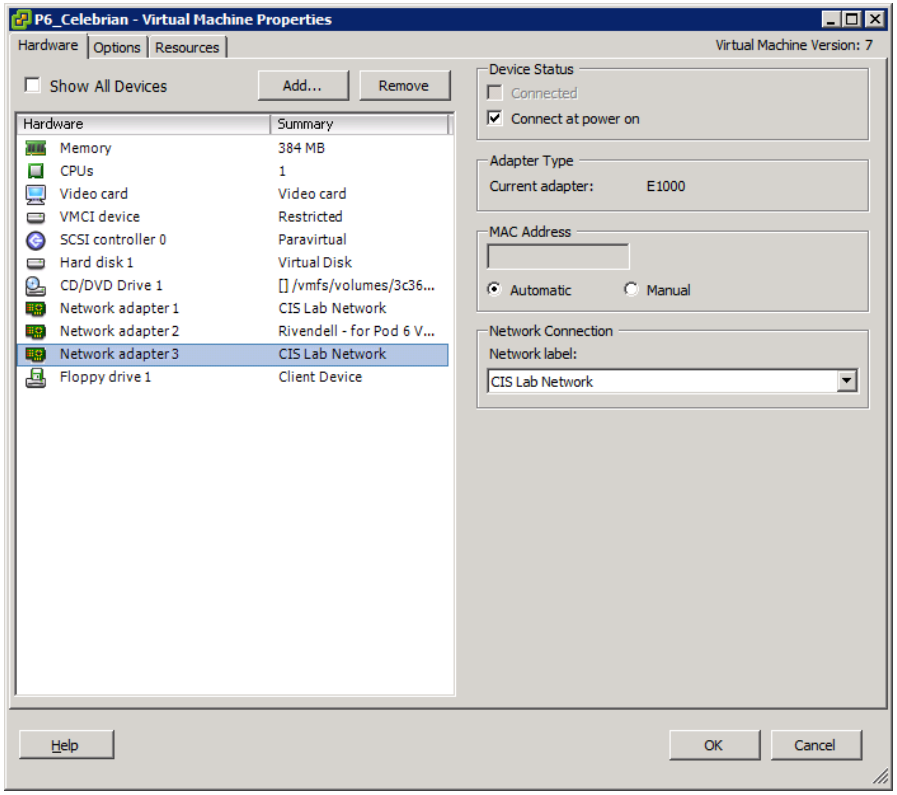


Adding another NIC
(VMware ESXi/vSphere)

Can still be cancelled

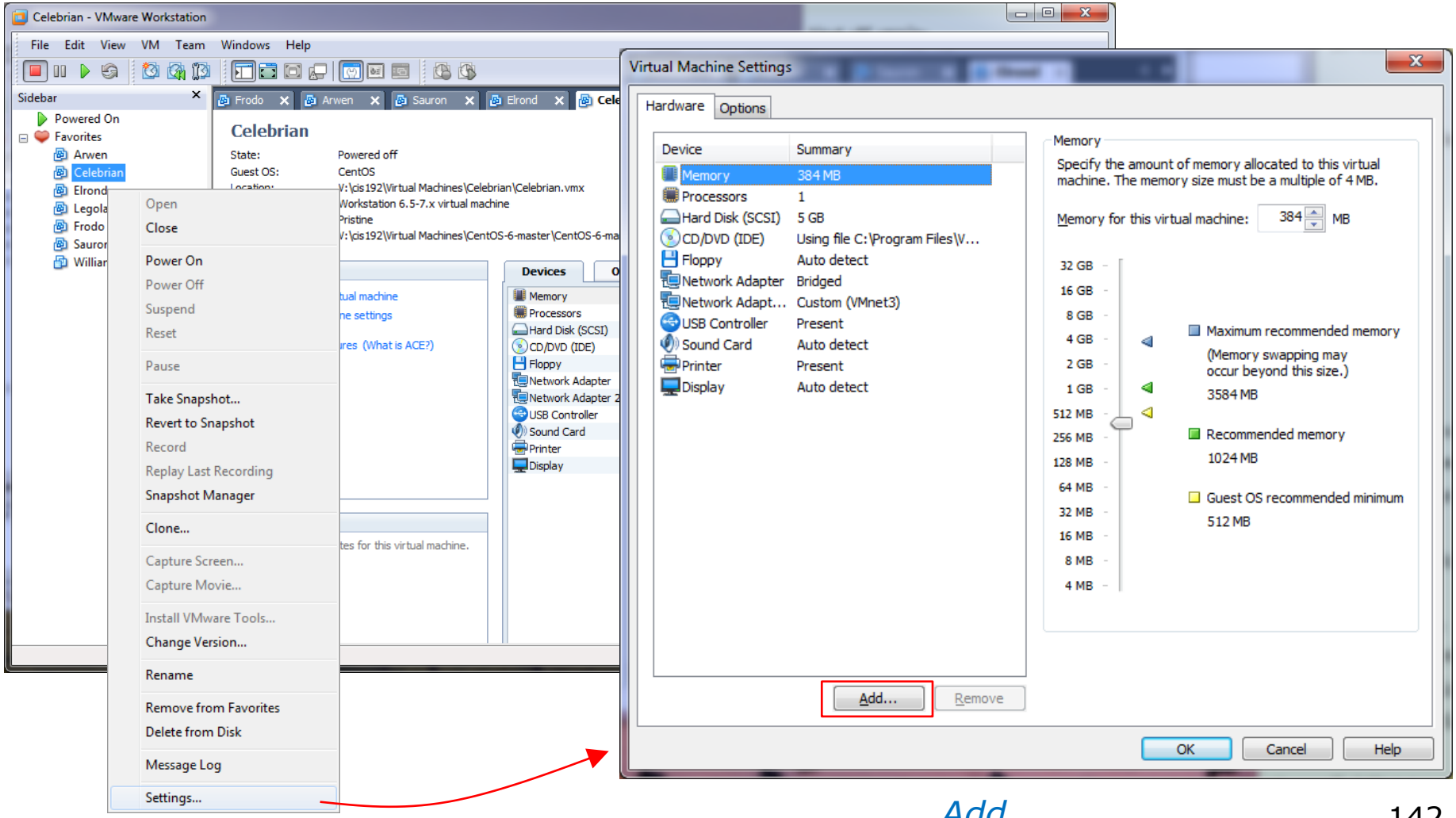


New NIC added



Adding another NIC
(VMware Workstation)

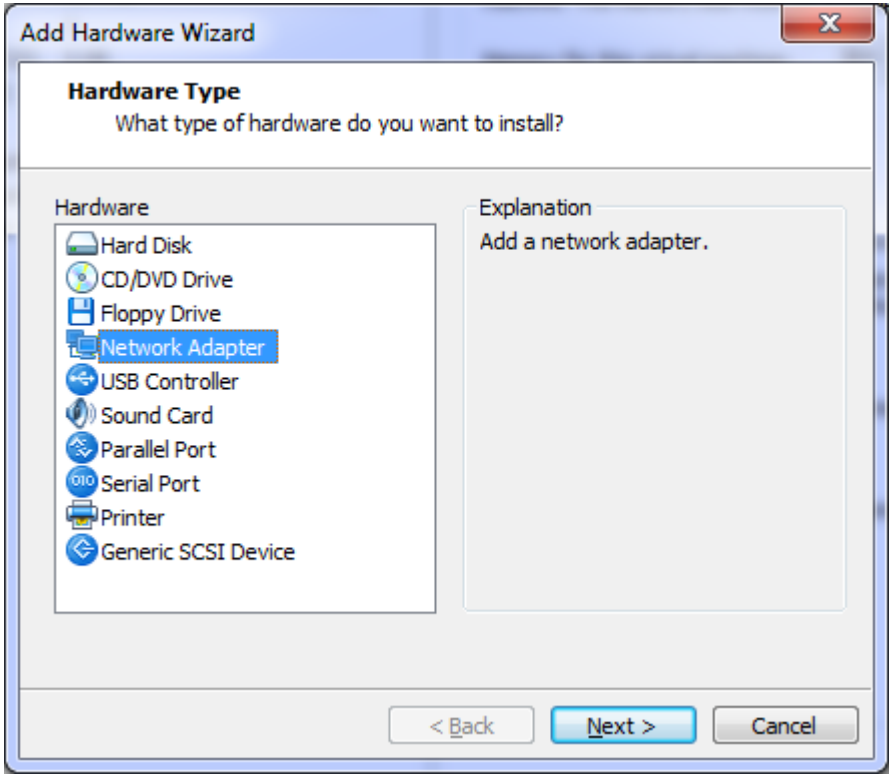
Right click > Settings...

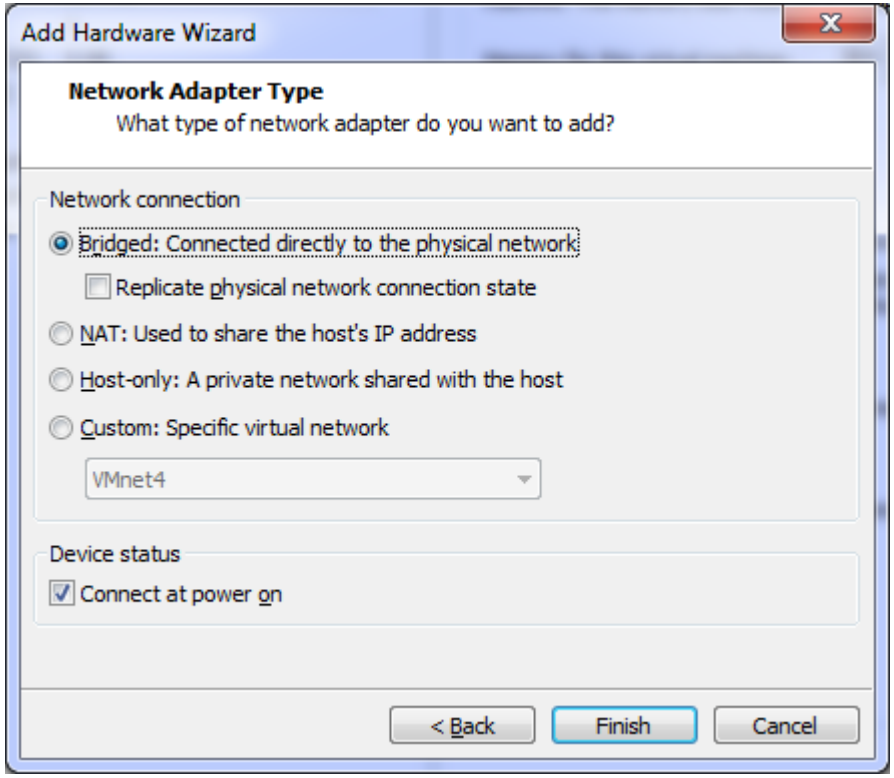


Add...

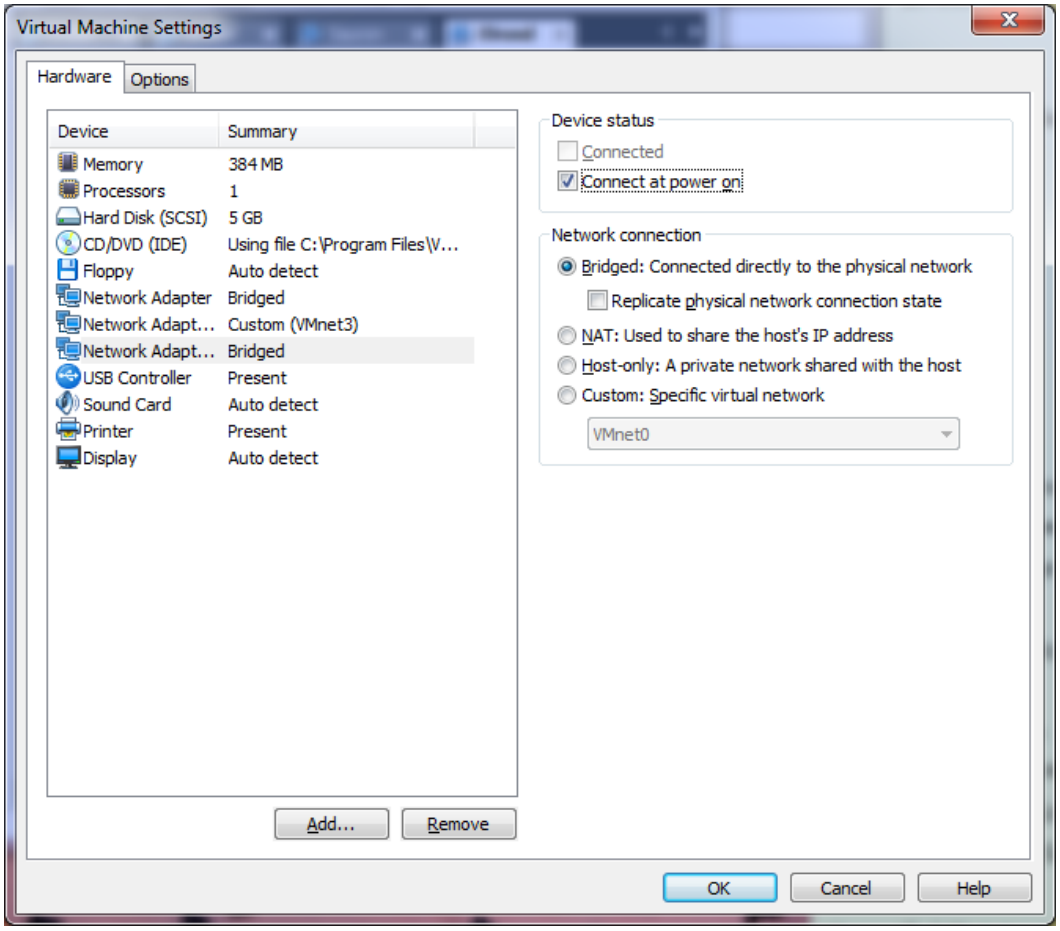
Adding another NIC
(VMware Workstation)

Add hardware wizard





Adding another NIC
(VMware Workstation)



New NIC added

Activity
(Adding new hardware)

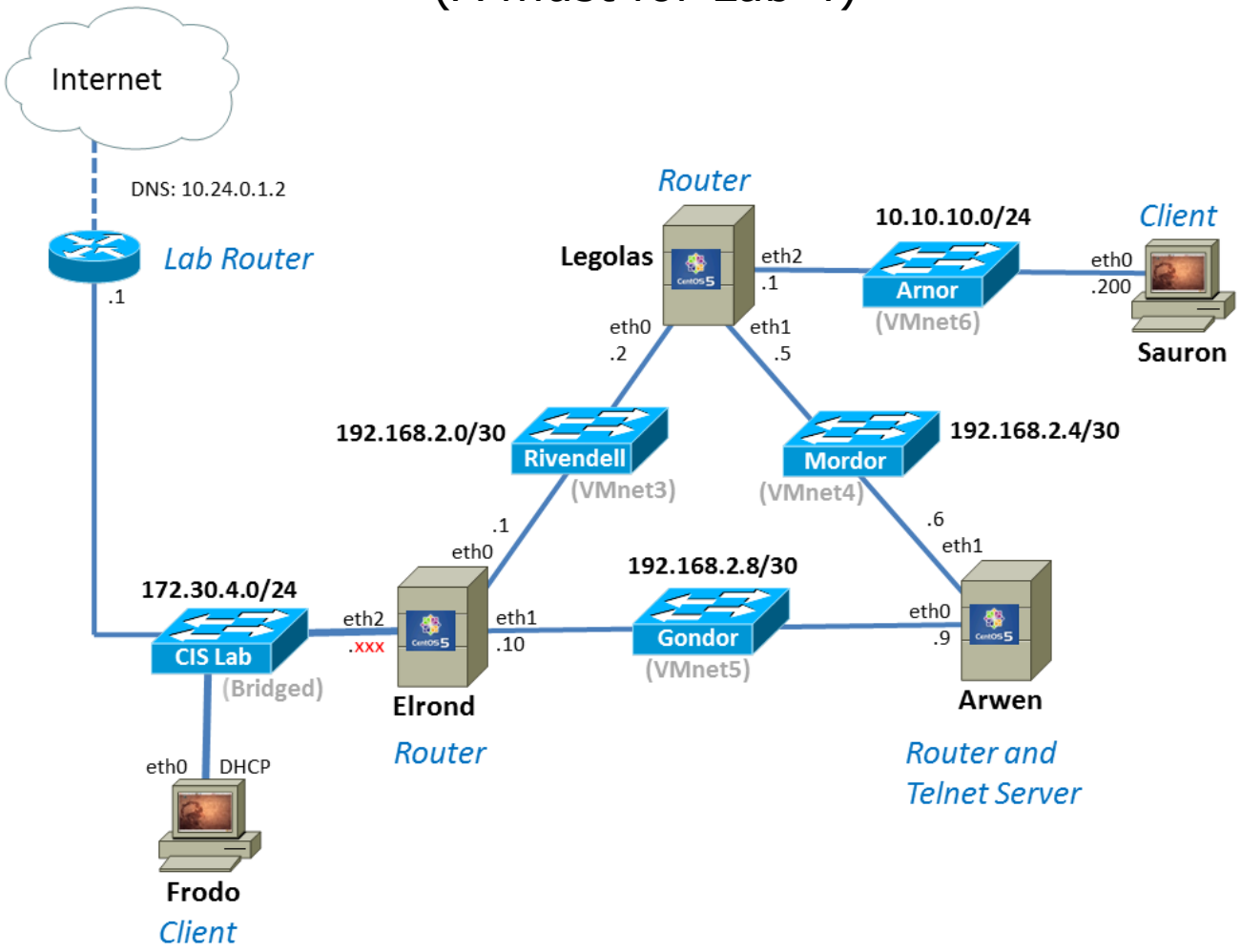
Live Demo

Cabling NICs

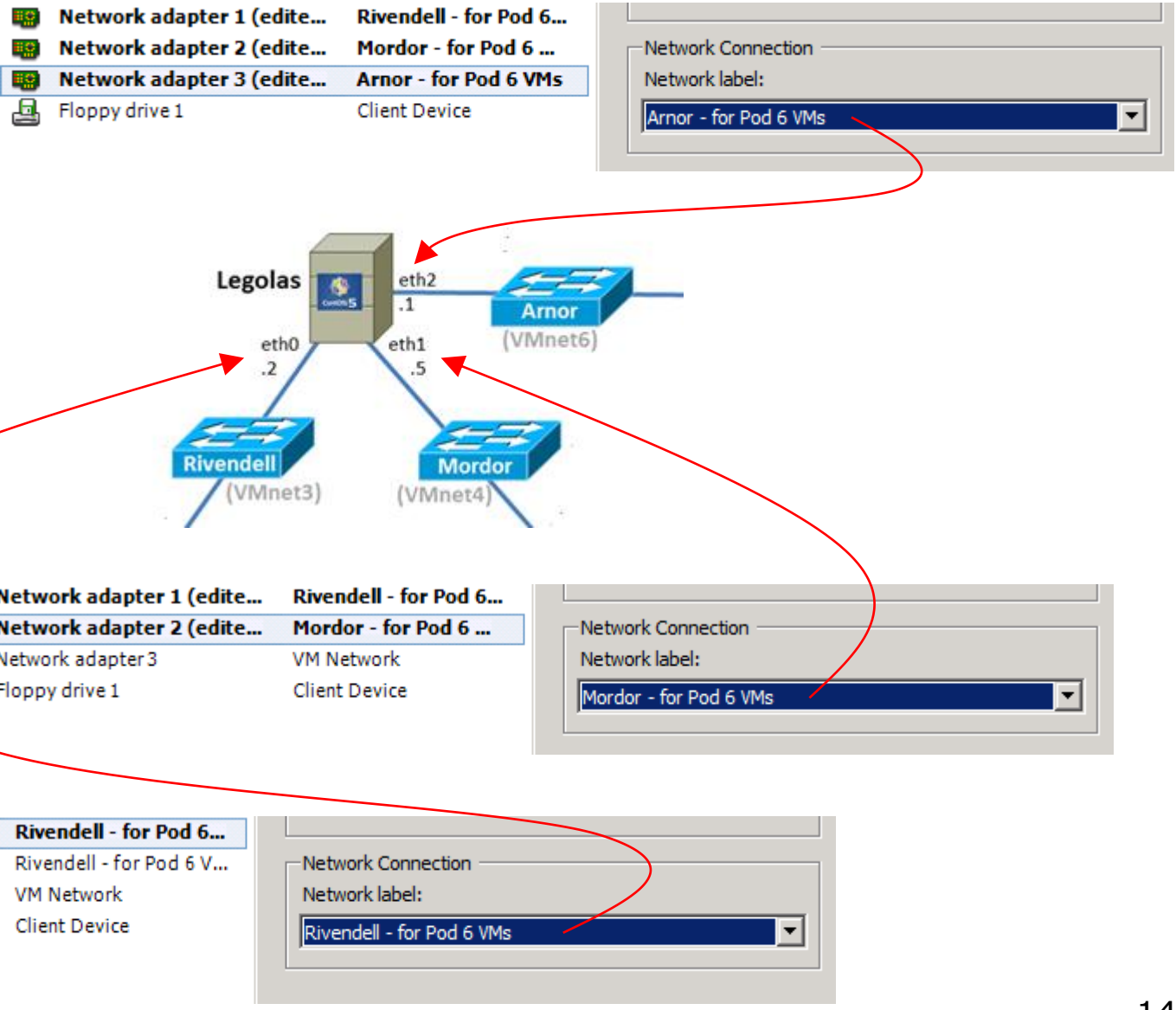
Cabling NICs
(A must for Lab 4)

- Cabling in the **real world** involves connecting the NICs with an Ethernet LAN cable to various hubs or switches.
- Cabling in the VMware **virtual world involves** configuring the Ethernet Adapters to various virtual networks.

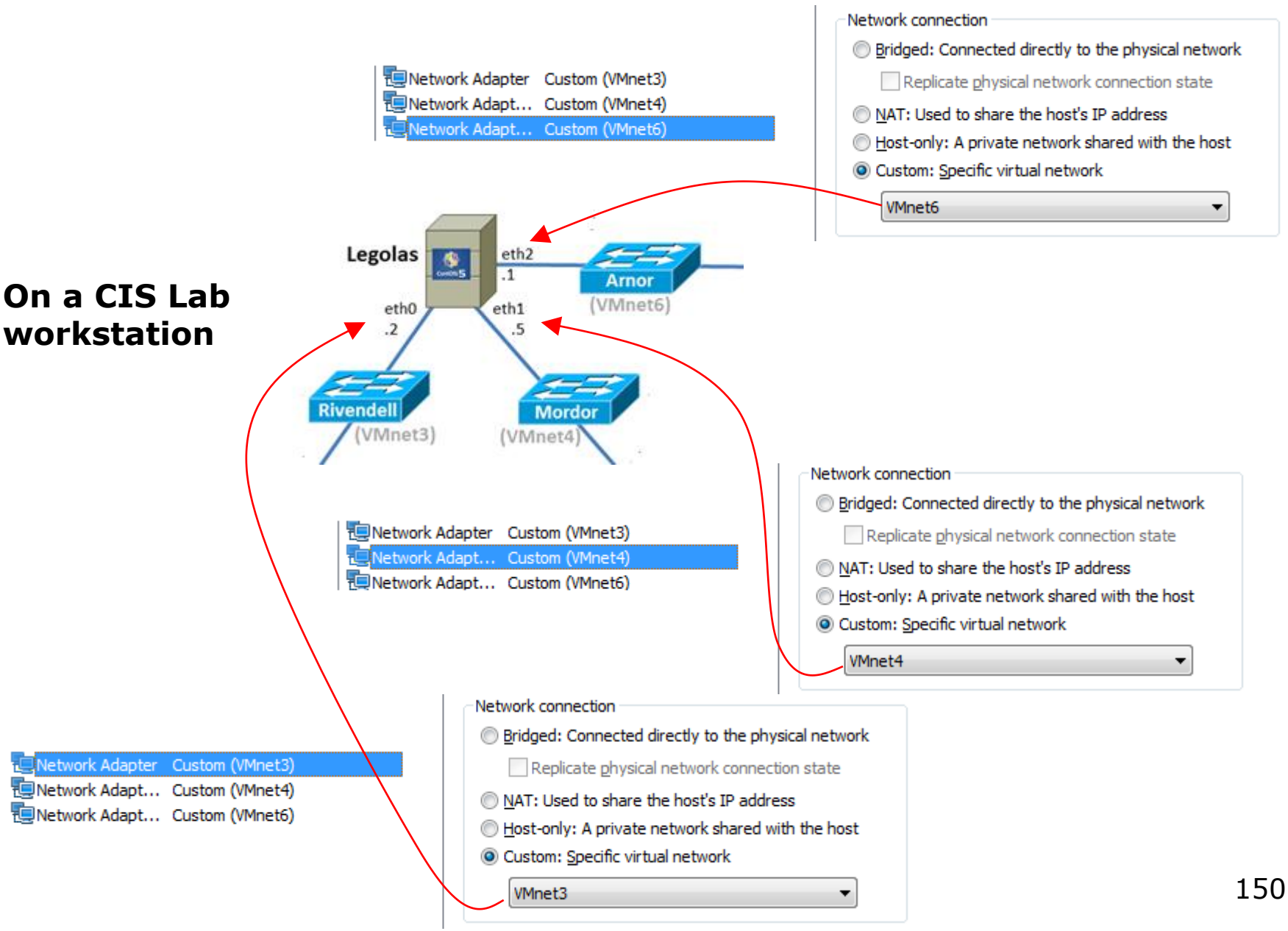
Cabling NICs
(A must for Lab 4)



On VLab Pod 6


Network adapter 1 (edite... Rivendell - for Pod 6...
Network adapter 2 (edite... Mordor - for Pod 6 ...
Network adapter 3 (edite... Arnor - for Pod 6 VMs
Floppy drive 1 Client Device
Network Connection
Network label:
Arnor - for Pod 6 VMs
Legolas
eth2
.1
Arnor
(VMnet6)
eth0
.2
eth1
.5
Rivendell
(VMnet3)
Mordor
(VMnet4)
Network adapter 1 (edite... Rivendell - for Pod 6...
Network adapter 2 (edite... Mordor - for Pod 6 ...
Network adapter 3 VM Network
Floppy drive 1 Client Device
Network Connection
Network label:
Mordor - for Pod 6 VMs
Network adapter 1 (edite... Rivendell - for Pod 6...
Network adapter 2 Rivendell - for Pod 6 V...
Network adapter 3 VM Network
Floppy drive 1 Client Device
Network Connection
Network label:
Rivendell - for Pod 6 VMs

On a CIS Lab workstation



Activity

Cabling Legolas for Lab 4

Live Demo

Telnet Server

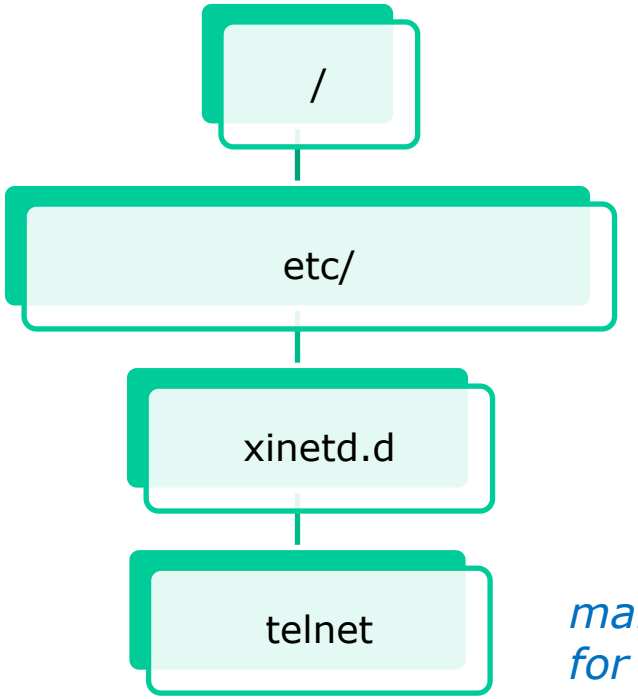
Installing and Configuring Telnet

Step 1 *Install software*

[root@arwen ~]# **yum install telnet-server**

Installing and Configuring Telnet

Step 2 *Customize the configuration files*



main configuration file for telnet

Installing and Configuring Telnet

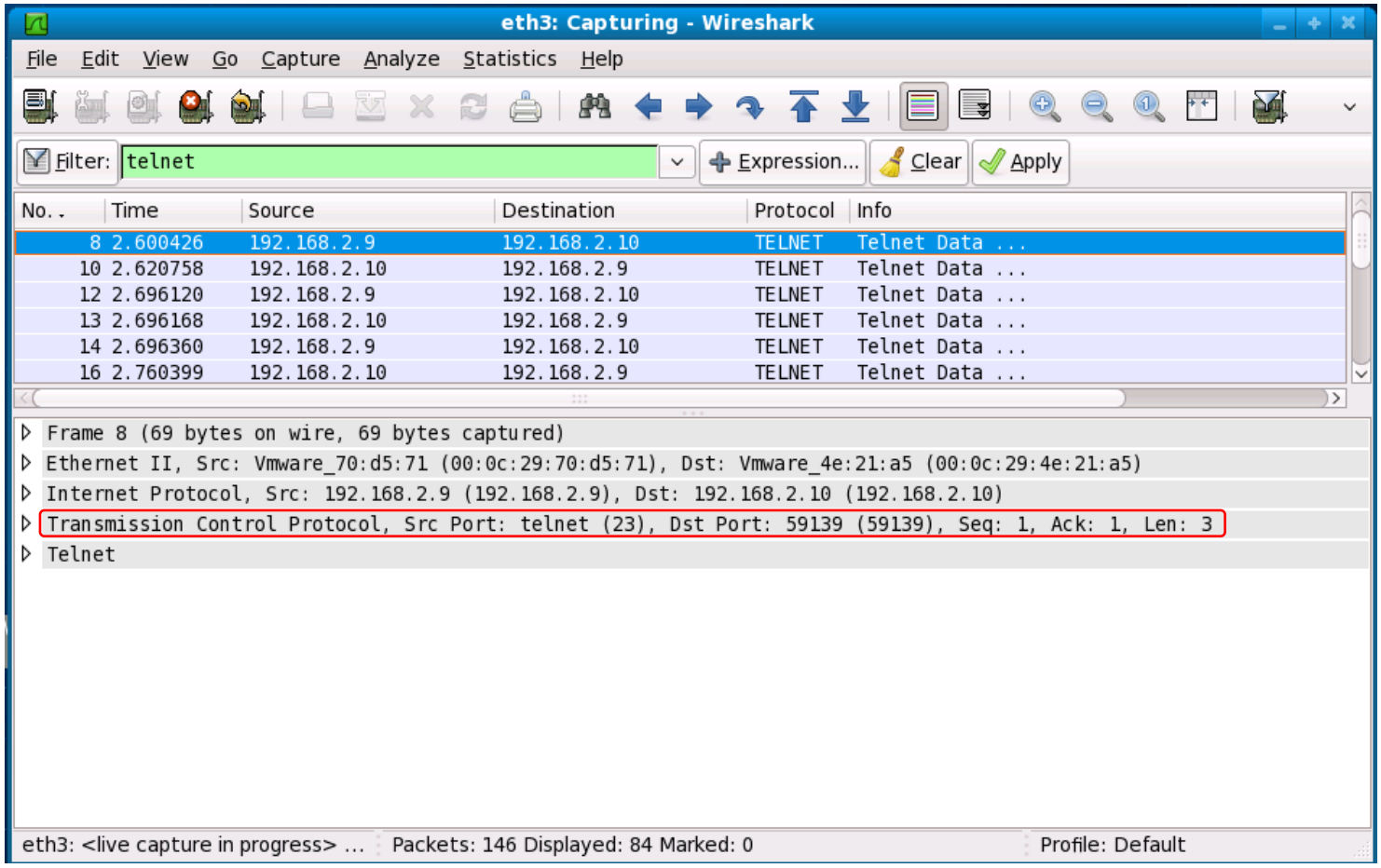
Step 2 *Customize the configuration file*

```
[root@arwen ~]# cat /etc/xinetd.d/telnet
# default: on
# description: The telnet server serves telnet sessions; it uses \
#       unencrypted username/password pairs for authentication.
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        only_from       = 192.168.2.10
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = no
}
[root@arwen ~]#
```

Installing and Configuring Telnet

Step 3 *Modify the firewall*

Firewall must be modified to accept new packets to TDP port 23



Installing and Configuring Telnet

Step 3 *Modify the firewall*

Show the firewall rules with line numbers

iptables -L --line-numbers

Insert rule to allow new incoming telnet connections

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT

Verify

```
[root@celebrian ~]# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere             anywhere
3    ACCEPT     all  --  anywhere             anywhere
4    ACCEPT     udp  --  anywhere             anywhere            udp dpt:router
5    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:telnet
6    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
7    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Installing and Configuring Telnet

Step 4 *Configure SELinux*

More later

Installing and Configuring Telnet

Step 5 *Start the service*

```
[root@arwen ~]# service xinetd restart
Stopping xinetd:                                           [  OK  ]
Starting xinetd:                                           [  OK  ]
[root@arwen ~]#
```

Step 6 *Start the service automatically during system startup*

```
[root@arwen ~]# chkconfig xinetd on
[root@arwen ~]# chkconfig --list xinetd
xinetd          0:off   1:off   2:on    3:on    4:on    5:on    6:off
[root@arwen ~]#
```

Installing and Configuring Telnet

```
[root@arwen ~]# chkconfig --list
```

< snipped >

```
xinetd based services:
        chargen-dgram:   off
        chargen-stream:  off
        daytime-dgram:   off
        daytime-stream:  off
        discard-dgram:   off
        discard-stream:  off
        echo-dgram:      off
        echo-stream:     off
        tcpmux-server:   off
        telnet:          on
        time-dgram:      off
        time-stream:     off
```

xinetd is a super daemon which acts as an umbrella for many other services

Installing and Configuring Telnet

Step 7 *Monitor and verify service is running*

```
[root@celebrian ~]# netstat -tlp
[root@celebrian ~]# netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 127.0.0.1:2601              0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:2602              0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN
tcp        0      0 :::22                       :::*                        LISTEN
tcp        0      0 :::23                       :::*                        LISTEN
tcp        0      0 ::1:25                      :::*                        LISTEN
[root@celebrian ~]#
```

telnet daemons listens on TCP port 23

Installing and Configuring Telnet

Step 8 *Troubleshoot*

More later

Telnet

Step 9 *Monitor log files*

```
Nov 15 09:13:19 celebrian xinetd[6922]: failed to parse
192.168.2.* [file=/etc/xinetd.d/telnet] [line=10]
Nov 15 09:13:19 celebrian xinetd[6922]: xinetd Version 2.3.14
started with libwrap loadavg labeled-networking options
compiled in.
Nov 15 09:13:19 celebrian xinetd[6922]: Started working: 1
available service
Nov 15 12:29:49 celebrian xinetd[6922]: Exiting...
Nov 15 12:29:49 celebrian xinetd[6998]: failed to parse
192.168.2. [file=/etc/xinetd.d/telnet] [line=10]
Nov 15 12:29:49 celebrian xinetd[6998]: xinetd Version 2.3.14
started with libwrap loadavg labeled-networking options
compiled in.
Nov 15 12:29:49 celebrian xinetd[6998]: Started working: 1
available service
[root@celebrian ~]#
```

Quagga

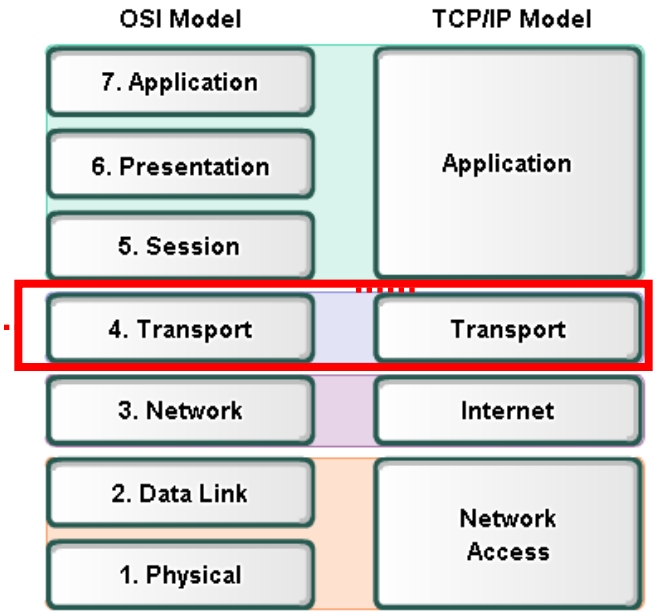
Step 10 *Configure additional security*

More later

Transport Layer Overview

Transport Layer

TCP UDP



- The Layer 4 data stream is a:
  - logical connection between the endpoints of a network,
  - provides transport services from a host to a destination.
- **End-to-end service**.
- The transport layer also provides two protocols
  - **TCP** - Transmission Control Protocol
  - **UDP** - User Datagram Protocol
- PDU: **Segment** *(TCP)*

Lingo: Ethernet frames, IP packets, TCP segments, and UDP datagrams

Transport Layer

The Protocols

There are two primary protocols operating at the Transport layer:

User Datagram Protocol (UDP)
- Connectionless *(snmp traps are "fire and forget")*
- Stateless
- *Unreliable*
- The UDP packet is called a ***packet***

Transmission Control Protocol (TCP)
- Connection-oriented
- Statefull *(like new or established states in firewalls)*
- *Reliable* The TCP packet is called a ***segment***

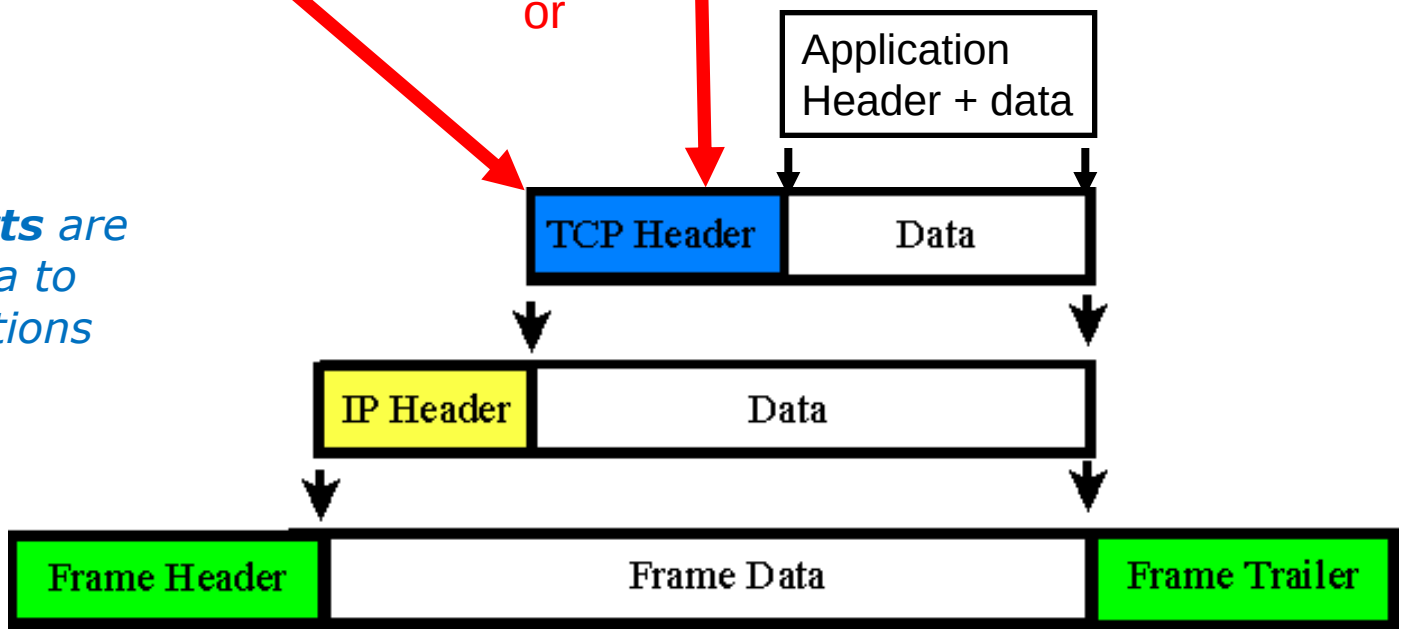
TCP Header

Source Port (16 bits)								Destination Port (16 bits)
Sequence Number (32 bits)								
Acknowledgement Number (32 bits)								
Data Offset (4 bits)	Reserved (6 bits)	URG	ACK	PSH	RST	SYN	FIN	Window (16 bits)
Checksum (16 bits)								Urgent Pointer (16 bits)
Options and Padding								

UDP Header

Source Port (16 bits)	Destination Port (16 bits)
Length (16 bits)	Checksum (16 bits)
Data....	

or



*The source and destination **ports** are used to get data to specific applications*

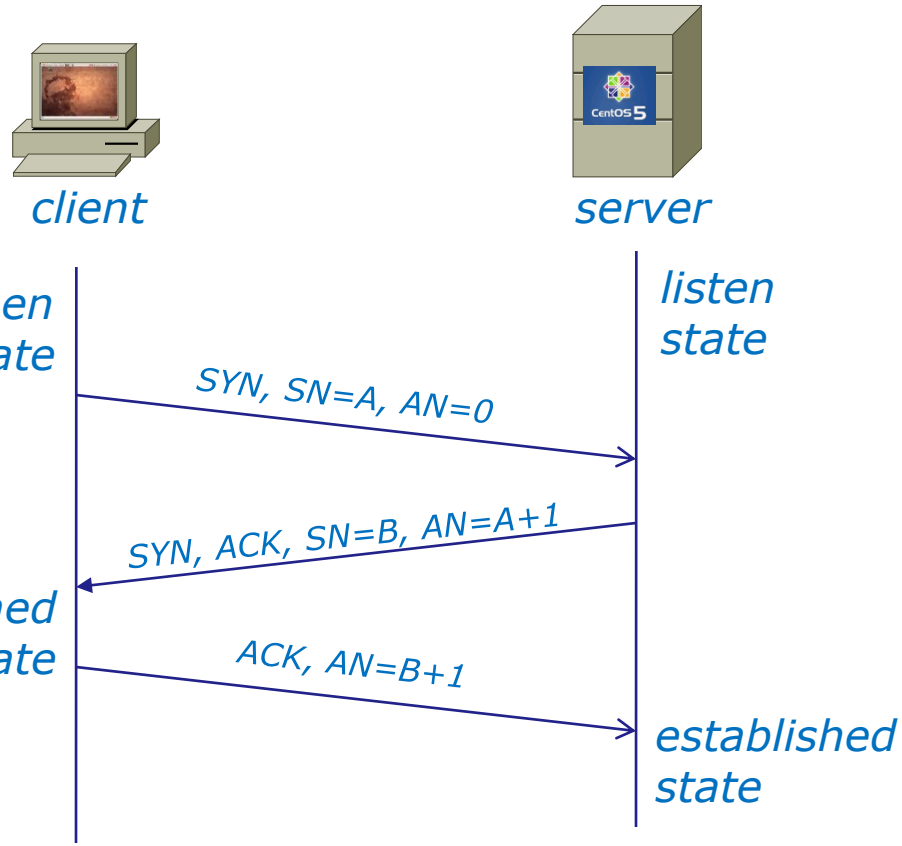
Transport Layer

The Transmission Control Protocol

Initial Connection

Three-Way Handshake

1. SYN
2. SYN-ACK
3. ACK



AN=Acknowledgment Number
SN=Sequence Number
ACK=ACK flag set

Transport Layer

Sockets

Sockets are communication endpoints which define a network connection between two computers (RFC 793).

- Source IP address
- Source port number
- Destination IP address
- Destination port number

The socket is associated to a port number so that the TCP layer can identify the application to send data to.

Application programs can read and write to a socket just like they do with files.

Transport Layer

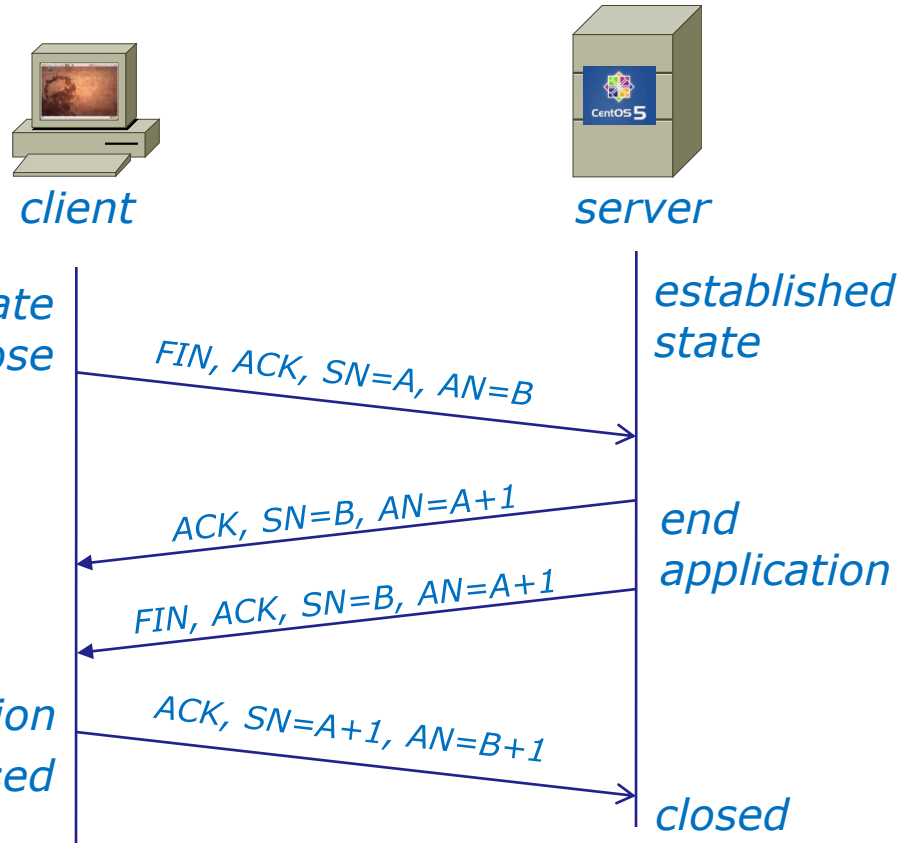
The Transmission Control Protocol

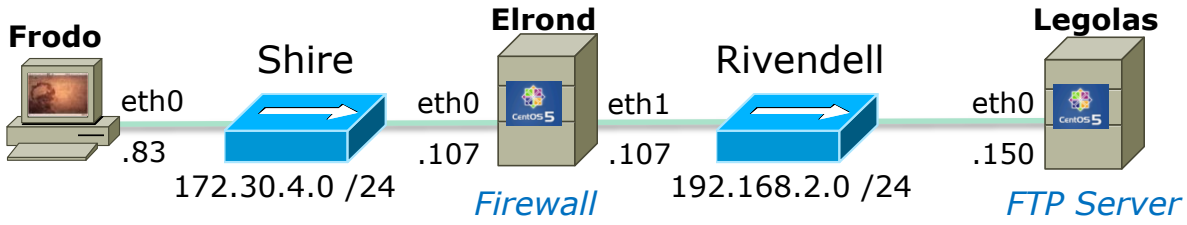
Closing a Connection

Four-Way Handshake

1. FIN, ACK
2. ACK
3. FIN, ACK
4. ACK

AN=Acknowledgment Number
SN=Sequence Number
ACK=ACK flag set
FIN=FIN flag set





Active Mode *is when server initiates new connection for data transfer*

Socket for data transfer

Client	Server
172.30.4.83	192.168.2.150
42571	20

```
ftp> get legolas
local: legolas remote: legolas
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for legolas (18 bytes).
226 File send OK.
18 bytes received in 0.04 secs (0.5 kB/s)
```

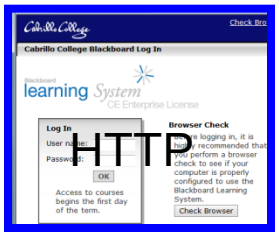
SIP	SP	DIP	DP	Protocol	Info
172.30.4.83	42855	192.168.2.150	21	FTP	Request: PORT 172,30,4,83,166,75
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 200 PORT command successful. Consider using PAS
172.30.4.83	42855	192.168.2.150	21	FTP	Request: RETR legolas
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [SYN] Seq=0 Wi
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [SYN, ACK] Seq
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=1 Ack
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 150 Opening BINARY mode data connection for leg
192.168.2.150	20	172.30.4.83	42571	FTP-DATA	FTP Data: 18 bytes
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [FIN, ACK] Seq=19 Ack=1 Win=5888 Len=0
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [ACK] Se
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [FIN, AC
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=20 Ack=2 Win=5888 Len=0
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 226 File send OK.
172.30.4.83	42855	192.168.2.150	21	TCP	42855 > ftp [ACK] Seq=82 Ack=263 Win=5856 Len=0

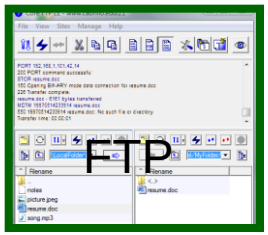
Retrieve legolas file

3 way handshake initiated by server

File transfer

4 way handshake to close connection





7. Application
6. Presentation
5. Session
4. Transport
3. Network
2. Data Link
1. Physical

TCP
TCP
TCP
TCP
TCP
UDP
TCP
UDP

7. Application
6. Presentation
5. Session
4. Transport
3. Network
2. Data Link
1. Physical

Cabrillo Web Server

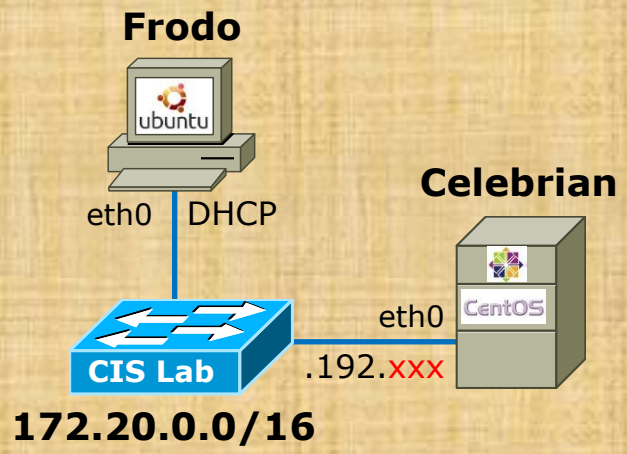
7. Application
6. Presentation
5. Session
4. Transport
3. Network
2. Data Link
1. Physical

ISP's Email and FTP Server

- A **single client** may have <u>multiple transport connections</u> with multiple servers.
- Notice that **TCP is a connection-oriented** service (two-way arrow) between the hosts, whereas **UDP is a connectionless** service (one-way arrow) . (later)

Activity

Observing a TCP handshake



On Frodo:
apt-get install wireshark
filter: tcp.port == 23 and ip.addr == 172.20.192.xxx

ssh into Frodo from Opus and use that session to telnet into Celebrian which is running the a telnet service

Stop capture and observe start and finish handshakes

Service Ports

Transport Layer

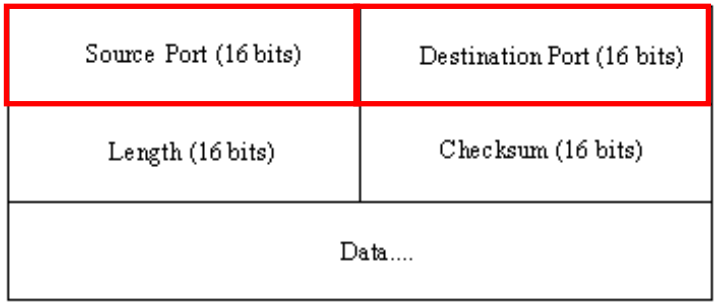
Service Ports

Defined and managed by the Internet Assigned Numbers Authority (IANA)and The Internet Corporation for Assigned Names and Numbers (ICANN)

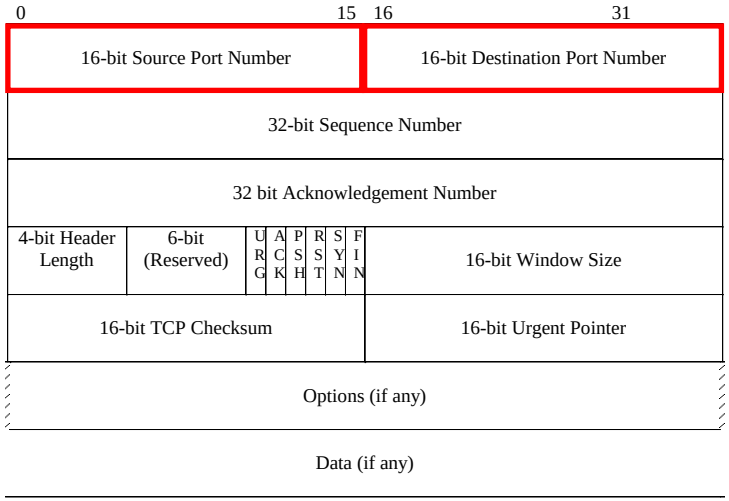
- Well known ports (0-1023)
- Registered ports (1024 through 49151)
- Dynamic or Private ports (49152 through 65535)

Well known ports (AKA privileged ports) are intended to only be used by system or root processes or programs executed by privileged users.

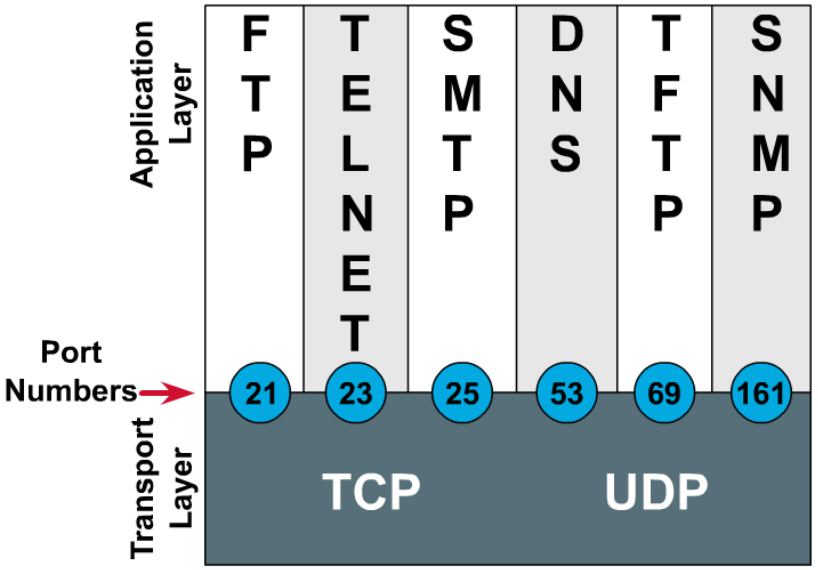
UDP Header



TCP Header



Port Numbers

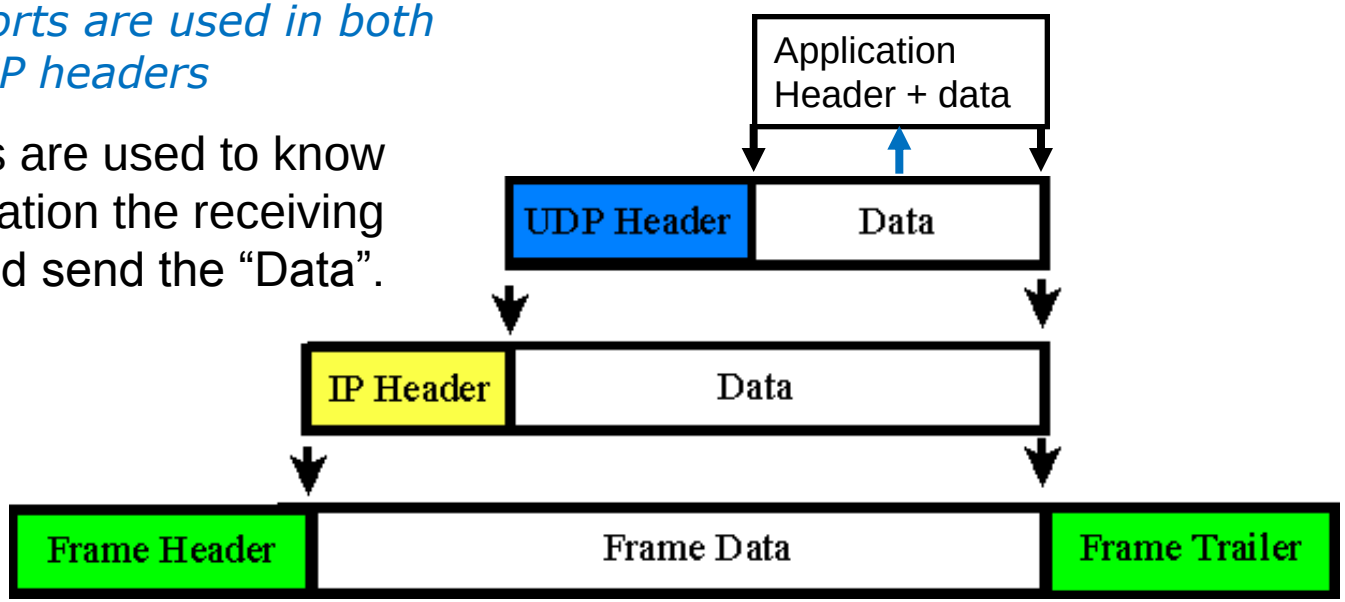


E.g. HTTP is Port 80

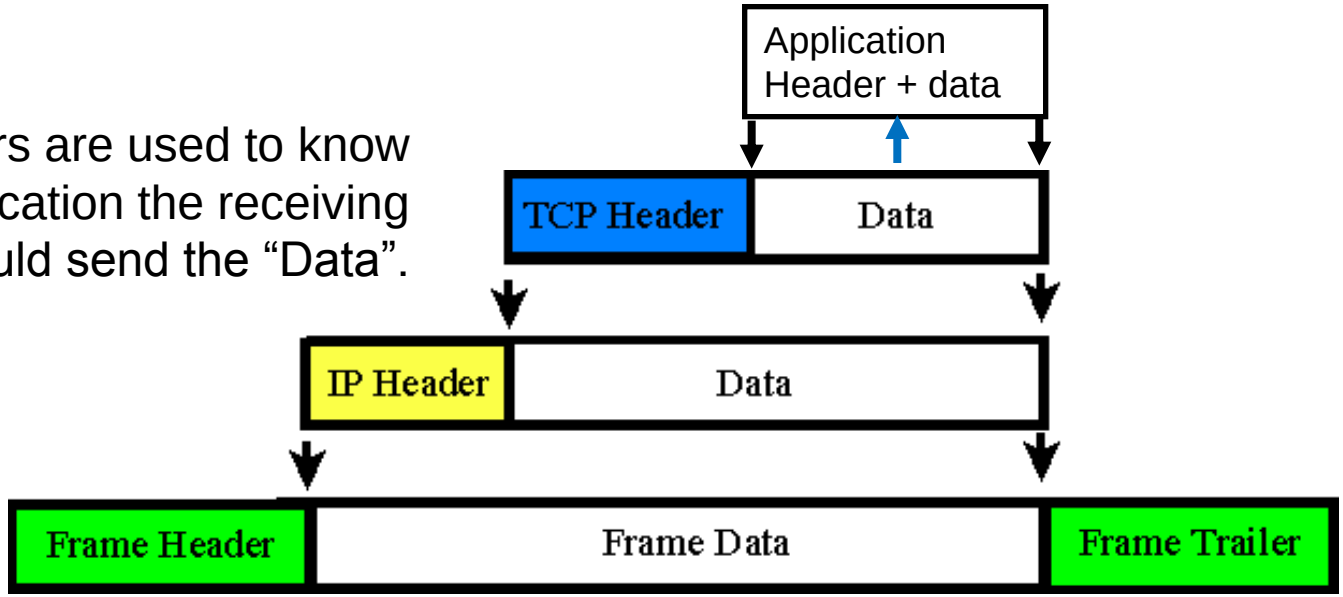
Both TCP and UDP use ports (or sockets) numbers to pass information to the upper layers.

Note that ports are used in both UDP and TCP headers

Port numbers are used to know which application the receiving host should send the "Data".



Port numbers are used to know which application the receiving host should send the "Data".



Service Ports

Well-known and registered ports listed in /etc/services

```
[root@elrond ~]# cat /etc/services | more
# /etc/services:
# $Id: services,v 1.42 2006/02/23 13:09:23 pknirsch Exp $
#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994).  Not all ports
# are included, only the more common ones.
#
# The latest IANA port assignments can be gotten from
#       http://www.iana.org/assignments/port-numbers
# The Well Known Ports are those from 0 through 1023.
# The Registered Ports are those from 1024 through 49151
# The Dynamic and/or Private Ports are those from 49152 through 65535
#
# Each line describes one service, and is of the form:
#
# service-name  port/protocol  [aliases ...]   [# comment]

tcpmux          1/tcp                           # TCP port service multiplexer
tcpmux          1/udp                           # TCP port service multiplexer
rje             5/tcp                           # Remote Job Entry
rje             5/udp                           # Remote Job Entry
```

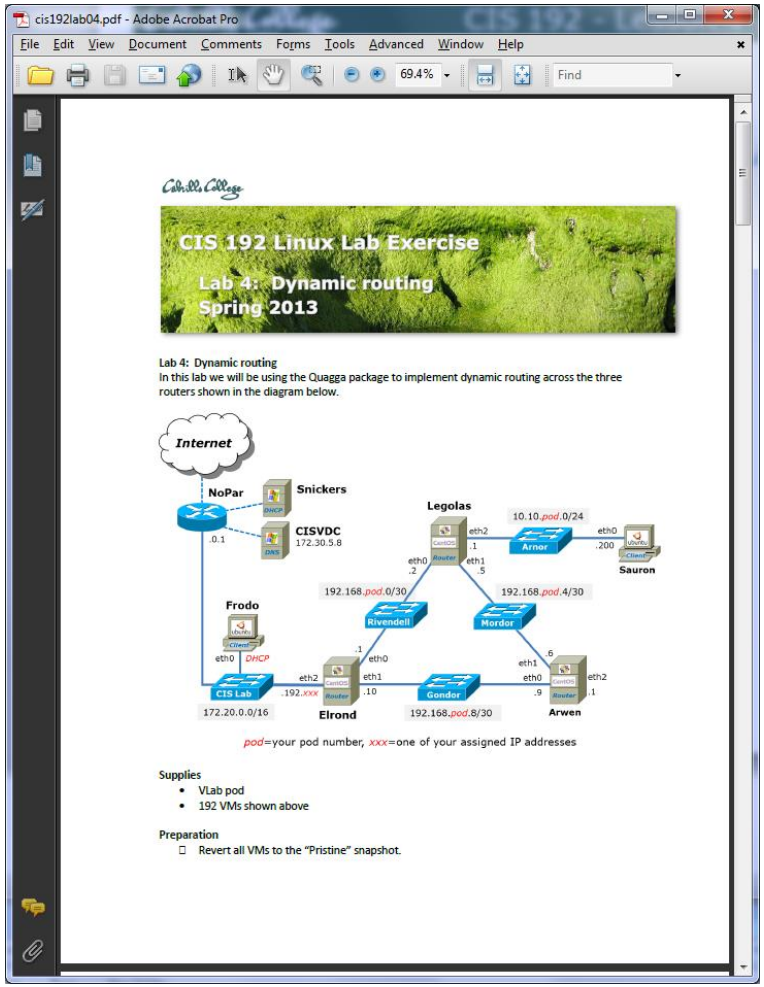
< snipped >

Service Ports

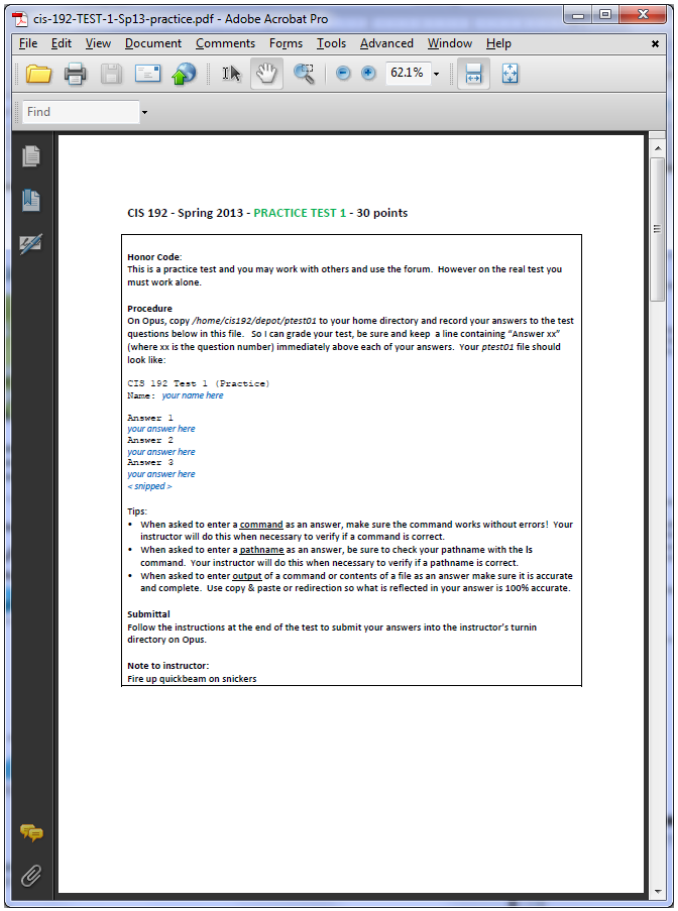
some favorites from /etc/services file

```
< snipped >
# 21 is registered to ftp, but also used by fsp
ftp             21/tcp
ftp             21/udp          fsp fspd
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
telnet          23/udp
# 24 - private mail system
lmtp            24/tcp                          # LMTP Mail Delivery
lmtp            24/udp                          # LMTP Mail Delivery
smtp            25/tcp          mail
smtp            25/udp          mail
< snipped >
domain          53/tcp                          # name-domain server
domain          53/udp
whois++         63/tcp
whois++         63/udp
bootps          67/tcp                          # BOOTP server
bootps          67/udp
bootpc          68/tcp          dhcpc           # BOOTP client
bootpc          68/udp          dhcpc
tftp            69/tcp
tftp            69/udp
finger          79/tcp
finger          79/udp
http            80/tcp          www www-http    # WorldWideWeb HTTP
http            80/udp          www www-http    # HyperText Transfer Protocol
kerberos        88/tcp          kerberos5 krb5  # Kerberos v5
< snipped >
```

Not a Wrap Yet



Start early on Lab 4 …
*it's a **beefy** one!*



Practice test for next week

New commands, tools and services:

iptables -L --line-numbers
service ripd restart
service xinetd restart
service zebra restart
service ripd restart
telnet localhost 2601
telnet localhost 2602
vtysh
yum install quagga

New Files and Directories:

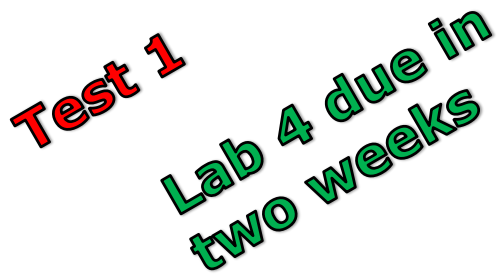
/etc/quagga/ripd.conf
/etc/quagga/zebra.conf
/etc/services
/etc/sysconfig/iptables
/etc/xinetd.d/telnet

Next Class

Lab 4 not due for two weeks

No Quiz for next class

Test for next class on lessons 1 through 4



- Open book, open notes, open VMs, during last hour of class
- Practice test available
- Doing Lab 4 early would be good practice for test
- Test will be given during last part of class
- If you would like extra time you can take it home and turn it in by 11:59PM

Students may work together and use the forum to work out the answers on the practice test.

*The actual test will be **almost identical** to the practice test.*

For the actual test, students must work individually and neither ask nor give assistance to others.

Backup

*Some **Distance Vector** routing protocols*
(The Cost) (The Direction)

Routing Information Protocol (RIP) was originally specified in RFC 1058.

- It is a **distance vector** routing protocol.
- **Hop count** is used as the metric for path selection.
- If the hop count is **greater than 15, the packet is discarded**.
- Routing updates are broadcast **every 30 seconds**, by default.

Interior Gateway Routing Protocol (IGRP) is a proprietary protocol developed by Cisco.

- It is a **distance vector** routing protocol.
- **Bandwidth, load, delay and reliability** are used to create a composite metric.
- Routing updates are broadcast **every 90 seconds**, by default.

EIGRP is a Cisco proprietary enhanced distance vector routing protocol.

- It is an **enhanced distance vector routing protocol**.
- Uses **unequal-cost and equal-cost** load balancing.
- Uses a combination of distance vector and link-state features.
- Uses **Diffused Update Algorithm (DUAL)** to calculate the shortest path.

Link-state routing protocols - each node knows the entire network topology and can compute the shortest paths

Open Shortest Path First (OSPF) is a nonproprietary link-state routing protocol.

- It is a **link-state** routing protocol.
- **Open standard** routing protocol described in RFC 2328.
- Uses the **SPF algorithm** to calculate the lowest cost to a destination.
- **Routing updates are flooded** as topology changes occur.

Intermediate System to Intermediate System (IS-IS)

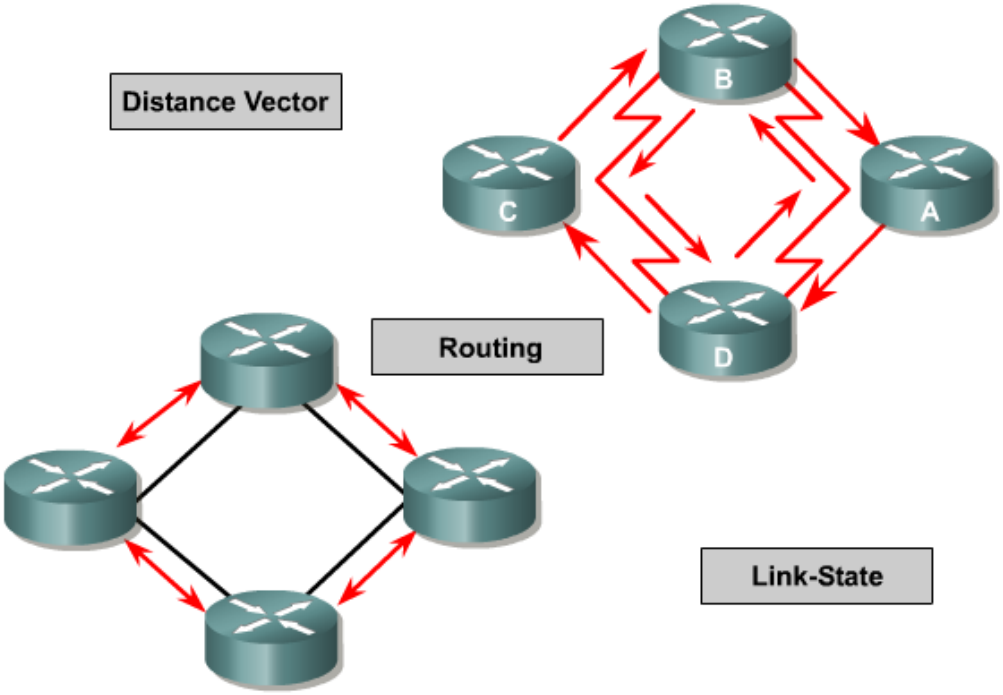
- IS-IS is an Open System Interconnection (OSI) routing protocol originally specified by International Organization for Standardization (ISO) 10589.
- It is a **link-state** routing protocol.

Exterior routing protocols - used between autonomous systems

Border Gateway Protocol (BGP) is an exterior routing protocol.

- It is a **distance vector** (or path vector) exterior routing protocol
- Used between **ISPs or ISPs and clients**.
- Used to **route Internet traffic** between autonomous systems.

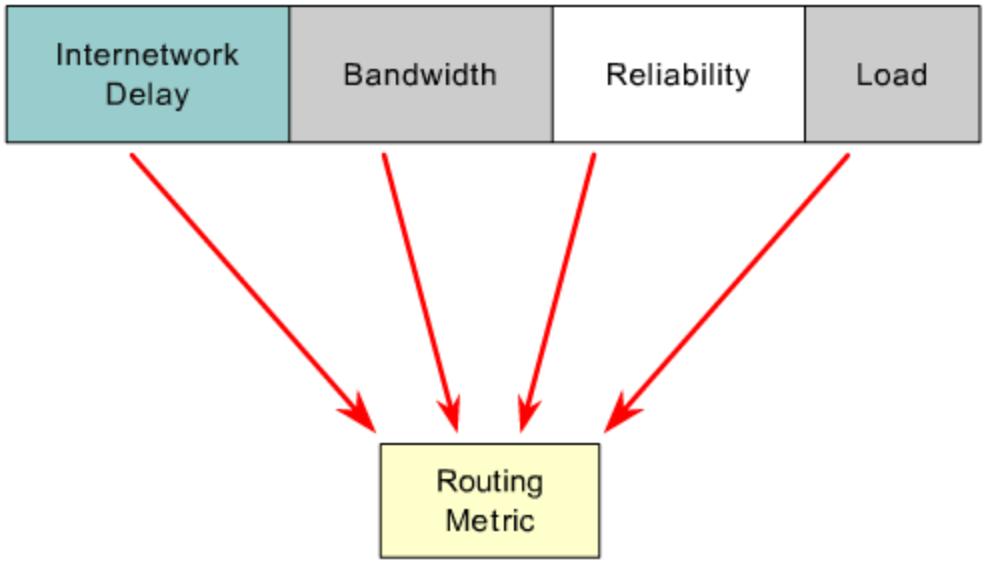
Types of Routing Protocols



- Distance Vector: RIP, IGRP, EIGRP
- Link State: OSPF, IS-IS
- Path Vector: BGP
- Note: IGRP and EIGRP are Cisco Proprietary

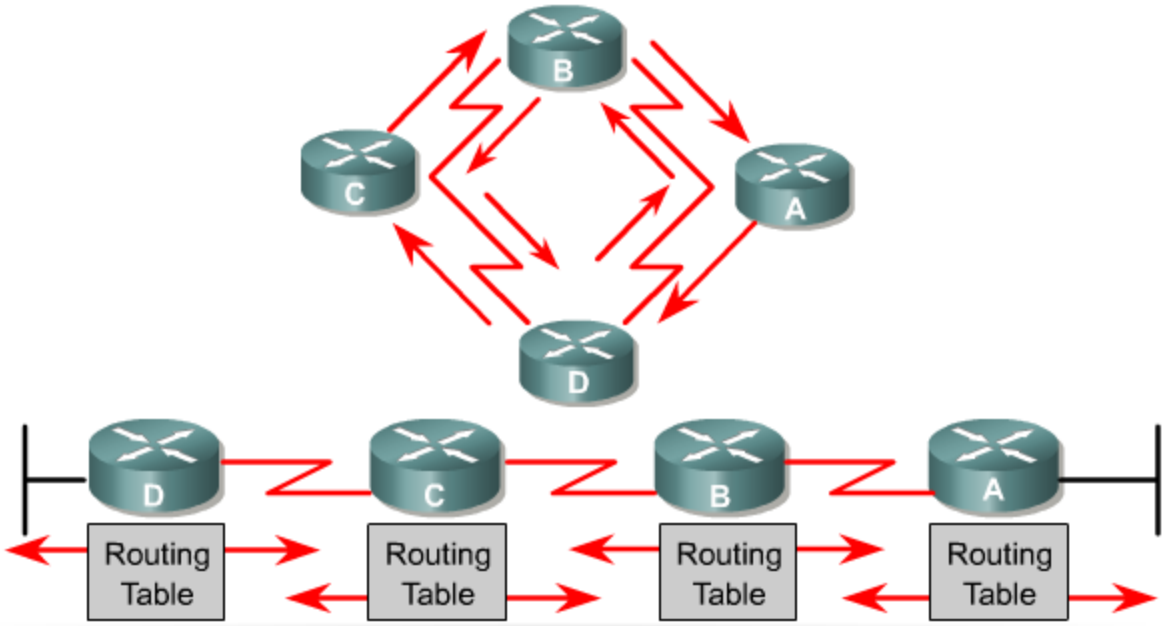
Path vector protocols (like BGP) are a class of distance vector protocols and not a link-state protocol

Routing Protocol Metrics (costs)



- RIP - Hop Count
- IGRP and EIGRP - Bandwidth, Delay, Reliability, Load
- Cisco's OSPF - Bandwidth
- IS-IS - Cost
- BGP - Number of AS or policy

Distance Vector Routing Protocols



Pass periodic copies of a routing table to neighbor routers and accumulate distance vectors.

Router B receives information from Router A.

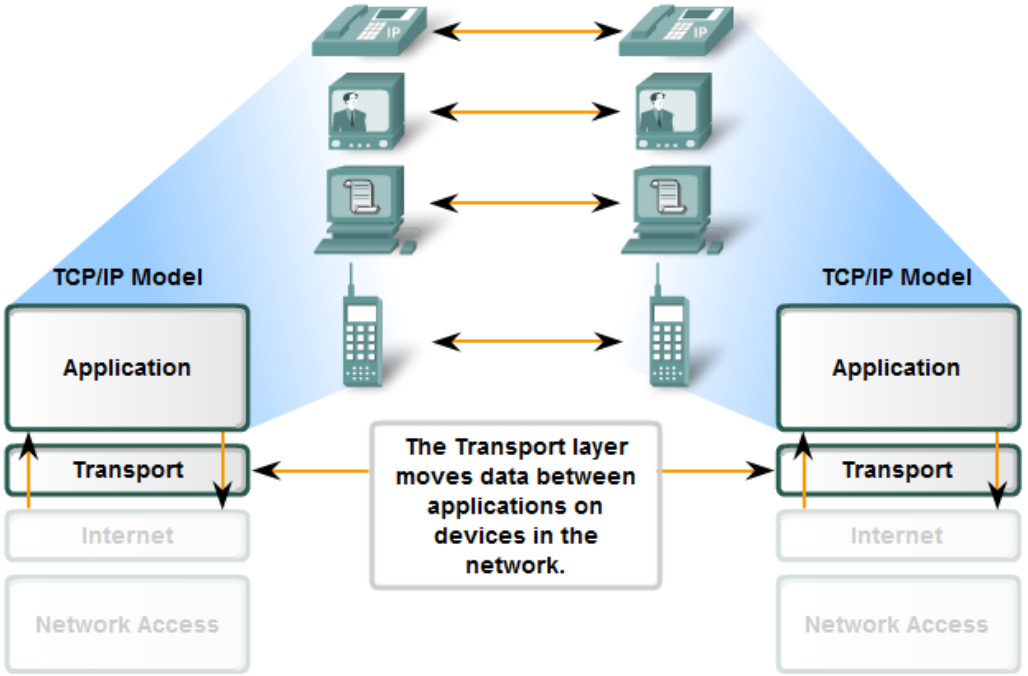
Router B adds a distance vector number (such as a number of hops), which increases the distance vector.

Then Router B passes this new routing table to its other neighbor, Router C.

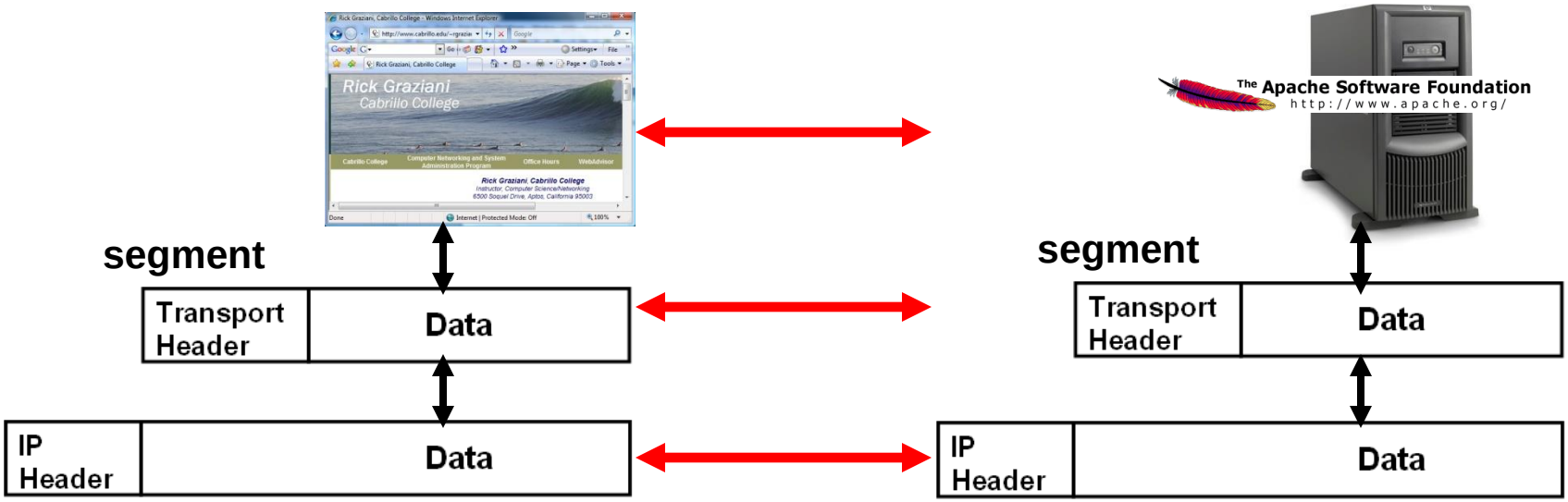
This same step-by-step process occurs in all directions between neighbor routers.

- "Routing by rumor"
- Each router receives a routing table from its directly connected neighbor routers.

Transport Layer



- Primary responsibilities:
  - Tracking the individual communication between applications
  - Segmenting data
  - Managing each segment
  - Reassembling the segments
  - Identifying the different applications

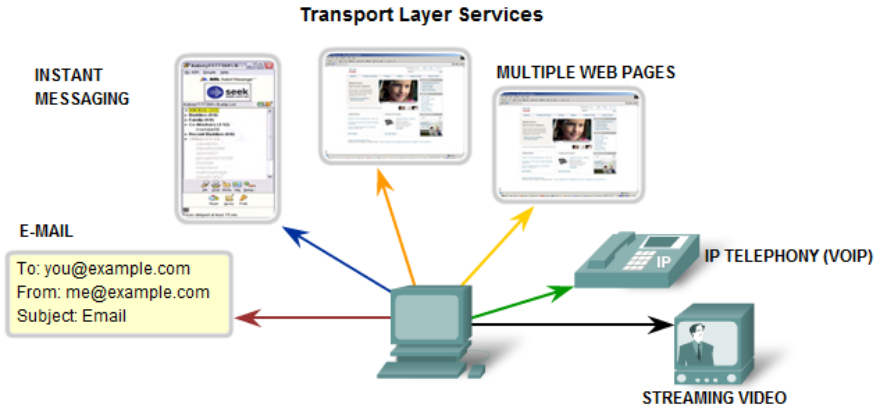


Transport Layer

- Protocols:
  - **TCP**
  - **UDP**
- **IP** is a best-effort delivery service
  - No guarantees
  - Best-effort service
  - "Unreliable service"
- TCP/UDP is responsible for extending IP's delivery service between two end systems.
  - Known as transport layer **multiplexing** and **demultiplexing**.

Breaking up into little pieces and reassembling at the end

TCP vs. UDP



TCP provides:

- Reliable delivery
- Error checking
- Flow control
- Congestion control
- Ordered delivery
- (Connection establishment)
- Applications:
  - HTTP
  - FTP
  - Telnet
  - MSN messenger

UDP provides:

- Unreliable delivery
- No error checking
- No flow control
- No congestion control
- No ordered delivery
- (No connection establishment)
- Applications
  - DNS (usually)
  - SMTP
  - DHCP
  - RTP (Real-Time Protocol)
  - VoIP

and SNMP "fire and forget" traps, RIP updates

Transmission Control Protocol

Transport Layer

The Transmission Control Protocol

Initial Connection

Three-Way Handshake

1. SYN
2. SYN-ACK
3. ACK

Continuing Communications

- The Sliding Window
- Flow Control (cumulative acknowledgment)
- SACK
- The RST Flag

Closing a Connection

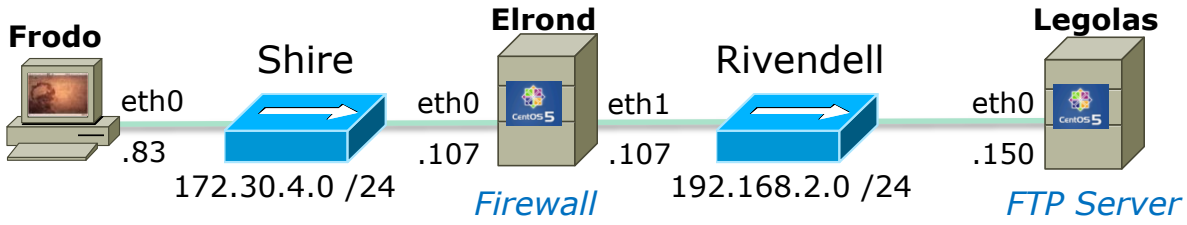
Four-Way Handshake

1. FIN, ACK
2. ACK
3. FIN, ACK
4. ACK

More on this later…

We want to be able to identify the start, flow and end of TCP connections as we start exploring network services.

Some quick preview examples for now



Socket for commands

Client	Server
172.30.4.83	192.168.2.150
42855	21

Socket for data transfer

Client	Server
172.30.4.83	192.168.2.150
42571	20

***Active Mode** is when server initiates new connection for data transfer*

```
ftp> get legolas
local: legolas remote: legolas
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for legolas (18 bytes).
226 File send OK.
18 bytes received in 0.04 secs (0.5 kB/s)
```

SIP	SP	DIP	DP	Protocol	Info
172.30.4.83	42855	192.168.2.150	21	FTP	Request: PORT 172,30,4,83,166,75
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 200 PORT command successful. Consider using PAS
172.30.4.83	42855	192.168.2.150	21	FTP	Request: RETR legolas
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [SYN] Seq=0 Wi
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [SYN, ACK] Seq
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=1 Ack
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 150 Opening BINARY mode data connection for leg
192.168.2.150	20	172.30.4.83	42571	FTP-DATA	FTP Data: 18 bytes
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [FIN, ACK] Seq=19 Ack=1 Win=5888 Len=0
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [ACK] Se
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [FIN, AC
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=20 Ack=2 Win=5888 Len=0
192.168.2.150	21	172.30.4.83	42855	FTP	Response: 226 File send OK.
172.30.4.83	42855	192.168.2.150	21	TCP	42855 > ftp [ACK] Seq=82 Ack=263 Win=5856 Len=0

PORT command to listen on 166, 75 = A64B = 42571

Retrieve legolas file

3 way handshake initiated by server

File transfer

4 way handshake to close connection

Tunable Kernel Parameters

Transport Layer

TCP Tunable Kernel Parameters

tcp_fin_timeout
tcp_keepalive_time
tcp_sack
tcp_timestamps
tcp_window_scaling
tcp_retries1
tcp_retries2
tcp_syn_retries

Security Issues

Transport Layer

Security Issues

Resource: *www.securityfocus.org*

- SYN Flooding
- Falsifying TCP Communications
- Hijacking connections

Quagga - Some RIP troubleshooting

ripd service

```
legolas(ripd)# show ip rip status
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 14 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected static
  Default version control: send version 2, receive any version
    Interface        Send  Recv   Key-chain
    eth0             2     1 2
    eth1             2     1 2
  Routing for Networks:
    eth0
    eth1
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    192.168.2.1               0         0       120   00:00:14
    192.168.2.6             481         0       120   00:00:11
  Distance: (default is 120)
legolas(ripd)#
```

If your routing table is not getting any RIP routes then check the rip status. Any BadPackets indicate the incoming RIP updates are being ignored!

Quagga - Some RIP troubleshooting

```
[root@legolas ~]# cat /etc/quagga/ripd.conf
!
! Zebra configuration saved from vty
!   2009/02/25 16:36:10
!
hostname legolas(ripd)
password <password>
log file /var/log/quagga/ripd.log
!
debug rip events
debug rip zebra
!
interface eth0
 no ip rip authentication mode text
 no ip rip authentication mode md5
!
interface eth1
 no ip rip authentication mode text
 no ip rip authentication mode md5
!
router rip
 redistribute connected
 redistribute static
 network eth0
 network eth1
!
[root@legolas ~]# service ripd restart
Shutting down ripd:                                        [  OK  ]
Starting ripd:                                             [  OK  ]
```

The BadPackets were caused by unauthenticated routing updates

The fix: If you are not going to authenticate incoming updates then add this to the configuration file or the routing tables will never update

Restart service if changes made to configuration file

Quagga - Some RIP troubleshooting

After changing the ripd configuration file, restart the service so the changes will take effect

```
[root@legolas ~]# service ripd restart
Shutting down ripd:                                        [  OK  ]
Starting ripd:                                             [  OK  ]
```

And login again to the shell to check the RIP status

```
[root@legolas ~]# telnet localhost 2602
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^]'.

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
legolas(ripd)> en
legolas(ripd)#
```

Quagga - Some RIP troubleshooting

```
legolas(ripd)# sh ip rip status
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 29 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected static
  Default version control: send version 2, receive any version
    Interface        Send  Recv   Key-chain
    eth0             2     1 2
    eth1             2     1 2
  Routing for Networks:
    eth0
    eth1
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    192.168.2.1               0         0       120   00:00:03
    192.168.2.6               0         0       120   00:00:02
  Distance: (default is 120)
legolas(ripd)#
```

Now RIP routes will be inserted into the routing table

Quagga and the Firewall

Step 3 *Modify the firewall*

Firewall ports used for implementing RIPv2 with Quagga

UDP 520 *RIP advertisements*

Quagga and the Firewall

For Lab 4:

- The routers need UDP port 520 open to allow incoming RIP packets
- The routers need to allow packets to pass through (via packet forwarding)

For Lab X1:

- For the Telnet Server, TCP port 23 needs to be open for incoming Telnet connections

Quagga and the Firewall

Default firewall (in memory)

```
[root@celebrian ~]# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    ACCEPT     icmp --  anywhere             anywhere
3    ACCEPT     all  --  anywhere             anywhere
4    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
5    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
[root@celebrian ~]#
```

- *There is no rule on the INPUT chain to accept incoming RIP packets (UDP port 520) so they will be rejected.*
- *All packets going through the FORWARD chain get rejected.*

Quagga and the Firewall

Default firewall (in configuration file)

```
[root@celebrian ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@celebrian ~]#
```

- *There is no rule on the INPUT chain to accept incoming RIP packets (UDP port 520) so they will be rejected.*
- *All packets going through the FORWARD chain get rejected.*

Quagga and the Firewall

```
[root@celebrian ~]# iptables -D FORWARD 1
```

Delete the first rule on the FORWARD chain

```
[root@celebrian ~]# iptables -I INPUT 4 -p udp -m udp --dport 520 -j ACCEPT
```

Insert a rule above rule 4 on the INPUT chain to accept incoming packets to UDP port 520

```
[root@celebrian ~]# iptables -I INPUT n -p tcp -m state --state NEW -m tcp --
dport 23 -j ACCEPT
```

(all on one line)

Insert a rule above rule 5 on the INPUT chain to accept incoming packets to TCP port 23

```
[root@celebrian ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]
```

Save the rules in memory to the configuration file

Modifying the Firewall
(Centos)

Modified firewall (in memory)

```
[root@celebrian ~]# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    ACCEPT     udp  --  anywhere             anywhere            udp dpt:router
3    ACCEPT     icmp --  anywhere             anywhere
4    ACCEPT     all  --  anywhere             anywhere
5    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:telnet
6    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
7    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
[root@celebrian ~]#
```

RIP and Telnet ports open

No filtering now on forwarded packets

Quagga and the Firewall

Modified firewall (in configuration file)

```
[root@celebrian ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.4.7 on Tue Nov 15 00:41:40 2011
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [11:1740]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p udp -m udp --dport 520 -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Tue Nov 15 00:41:40 2011
[root@celebrian ~]#
```

No filtering now on forwarded packets

RIP and Telnet ports open

Quagga and the Firewall

Modified firewall

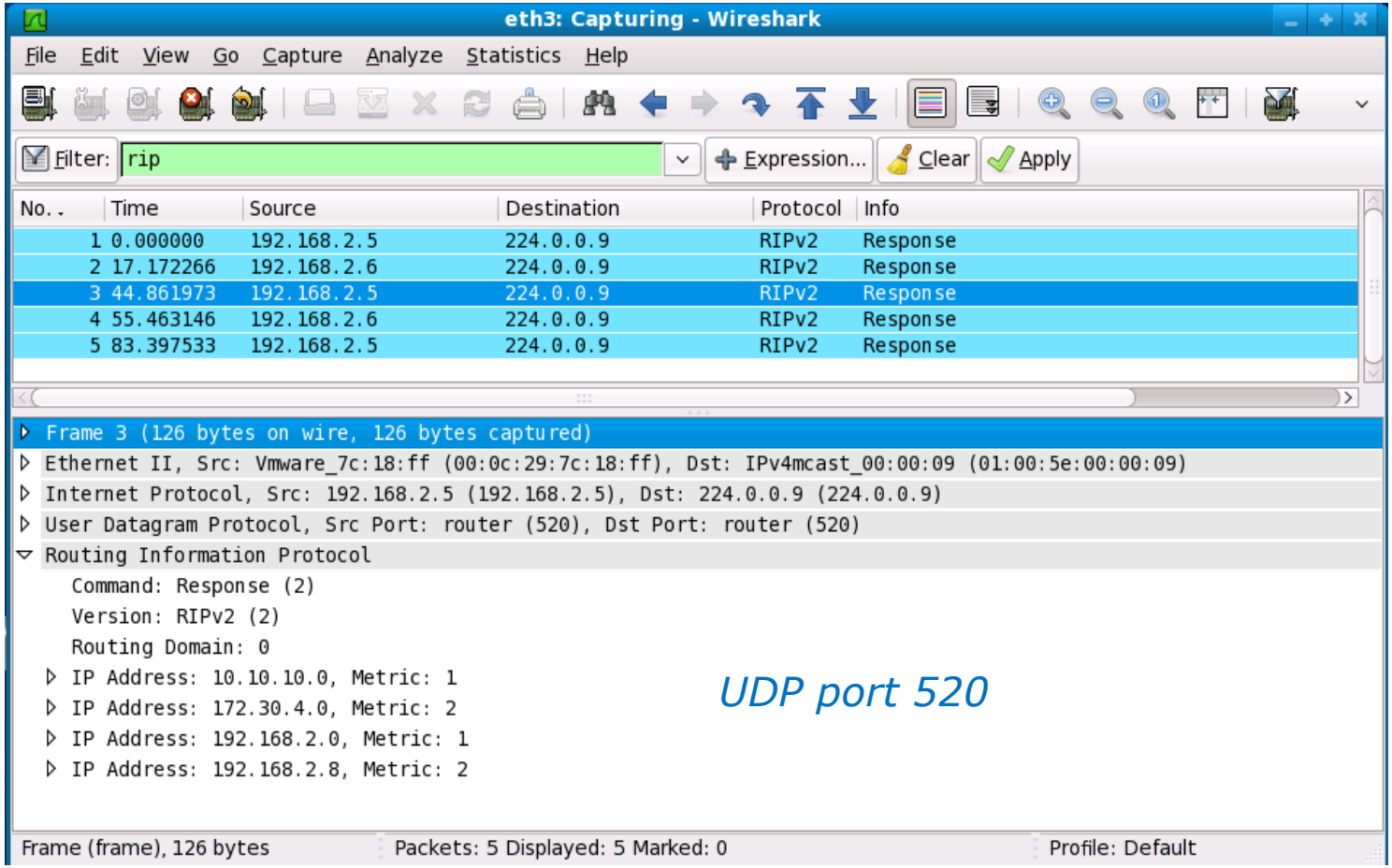
```
[root@arwen ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.3.5 on Thu Feb 26 08:22:29 2009
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [946:71747]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp -m icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p esp -j ACCEPT
-A RH-Firewall-1-INPUT -p ah -j ACCEPT
-A RH-Firewall-1-INPUT -d 224.0.0.251 -p udp -m udp --dport 5353 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m state --state NEW -m udp --dport 520 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Thu Feb 26 08:22:29 2009
[root@arwen ~]#
```

No filtering now on any forwarded packets

RIP (UDP port 520) and Telnet (TCP port 23) ports open

Quagga and the Firewall

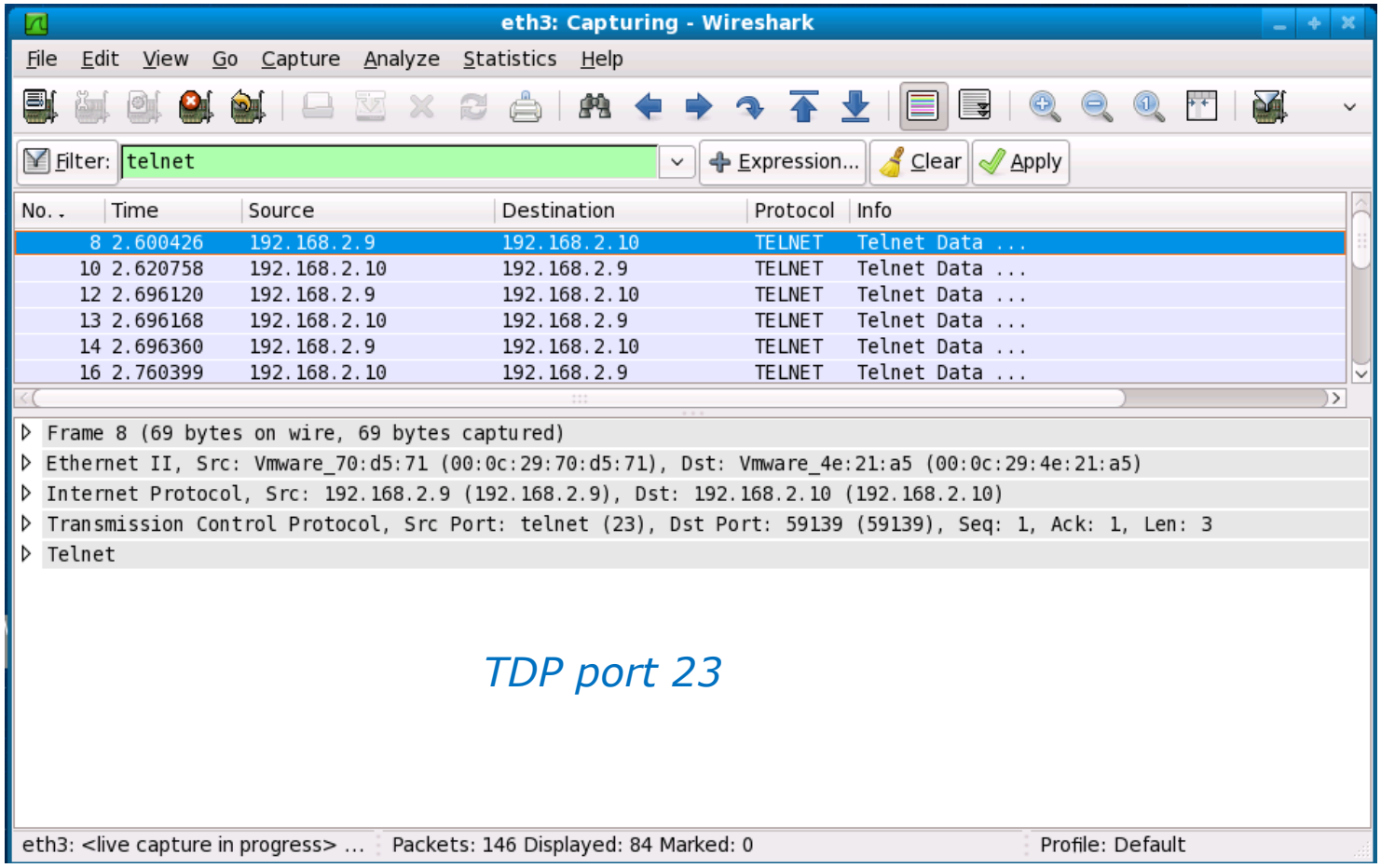
We would like RIP updates to be passed between the routers



UDP port 520

Modifying the Firewall
(Centos)

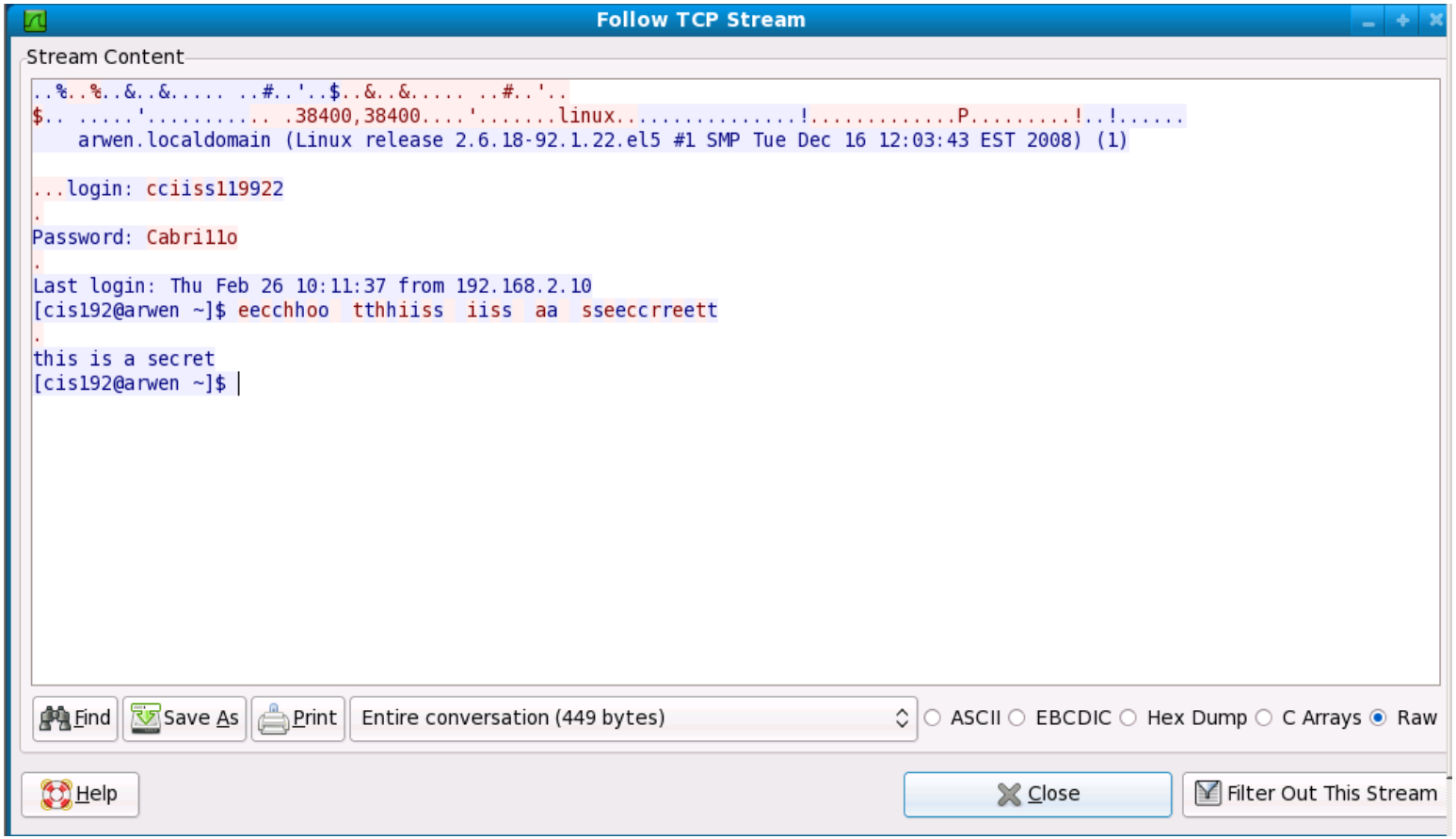
We would like Arwen to accept Telnet sessions



TDP port 23

Modifying the Firewall
(Centos)

BTW … this is why we use SSH!
We are using a Telnet server in Lab 4 so we don't forget why!



/etc/quagga/zebra.conf and /etc/quagga/ripd.conf

Zebra service configuration file

```
[root@legolas quagga]# cat /etc/quagga/zebra.conf
hostname legolas
password <password>
log stdout
log file /var/log/quagga/zebra.log
```

/etc/quagga/zebra.conf and /etc/quagga/ripd.conf

```
[root@legolas ~]# cat /etc/quagga/ripd.conf
!
! Zebra configuration saved from vty
!   2009/02/25 16:36:10
!
hostname legolas(ripd)
password <password>
log file /var/log/quagga/ripd.log
!
debug rip events
debug rip zebra
!
interface eth0
 no ip rip authentication mode text
 no ip rip authentication mode md5
!
interface eth1
 no ip rip authentication mode text
 no ip rip authentication mode md5
!
router rip
 version 2
 redistribute connected
 redistribute static
 network eth0
 network eth1
!
!line vty
!
[root@legolas ~]#
```

ripd service configuration file