

Lesson Module Status

- Slides – draft
- Properties - done
- Flashcards - na
- 1st minute quiz – na, test today
- Web Calendar summary – done
- Web book pages –
- Commands – done
- Howtos – na
- Skills pacing - na
- Lab – na
- Depot (VMs) – added VBox VMs
- Labs published – na, no new labs this week
- Real test publish – done
- Real test copies made – done
- Set up mystery server for test driver question - done
- New quiz ?'s for next week – NA
- Add email option for all lesson quizzes

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/>



www.cccconfer.org
dial-in: 888-886-3951
passcode: 439080

▶ STUDENT LOG IN
▶ View Teach & Confer Archives



Joe A.



Joe P.



Chris B.



Chuck



Rich



Josh



Ryan



John



Robert



Chris H.



Lieven



Jack



Patrick



Casady



Kay



Edwin



Julio



Drew



Edgar



Aaron



Junious



Joe B.



Brynden

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

Quiz

No Quiz Today

The Application Layer

Objectives

- Use basic network terminology to describe the five layers of the TCP/IP Reference Model, and describe at least one major function of each layer.
- Configure a network service with security restrictions for its use using either TCP Wrappers or a superdaemon.

Agenda

- No Quiz today
- Questions on previous material
- Housekeeping
- Review
- TCP continued
- Security issues
- Application layer
- xinetd and Telnet
- Very Secure FTP
- Wrap
- Test 1

Questions on previous material



Questions?

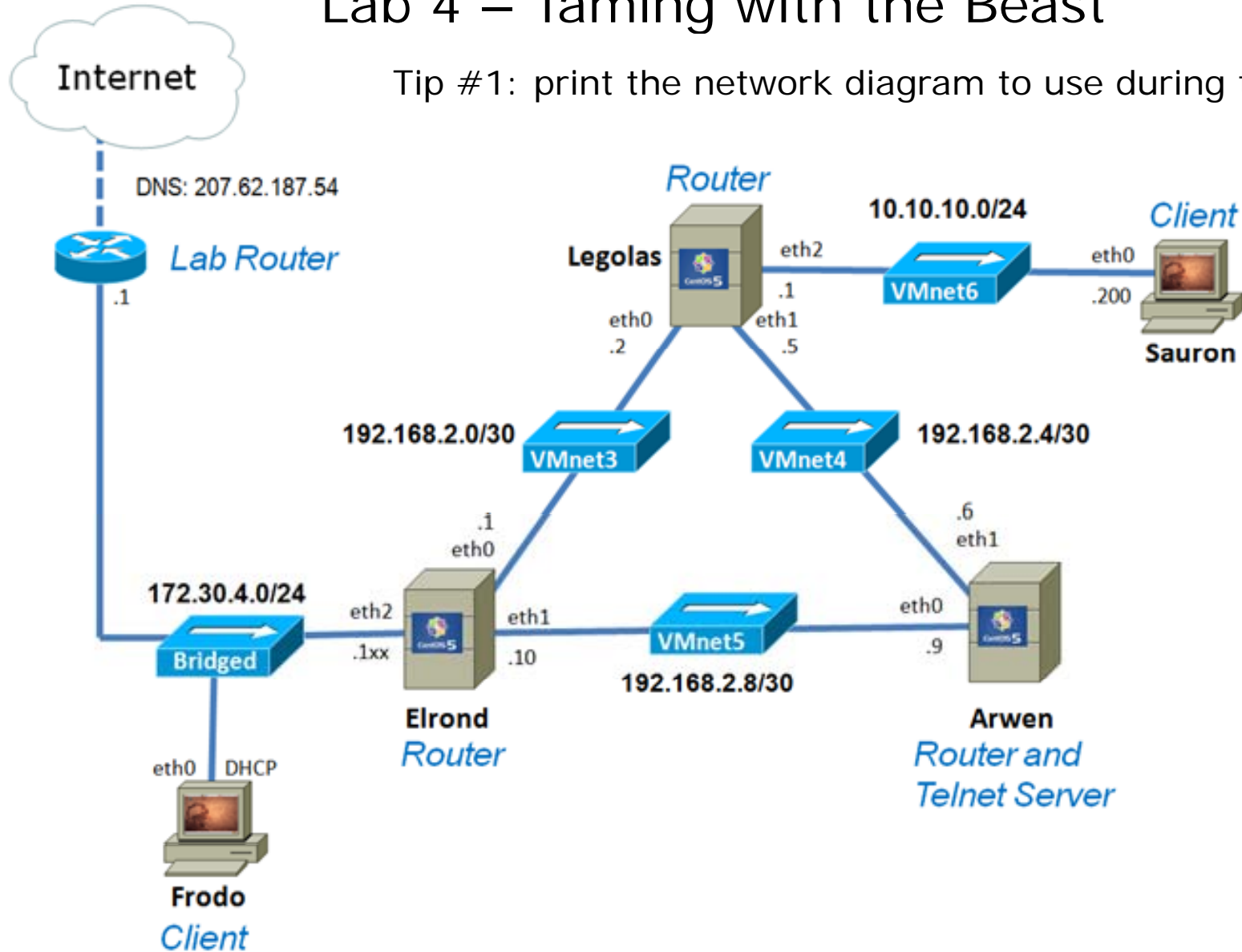
- Previous lesson material
- Lab assignment
- How this class works

Taming the Beast

(Lab 4)

Lab 4 – Taming with the Beast

Tip #1: print the network diagram to use during the lab



Lab 4 – Taming the Beast

Tip #2: Use permanent network settings

On Legolas ...

```
[root@legolas ~]# head /etc/sysconfig/{network,network-scripts/ifcfg-eth[0-3]}
==> /etc/sysconfig/network <==
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=legolas.localdomain
GATEWAY=192.168.2.1

==> /etc/sysconfig/network-scripts/ifcfg-eth0 <==
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.2.2
NETMASK=255.255.255.252

==> /etc/sysconfig/network-scripts/ifcfg-eth1 <==
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.2.5
NETMASK=255.255.255.252

==> /etc/sysconfig/network-scripts/ifcfg-eth2 <==
DEVICE=eth2
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.10.10.1
NETMASK=255.255.255.0
[root@legolas ~]#
```

*You can restart
the VM or the
network services
without having to
reconfigure
interfaces*

*... do the same on the
other VMs*

Lab 4 – Taming the Beast

Tip #3: Populate /etc/hosts files with names used in Lab 4

On Elrond ...

```
[root@elrond ~]# cat /etc/hosts
# Do not remove the following line, or various
programs
# that require network functionality will fail.
127.0.0.1          elrond.localdomain
elrond localhost.localdomain localhost
::1               localhost6.localdomain6
localhost6
```

```
172.30.4.1 router
192.168.2.2 legolas
192.168.2.9 arwen
10.10.10.200 sauron
```

```
[root@elrond ~]#
```

Do the same for Arwen, Frodo, and Sauron and then you can use names rather than IP address for testing and troubleshooting

On Legolas ...

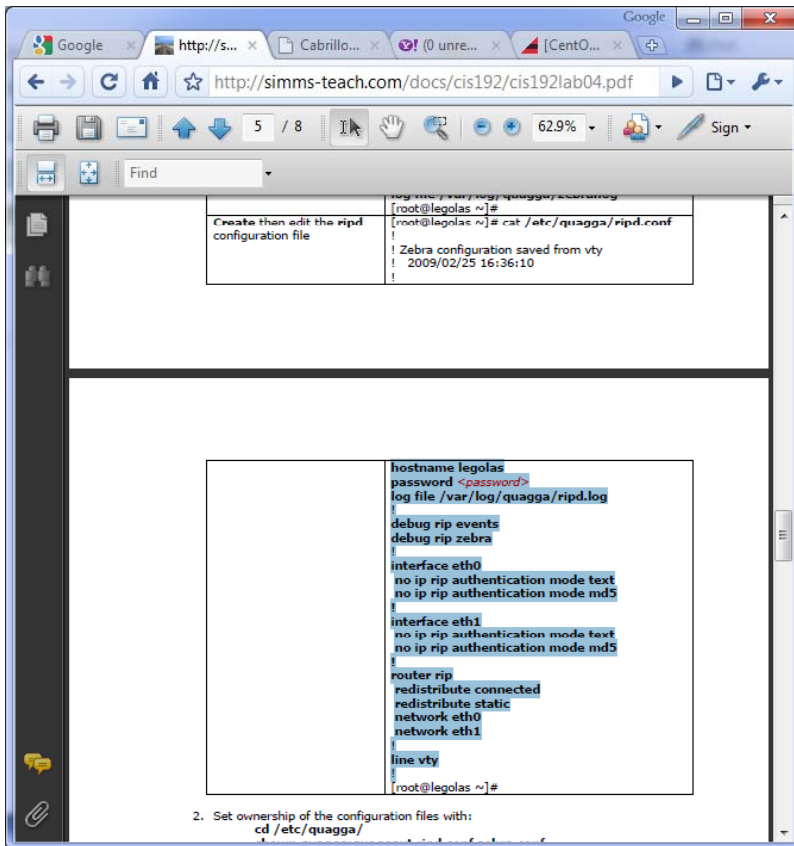
```
[root@legolas ~]# cat /etc/hosts
# Do not remove the following line, or various
programs
# that require network functionality will fail.
127.0.0.1          legolas.localdomain
legolas localhost.localdomain localhost
::1               localhost6.localdomain6
localhost6
```

```
192.168.2.6 arwen
192.168.2.1 elrond
172.30.4.1 router
10.10.10.200 sauron
```

```
[root@legolas ~]#
```

Lab 4 – Taming the Beast

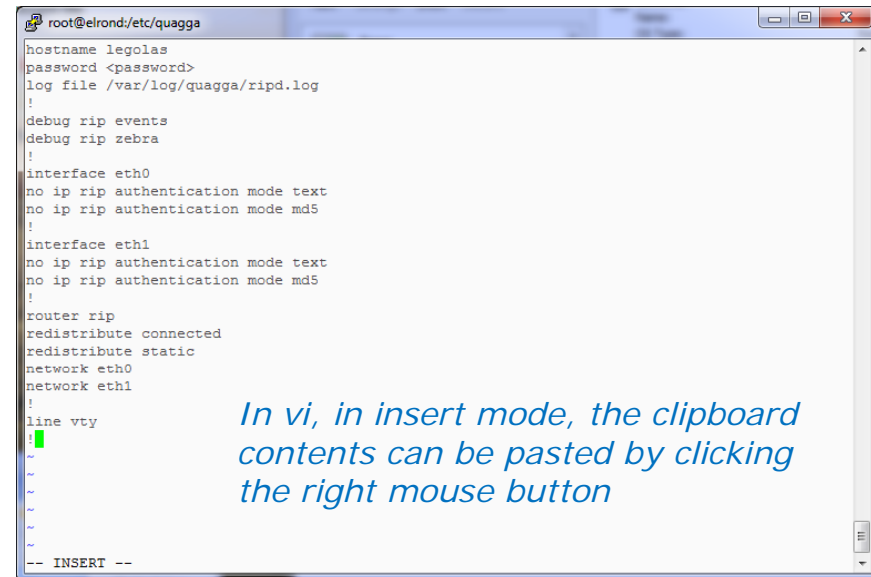
Tip #4: Use Putty to copy/paste Quagga configuration files



Lab 4 PDF

Putty into Elrond at 172.30.x.1xx ...

```
login as: root
root@172.30.1.125's password:
Last login: Tue Mar  9 23:04:46 2010 from
172.30.1.150
[root@elrond ~]# cd /etc/quagga
[root@elrond quagga]# vi ripd.conf
```



/etc/quagga/ripd.conf on Elrond

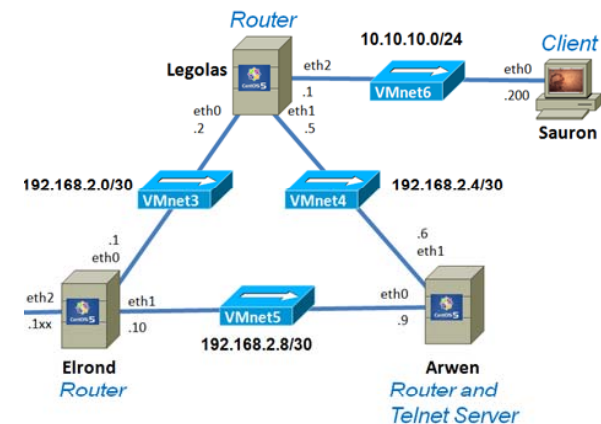
After copying, be sure and modify the hostname to match the system you are configuring

Lab 4 – Taming the Beast

Tip #5: Use scp command to copy Quagga configuration files

After Elrond is configured, ssh into Legolas and use scp to copy Quagga configuration files from Elrond. Repeat with Arwen.

```
Using username "root".
root@172.30.1.125's password:
Last login: Tue Mar  9 22:43:57 2010
[root@elrond ~]# ssh legolas
root@legolas's password:
Last login: Tue Mar  9 22:04:50 2010
[root@legolas ~]# cd /etc/quagga
[root@legolas quagga]# scp elrond:/etc/quagga/[rz]*conf .
The authenticity of host 'elrond (192.168.2.1)' can't be established.
RSA key fingerprint is ec:e3:71:94:41:bb:9a:c6:61:a8:06:2c:ed:f2:d7:7c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'elrond,192.168.2.1' (RSA) to the list of known hosts.
root@elrond's password:
ripd.conf                100%  400    0.4KB/s   00:00
zebra.conf                100%   70    0.1KB/s   00:00
[root@legolas quagga]#
```



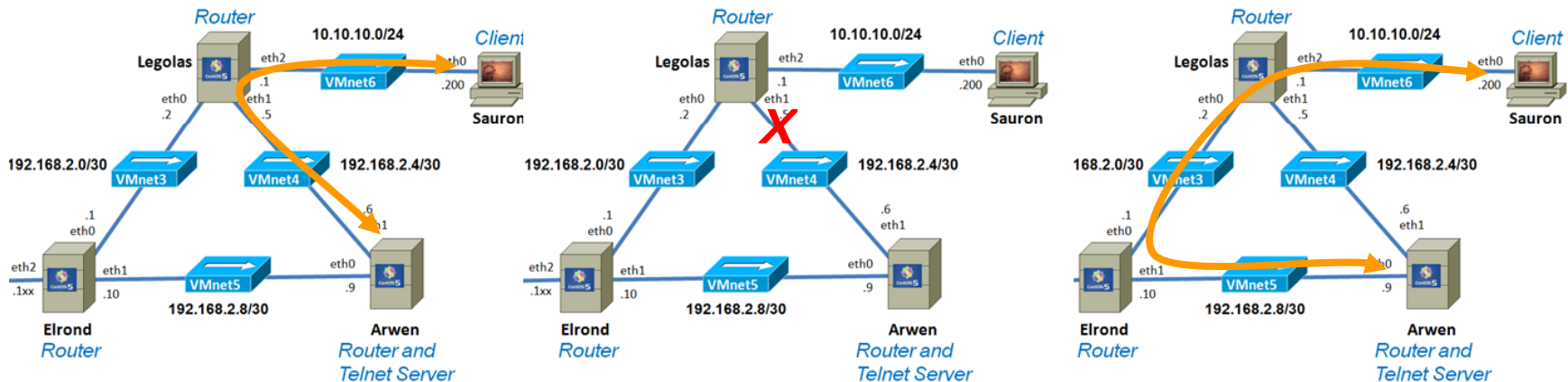
After copying, be sure and modify the hostname to match the system you are configuring

Playing with the Beast

(Lab 4)

Lab 4 – Playing with the Beast

Playing #1: Force routing table to adapt to network changes you make



Pinging Sauron from Arwen via Legolas

Making trouble: The cable attached to the second Ethernet Controller on Legolas (eth1) is disconnected

After 180 failed pings, routing tables adjust and a new, longer route via Elrond and Legolas is used

In Lab 4 you can observe routing tables update themselves as the network changes

Lab 4 – Playing with the Beast

```
[root@arwen ~]# 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
172.30.1.0       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        0     0      0 eth1
```

```
[root@arwen ~]# ping sauron
PING sauron (10.10.10.200) 56(84) bytes of data:
64 bytes from sauron (10.10.10.200): icmp_seq=1 ttl=63 time=5.60 ms
< snipped >
64 bytes from sauron (10.10.10.200): icmp_seq=7 ttl=63 time=6.20 ms
From 192.168.2.6 icmp_seq=53 Destination Host Unreachable
< snipped >
From 192.168.2.6 icmp_seq=181 Destination Host Unreachable
ping: sendmsg: Network is unreachable
< snipped >
ping: sendmsg: Network is unreachable
64 bytes from sauron (10.10.10.200): icmp_seq=188 ttl=62 time=7.16 ms
< snipped >
64 bytes from sauron (10.10.10.200): icmp_seq=226 ttl=62 time=6.27 ms
```

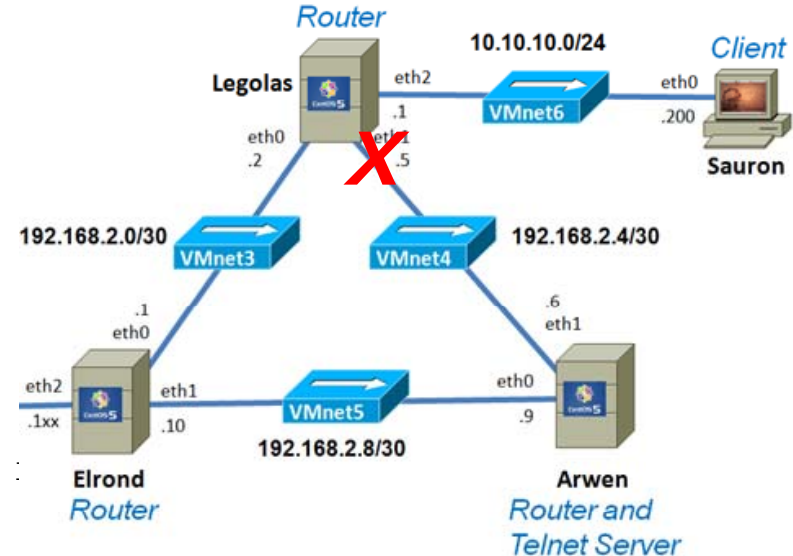
```
--- sauron ping statistics ---
226 packets transmitted, 46 received, +126 errors, 79% packet :
rtt min/avg/max/mdev = 0.142/5.446/21.506/3.115 ms, pipe 4
```

```
[root@arwen ~]# 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      192.168.2.10    255.255.255.252 UG       2     0      0 eth0
192.168.2.4      0.0.0.0         255.255.255.252 U        0     0      0 eth1
172.30.1.0       192.168.2.10    255.255.255.0   UG       2     0      0 eth0
10.10.10.0       192.168.2.10    255.255.255.0   UG       3     0      0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U        0     0      0 eth1
```

```
[root@arwen ~]#
```

Pinging Sauron from Arwen via Legolas

Trouble: cable on Legolas eth1 is disconnected



After 180 failed pings, routing tables adjust and now use longer route via Elrond and Legolas

Lab 4 – Playing with the Beast

Playing #2: Debug RIP events and packets

```
[root@legolas ~]# vtysh
```

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

```
legolas.localdomain# debug rip events  
legolas.localdomain# debug rip packet  
legolas.localdomain# exit
```

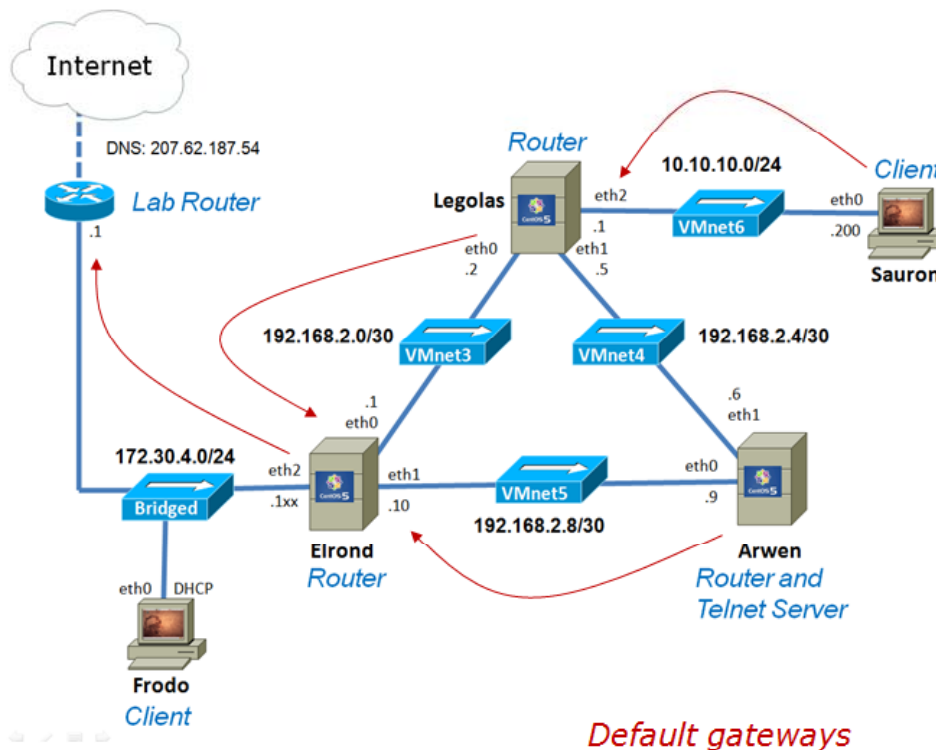
*Use the debug command to
enable debugging*

```
[root@legolas ~]# tail -f /var/log/quagga/ripd.log  
2010/03/10 00:39:43 RIP: ignore packet comes from myself  
2010/03/10 00:39:47 RIP: RECV packet from 192.168.2.1 port 520 on eth0  
2010/03/10 00:39:47 RIP: RECV RESPONSE version 2 packet size 64  
2010/03/10 00:39:47 RIP: 0.0.0.0/0 -> 0.0.0.0 family 2 tag 0 metric 1  
2010/03/10 00:39:47 RIP: 172.30.1.0/24 -> 0.0.0.0 family 2 tag 0 metric 1  
2010/03/10 00:39:47 RIP: 192.168.2.8/30 -> 0.0.0.0 family 2 tag 0 metric 1  
2010/03/10 00:39:55 RIP: RECV packet from 192.168.2.6 port 520 on eth1  
2010/03/10 00:39:55 RIP: RECV RESPONSE version 2 packet size 84  
[root@legolas ~]#
```

*Use tail with the -f option to monitor debug messages as
they are written to /var/quagga/ripd.conf*

Lab 4 – Playing with the Beast

Playing #3: Use RIP to set default gateways on Legolas and Arwen



In Lab 4 we manually set up default routes before configuring dynamic routing on Elrond, Legolas, and Arwen

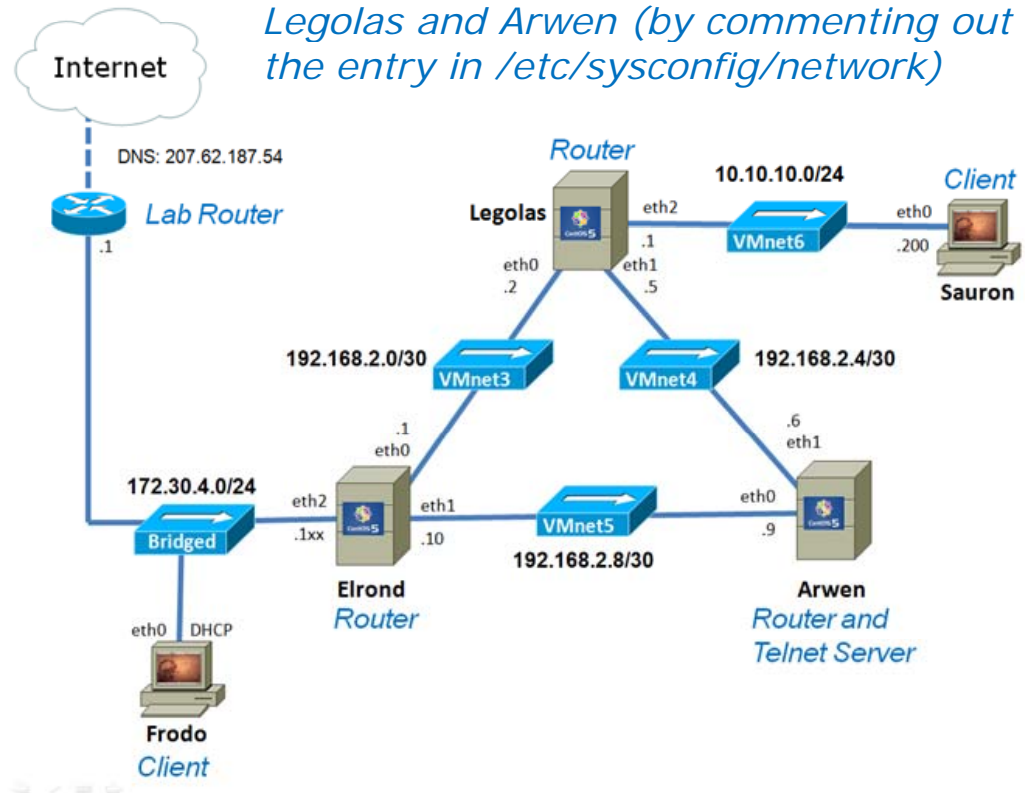
*Rather than manually configuring the default gateway on Legolas and Arwen, Elrond can propagate its default gateway using the RIP **default-information originate** command to Legolas and Arwen*

Lab 4 – Playing with the Beast

Playing #3: Use RIP to set default gateways on Legolas and Arwen

```
[root@elrond ~]# cat /etc/quagga/ripd.conf
hostname elrond
password Cabrillo
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
default-information originate
!
line vty
!
[root@elrond ~]#
```

The default gateways were removed on Legolas and Arwen (by commenting out the entry in /etc/sysconfig/network)



*The **default-information originate** command is added just on Elrond (not Legolas or Arwen) to propagate the default gateway using RIP to Legolas and Arwen*

Lab 4 – Playing with the Beast

Playing #3: Use RIP to set default gateways on Legolas and Arwen

```
[root@elrond ~]# ssh legolas
root@legolas's password:
Last login: Tue Mar  9 23:36:55 2010 from elrond
```

... on Legolas

```
[root@legolas ~]# route -n
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
192.168.2.8      192.168.2.1     255.255.255.252 UG    2     0      0 eth0
192.168.2.0      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
172.30.1.0       192.168.2.1     255.255.255.0   UG    2     0      0 eth0
10.10.10.0       0.0.0.0         255.255.255.0   U      0     0      0 eth2
169.254.0.0      0.0.0.0         255.255.0.0     U      0     0      0 eth2
0.0.0.0          192.168.2.1     0.0.0.0         UG    2     0      0 eth0
```

```
[root@legolas ~]# vtysh
```

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

```
legolas.localdomain# sh ip route
```

```
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route
```

```
R>* 0.0.0.0/0 [120/2] via 192.168.2.1, eth0, 00:55:43
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, eth2
R>* 172.30.1.0/24 [120/2] via 192.168.2.1, eth0, 00:55:43
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, 00:55:43
legolas.localdomain#
```

*Legolas has learned the default gateway from Elrond using RIP because of the **default-information originate** command added on Elrond*

Lab 4 – Playing with the Beast

Playing #3: Use RIP to set default gateways on Legolas and Arwen

```
[root@legolas ~]# vtysh
```

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

*With the **default-information originate** command added on Elrond ...*

```
legolas.localdomain# debug rip events
legolas.localdomain# debug rip packet
legolas.localdomain# exit
[root@legolas ~]# tail -f /var/log/quagga/ripd.log
2010/03/10 00:39:43 RIP: ignore packet comes from myself
2010/03/10 00:39:47 RIP: RECV packet from 192.168.2.1 port 520 on eth0
2010/03/10 00:39:47 RIP: RECV RESPONSE version 2 packet size 64
2010/03/10 00:39:47 RIP: 0.0.0.0/0 -> 0.0.0.0 family 2 tag 0 metric 1
2010/03/10 00:39:47 RIP: 172.30.1.0/24 -> 0.0.0.0 family 2 tag 0 metric 1
2010/03/10 00:39:47 RIP: 192.168.2.8/30 -> 0.0.0.0 family 2 tag 0 metric 1
2010/03/10 00:39:55 RIP: RECV packet from 192.168.2.6 port 520 on eth1
2010/03/10 00:39:55 RIP: RECV RESPONSE version 2 packet size 84
[root@legolas ~]# vtysh
```

*On Legolas ...
RIP updates from Elrond
include the default gateway*

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

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

Legolas adds a default gateway

```
R>* 0.0.0.0/0 [120/2] via 192.168.2.1, eth0, 01:19:50
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, eth2
R>* 172.30.1.0/24 [120/2] via 192.168.2.1, eth0, 01:19:50
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, 01:19:50
legolas.localdomain#
```

Lab 4 – Playing with the Beast

Playing #3: Use RIP to set default gateways on Legolas and Arwen

```
[root@legolas ~]# vtysh
```

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

```
legolas.localdomain# debug rip events  
legolas.localdomain# debug rip packet  
legolas.localdomain# exit
```

```
[root@legolas ~]# tail -f /var/log/quagga/ripd.log  
2010/03/10 00:48:46 RIP: RECV packet from 192.168.2.1 port 520 on eth0  
2010/03/10 00:48:46 RIP: RECV RESPONSE version 2 packet size 44  
2010/03/10 00:48:46 RIP: 172.30.1.0/24 -> 0.0.0.0 family 2 tag 0 metric 1  
2010/03/10 00:48:46 RIP: 192.168.2.8/30 -> 0.0.0.0 family 2 tag 0 metric 1  
2010/03/10 00:49:01 RIP: update timer fire!  
[root@legolas ~]# vtysh
```

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

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

```
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, eth2  
R>* 172.30.1.0/24 [120/2] via 192.168.2.1, eth0, 01:29:20  
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, 01:29:20  
legolas.localdomain#
```

Without a default-information originate command added on Elrond ...

*On Legolas ...
RIP updates from Elrond
are limited to connected
networks.*

*Legolas no longer
has a default
gateway*

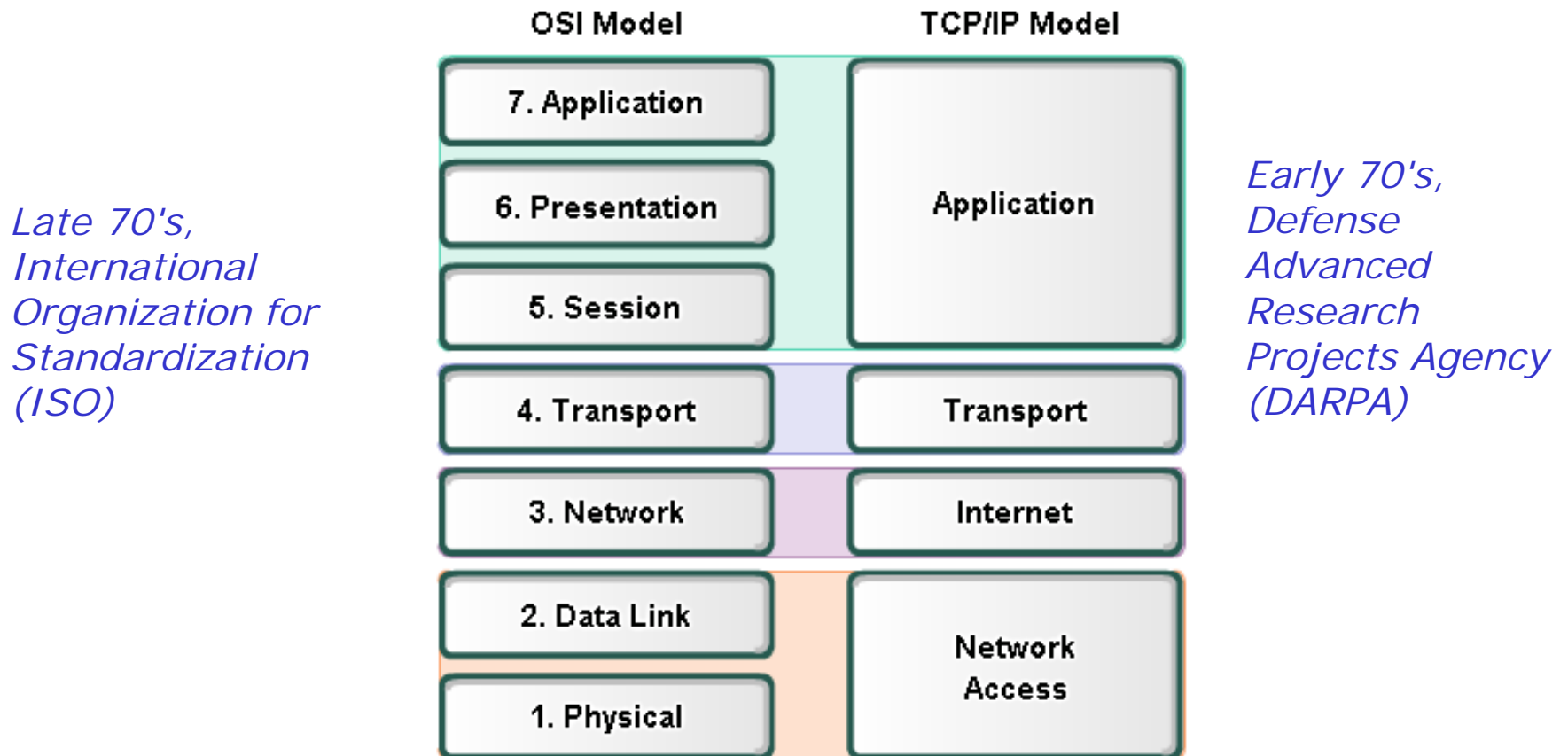
Housekeeping

- Nothing due today!
- MSDN AA accounts
- CCC Confer
 - Group chat window test
 - 2nd microphone solution ??
- VTEA/Perkins Survey from Rock
 - Allows Cabrillo to qualify for grant funds
 - Confidential and paper surveys are shredded
 - Not necessary if completed in another class this Spring 2009 semester
 - Must have correct Name, SSN or Student ID and birthday
- Grades page check

Random Review

Layers

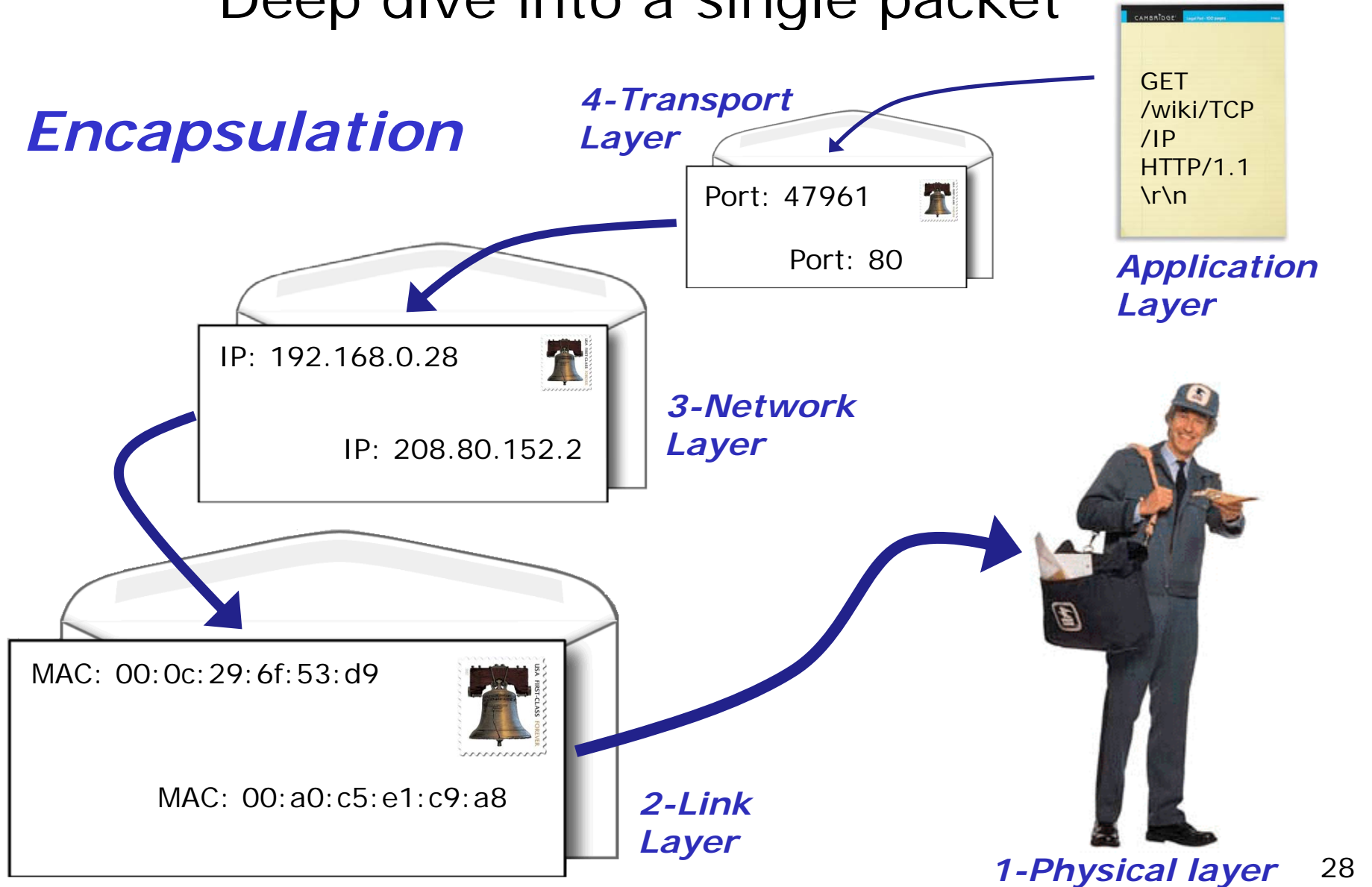
Protocol and Reference Models



- The **Open Systems Interconnection (OSI)** model is the *most widely known internetwork reference model*.

Deep dive into a single packet

Encapsulation



Deep dive into a single packet

Note how Wireshark shows each layer for the selected HTTP GET packet

- 1-Physical →
- 2-Link →
- 3-Network →
- 4-Transport →
- Application →

The screenshot shows the Wireshark interface with a packet capture. The packet list pane shows the following packets:

No.	Time	Source	Destination	Protocol	Info
2189	32.195688	192.168.0.27	192.168.0.32	TCP	60999 > ms-wbt-server [ACK] Seq=19632
2190	32.206077	192.168.0.32	192.168.0.27	TPKT	Continuation
2191	32.227457	208.80.152.2	192.168.0.28	TCP	http > 47961 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0
2192	32.227811	192.168.0.28	208.80.152.2	TCP	47961 > http [ACK] Seq=1 Ack=1 Win=582 Len=0
2193	32.228731	192.168.0.28	208.80.152.2	HTTP	GET /wiki/TCP/IP HTTP/1.1
2194	32.306985	192.168.0.32	192.168.0.27	TPKT	Continuation

The packet details pane for the selected packet (2193) shows the following layers:

- Frame 2193 (636 bytes on wire, 636 bytes captured)
- Ethernet II, Src: Vmware_6f:53:d9 (00:0c:29:6f:53:d9), Dst: ZyxelCom_e1:c9:a8 (00:a0:c5:e1:c9:a8)
- Internet Protocol, Src: 192.168.0.28 (192.168.0.28), Dst: 208.80.152.2 (208.80.152.2)
- Transmission Control Protocol, Src Port: 47961 (47961), Dst Port: http (80), Seq: 1, Ack: 1, Len: 582
- Hypertext Transfer Protocol

The packet bytes pane shows the raw data of the packet:

```

0000 00 a0 c5 e1 c9 a8 00 0c 29 6f 53 d9 08 00 45 00  .... )oS...E.
0010 02 6e 5b 3e 40 00 40 06 b4 34 c0 a8 00 1c d0 50  .n[>@.@. .4....P
0020 98 02 bb 59 00 50 56 18 29 23 78 7c 57 9b 50 18  ...Y.PV. )#x|W.P.
0030 00 b7 48 00 00 00 47 45 54 20 2f 77 69 6b 69 2f  ..H...GE T /wiki/
0040 54 43 50 2f 49 50 20 48 54 54 50 2f 31 2e 31 0d  TCP/IP H TTP/1.1.
0050 0a 48 6f 73 74 3a 20 65 6e 2e 77 69 6b 69 70 65  .Host: e n.wikipe
0060 64 69 61 2e 6f 72 67 0d 0a 55 73 65 72 2d 41 67  dia.org. .User-Ag
0070 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30  ent: Moz illa/5.0
0080 20 28 58 31 31 3b 20 55 3b 20 4c 69 6e 75 78 20  (X11; U ; Linux
    
```

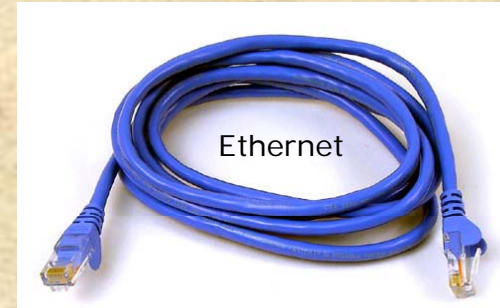
Frame (frame), 636 bytes Packets: 4260 Displayed: 4260 Marked: 0 Dropped: 0 Profile: Default



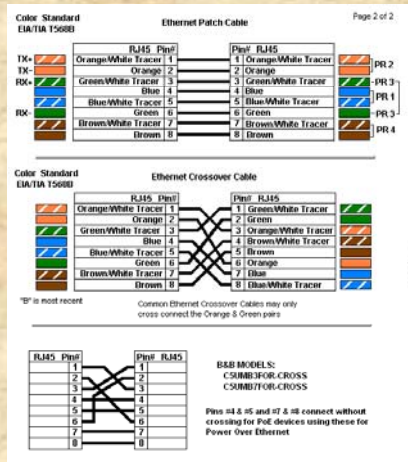
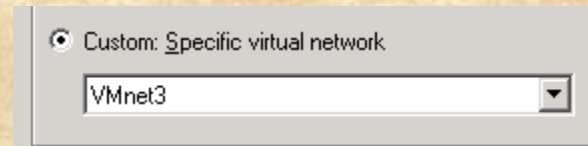
What layer is this?



Fiber Optic



Ethernet



Serial



Fiber to Ethernet adapter



Hub

Network connection

- Bridged: Connected directly to the physical network



What layer is this?

```
ifconfig eth0 172.30.4.125 netmask 255.255.255.0
```

Header

0		15		16		31	
4-bit Version	4-bit Header Length	8-bit Type Of Service (TOS)		16-bit Total Length (in bytes)			
16-bit Identification				3-bit Flags	13-bit Fragment Offset		
8 bit Time To Live TTL		8-bit Protocol		16-bit Header Checksum			
32-bit Source IP Address							
32-bit Destination IP Address							
Options (if any)							

```
ping 172.30.1.1
```

```
route add default gw 172.30.1.1
```

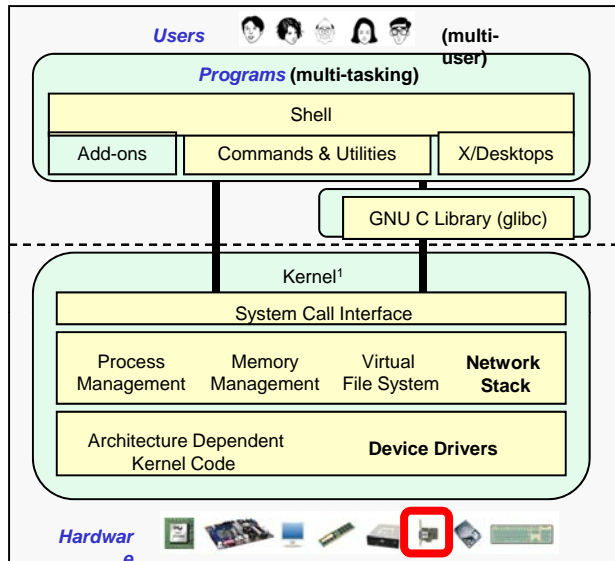
```

Internet Protocol, Src: 172.30.1.150 (172.30.1.150), Dst: 207.211.65.20 (207.211.65.20)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 752
    Identification: 0x6d71 (28017)
  Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 128
    Protocol: TCP (0x06)
  Header checksum: 0xcbfa [correct]
    Source: 172.30.1.150 (172.30.1.150)
    Destination: 207.211.65.20 (207.211.65.20)
  
```

Wireshark view

NIC/Driver Inventory

NIC Hardware Inventory



- The Network Stack is built into the Linux kernel
- NIC drivers are dynamic kernel modules that can be loaded and unloaded while Linux is running
- Linux systems can have one or more NICs to connect with one or more networks



NIC Hardware Inventory



Use **lspci** to show Ethernet controllers

```
[root@celebrian ~]# lspci
00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host bridge (rev 01)
00:01.0 PCI bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX AGP bridge (rev 01)
00:07.0 ISA bridge: Intel Corporation 82371AB/EB/MB PIIX4 ISA (rev 08)
00:07.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:07.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:0f.0 VGA compatible controller: VMware Inc Abstract SVGA II Adapter
00:10.0 SCSI storage controller: LSI Logic / Symbios Logic 53c1030 PCI-X Fusion-MPT Dual
  Ultra320 SCSI (rev 01)
00:11.0 Ethernet controller: Advanced Micro Devices [AMD] 79c970 [PCnet32 LANCE] (rev 10)
00:12.0 Ethernet controller: Advanced Micro Devices [AMD] 79c970 [PCnet32 LANCE] (rev 10)
[root@celebrian ~]#
```

This Celebrian VM has two AMD 79C970 NICs installed

NIC Hardware Inventory



Use **lspci** to show Ethernet controllers

```

cis192@hershey:~
[cis192@hershey cis192]$ /sbin/lspci
00:00.0 Host bridge: Intel Corp. 82845 845 (Brookdale) Chipset Host Bridge (rev
11)
00:01.0 PCI bridge: Intel Corp. 82845 845 (Brookdale) Chipset AGP Bridge (rev 11
)
00:1e.0 PCI bridge: Intel Corp. 82801BA/CA/DB PCI Bridge (rev 05)
00:1f.0 ISA bridge: Intel Corp. 82801BA ISA Bridge (LPC) (rev 05)
00:1f.1 IDE interface: Intel Corp. 82801BA IDE U100 (rev 05)
00:1f.2 USB Controller: Intel Corp. 82801BA/BAM USB (Hub #1) (rev 05)
00:1f.3 SMBus: Intel Corp. 82801BA/BAM SMBus (rev 05)
00:1f.4 USB Controller: Intel Corp. 82801BA/BAM USB (Hub #2) (rev 05)
02:0c.0 Ethernet controller: Intel Corp. 82557/8/9 [Ethernet Pro 100] (rev 0d)
02:0d.0 Ethernet controller: Intel Corp. 82557/8/9 [Ethernet Pro 100] (rev 0d)
02:0f.0 VGA compatible controller: ATI Technologies Inc Rage XL (rev 27)
[cis192@hershey cis192]$ █

```

`/sbin/lspci`

Hershey has two Intel 82557/8/9 Pro 100 NICs installed

NIC Hardware Inventory



Use **lspci** to show Ethernet controllers

Tip: Add /sbin to command if you are not logged in as root

```
[cis192@elrond ~]$ /sbin/lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:02.0 VGA compatible controller: InnoTek Systemberatung GmbH VirtualBox Graphics Adapter
00:03.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
    (Copper) (rev 02)
00:04.0 System peripheral: InnoTek Systemberatung GmbH VirtualBox Guest Service
00:05.0 Multimedia audio controller: Intel Corporation 82801AA AC'97 Audio Controller (rev
    01)
00:06.0 USB Controller: Apple Computer Inc. KeyLargo/Intrepid USB
00:07.0 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:08.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
    (Copper) (rev 02)
00:09.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
    (Copper) (rev 02)
00:0b.0 USB Controller: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) USB2 EHCI
    Controller
```

This Elrond VM has three Intel 82543GC NICs installed

NIC Hardware Inventory



Use **lspci** to show Ethernet controllers

```
[cis192@elrond ~]$ /sbin/lspci | grep Ethernet
00:03.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
(Copper) (rev 02)
00:08.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
(Copper) (rev 02)
00:09.0 Ethernet controller: Intel Corporation 82543GC Gigabit Ethernet Controller
(Copper) (rev 02)
[cis192@elrond ~]$
```

*Tip: Pipe **lspci** output into **grep** to make sure you don't miss any controllers which may be buried visually in the output*



NIC Drivers

<http://tldp.org/HOWTO/Ethernet-HOWTO.html>

Linux Ethernet-Howto - Windows Internet Explorer
http://tldp.org/HOWTO/Ethernet-HOWTO.html#toc1
File Edit View Favorites Tools Help
Linux Ethernet-Howto
Next Previous Contents

Linux Ethernet-Howto

by Paul Gortmaker

v2.9, Aug 25, 2003

This is the Ethernet-Howto, which is a compilation of information about which ethernet devices can be used for Linux, and how to set them up. Note that this Howto is focused on the hardware and low level driver aspect of the ethernet cards, and does not cover the software end of things like ifconfig and route. That information is found in various other Linux documentation.

1. Introduction

- 1.1 New Versions of this Document
- 1.2 Using the Ethernet-Howto
- 1.3 What do I need to
- 1.4 HELP - It doesn't
- 1.5 Type of cable that

2. Frequently Asked Questions

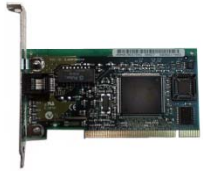
- 2.1 How do I tell Linux
- 2.2 What card should
- 2.3 Alpha Drivers --
- 2.4 Using More than

Device	Status	Driver Name
AMD 79C970/970A (PCnet-PCI)	Supported	pcnet32

This is the PCnet-PCI -- similar to the PCnet-32, but designed for PCI bus based systems. Please see the above PCnet-32 information. This means that you need to build a kernel with PCI BIOS support enabled. The '970A adds full duplex support along with some other features to the original '970 design.

Note that the Boca implementation of the 79C970 fails on fast Pentium machines. This is a hardware problem, as it affects DOS users as well. See the Boca section for more details.

*The AMD 79c970
NIC uses the
pcnet32 driver*



NIC Drivers

Googling for Linux drivers

Search results for "linux driver for 82557 Intel Pro 100".

Linux Drivers
www.Intel.com Read How Intel is a Key Contributor to the Linux* Kernel.

Network Connectivity — A guide to Intel® PRO/100 Network Adapter ... Drivers for Intel® PRO/100 Adapters (current and recently produced models) ... but usually they are all the same driver except for Linux* based downloads. ... LAN on Motherboard designs utilizing an 82557 or 82558 LAN controller. ... www.intel.com/support/network/sb/CS-006103.htm - Cached - Similar

Intel® Desktop Adapters — Where can I find drivers for the 82557 ... Nov 16, 2009 ... The Intel(R) PRO/100 adapter family is based on the 82557, ... www.intel.com/support/network/adapter/pro100/sb/cs-000596.htm

Intel® PRO/100B Adapter — Windows NT* 4.0 Installation Notes Nov 16, 2009 ... Insert the PRO/100+ Adapter Drivers Disk in the disk driver ... www.intel.com/support/network/adapter/pro100/pro100b/.../cs-008433.htm

More results from intel.com »

Intel PRO/100(82557/82558/82559) Series Ethernet Driver 4.1.3 ☆
Intel PRO/100(82557/82558/82559) Series Ethernet Driver 4.1.3 is listed for download to install, add, update, setup Intel Network.
www.opendrivers.com/driver/.../intel-pro-100(82557-82558-82559)-series-ethernet-driver-4.1.3-free-download.html - Cached - Similar

Intel PRO/100(82557/82558/82559) Series Ethernet Driver 4.1.3 ... ☆
Download Intel PRO/100(82557/82558/82559) Series Ethernet Driver 4.1.3 from Downloadatoz.com.

This Intel NIC uses the e100 driver found on the Intel web site

OS	Base Drivers	Adapter Teaming	Instrumentation (DMI/SNMP)
NetWare* 6.x or later	PRONWARE	ANSNWARE	
NetWare* 5.x	PRONW5	PRONW5	
NetWare* 4.x ³	PRONW4	PRONW4	
OS/2	PROOS2		
Linux*	e100-x.x.x.tar.gz	iANS-x.x.x.tar.gz	inic-snmip
Solaris* ²	http://www.sun.com/* ²		
UnixWare* 7.x OpenUNIX* ⁸	eeE8.pkg		
SCO5*	Please see SCO* for Intel Gigabit Network Adapter drivers. http://www.sco.com/*		

NIC Hardware Inventory



Use **dmesg** to show NIC/Driver related information

```
[root@celebrian ~]# dmesg | grep eth0
```

```
eth0: registered as PCnet/PCI II 79C970A
```

```
eth0: link up
```

```
eth0: link up
```

```
eth0: no IPv6 routers present
```

```
[root@celebrian ~]# dmesg | grep net
```

```
audit: initializing netlink socket (disabled)
```

```
SELinux: Registering netfilter hooks
```

```
Initializing IPsec netlink socket
```

```
pcnet32.c:v1.32 18.Mar.2006 tsbogend@alpha.franken.de
```

```
pcnet32: PCnet/PCI II 79C970A at 0x1400, 00 0c 29 12 50 1e assigned IRQ 177.
```

```
eth0: registered as PCnet/PCI II 79C970A
```

```
pcnet32: PCnet/PCI II 79C970A at 0x1480, 00 0c 29 12 50 28 assigned IRQ 185.
```

```
eth1: registered as PCnet/PCI II 79C970A
```

```
pcnet32: 2 cards_found.
```

```
VMware vmxnet virtual NIC driver release 1.0.8 build-126538
```

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

The Celebrian VM is using the pcnet32 driver for the two AMD 79C970 NICs

NIC Hardware Inventory



Use **dmesg** to show NIC/Driver related information

```
cis192@hershey:~  
[cis192@hershey cis192]$ dmesg | grep eth  
divert: allocating divert_blk for eth0  
e100: eth0: Intel(R) 8255x-based Ethernet Adapter  
divert: allocating divert_blk for eth1  
e100: eth1: Intel(R) 8255x-based Ethernet Adapter  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth1 NIC Link is Up 100 Mbps Full duplex  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth0 NIC Link is Down  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth0 NIC Link is Down  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth0 NIC Link is Up 100 Mbps Full duplex  
e100: eth1 NIC Link is Up 100 Mbps Full duplex  
[cis192@hershey cis192]$
```

dmesg | grep eth

Hershey is using the e100 driver for the two Intel NICs

NIC Drivers



*NIC drivers are **kernel object modules** and are located in a specific directory so the kernel knows where to find them. Note, they were .o files in older distros.*

```
[root@celebrian ~]# uname -r
2.6.18-92.1.22.el5
```

```
[root@celebrian ~]# ls /lib/modules/$(uname -r)/kernel/drivers/net
```

```
3c59x.ko      dummy.ko      natsemi.ko    ppp_synctty.ko  sunhme.ko
8139cp.ko    e1000         ne2k-pci.ko   qla3xxx.ko      tg3.ko
8139too.ko   e1000e        netconsole.ko r8169.ko         tlan.ko
8390.ko      e100.ko       netxen        s2io.ko         tokenring
acenic.ko    epic100.ko    ns83820.ko    sis190.ko       tulip
amd8111e.ko  fealnx.ko     pcmcia        sis900.ko       tun.ko
b44.ko       forcedeth.ko  pcnet32.ko    skge.ko         typhoon.ko
bnx2.ko      ifb.ko        phy           sky2.ko         via-rhine.ko
bnx2x.ko     igb           ppp_async.ko  slhc.ko         via-velocity.ko
bonding      ixgb         ppp_deflate.ko  slip.ko         wireless
cassini.ko   ixgbe        ppp_generic.ko  starfire.ko
chelsio     mii.ko       ppp_mppe.ko    sundance.ko
cxgb3       mlx4         pppoe.ko       sungem.ko
dl2k.ko     myri10ge     pppox.ko       sungem_phy.ko
```

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

```
[root@celebrian ~]# file /lib/modules/$(uname -r)/kernel/drivers/net/e100.ko
/lib/modules/2.6.18-92.1.22.el5/kernel/drivers/net/e100.ko: ELF 32-bit LSB relocatable,
  Intel 80386, version 1 (SYSV), not stripped
```

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

Commands for handling NIC drivers (kernel modules)

- To show available NIC drivers:
ls /lib/modules/\$(uname -r)/kernel/drivers/net
- To show loaded kernel modules including NIC drivers
lsmod
example: **lsmod | grep pcnet32** (show NIC drivers used on VMs)
- To remove (unload) a NIC driver
rmmod driver
example: **rmmod pcnet32** (removes pcnet32 VM NIC driver)
Do not specify the path or suffix (.ko) for drivers
- To insert (load) a NIC driver
modprobe driver
example: **modprobe pcnet32** (installs pcnet32 VM NIC driver)
modprobe is more intelligent and recommended over insmod



1. What are the NICs on the system at 172.30.1.20?
2. What driver is used for those NICs?
3. Where is that driver located?
4. What command could be used to unload that driver?
5. What command could be used to reload that driver?

ifconfig and aliases

Create an Alias IP Address (more than one IP address per interface)

Set

- To set an alias IP address and subnet mask:

```
ifconfig ethx:n xxx.xxx.xxx.xxx netmask xxx.xxx.xxx.xxx broadcast xxx.xxx.xxx.xxx
```

Verify

- To show all interfaces (and to show your IP address):

```
ifconfig
```

- To show a single alias interface:

```
ifconfig ethx:n
```

It is possible to have more than one IP address on an interface using aliases. This is different than multi-homing which is having multiple interfaces on a computer.

Create an Alias IP Address (more than one IP address per interface)

Example setting a temporary IP alias address

```
[root@elrond ~]# ifconfig eth0:1 172.30.4.122 netmask 255.255.255.0 broadcast 172.30.4.255
```

```
[root@elrond ~]#
```

```
[root@elrond ~]# ifconfig eth0:1
```

```
eth0:1 Link encap:Ethernet HWaddr 00:0C:29:82:68:7A
       inet addr:172.30.4.122 Bcast:172.30.4.255 Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
       Interrupt:177 Base address:0x1400
```

```
[root@elrond ~]# ifconfig eth0
```

```
eth0 Link encap:Ethernet HWaddr 00:0C:29:82:68:7A
      inet addr:172.30.4.121 Bcast:172.30.4.255 Mask:255.255.255.0
      inet6 addr: fe80::20c:29ff:fe82:687a/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:4863 errors:0 dropped:0 overruns:0 frame:0
      TX packets:3442 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:566772 (553.4 KiB) TX bytes:382355 (373.3 KiB)
      Interrupt:177 Base address:0x1400
```

```
[root@elrond ~]#
```

*Frodo now can ping either of
Elrond's two host IP addresses*

```
root@frodo:~# ping -c 1 172.30.1.121
PING 172.30.1.121 (172.30.1.121) 56(84) bytes of data:
64 bytes from 172.30.1.121: icmp_seq=1 ttl=127 time=6.04 ms

--- 172.30.1.121 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 6.049/6.049/6.049/0.000 ms
root@frodo:~# ping -c 1 172.30.1.122
PING 172.30.1.122 (172.30.1.122) 56(84) bytes of data:
64 bytes from 172.30.1.122: icmp_seq=1 ttl=127 time=0.900 ms

--- 172.30.1.122 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.900/0.900/0.900/0.000 ms
root@frodo:~#
```

Create an Alias IP Address (more than one IP address per interface)

Example setting a permanent IP alias address

*Create and
edit this file*

```
[root@elrond ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0:1
# Intel Corporation 82543GC Gigabit Ethernet Controller (Copper)
DEVICE=eth0:1
ONBOOT=yes
BOOTPROTO=static
IPADDR=172.30.4.122
NETMASK=255.255.255.0
```

```
[root@elrond ~]# service network restart
```

*For new settings to
take effect*

Shortcuts

Slash notation and broadcast addresses with ifconfig command

```
[root@elrond ~]# ifconfig eth1:4 10.205.163.203 netmask 255.255.248.0 broadcast
10.205.167.255
```

```
[root@elrond ~]# ifconfig eth1:4
eth1:4    Link encap:Ethernet  HWaddr 08:00:27:C1:6B:A5
          inet addr:10.205.163.203  Bcast:10.205.167.255  Mask:255.255.248.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5 Base address:0xd260
```

```
[root@elrond ~]# ifconfig eth1:4 10.205.163.203 netmask 255.255.248.0
```

```
[root@elrond ~]# ifconfig eth1:4
eth1:4    Link encap:Ethernet  HWaddr 08:00:27:C1:6B:A5
          inet addr:10.205.163.203  Bcast:10.205.167.255  Mask:255.255.248.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5 Base address:0xd260
```

```
[root@elrond ~]# ifconfig eth1:4 10.205.163.203/21
```

```
[root@elrond ~]# ifconfig eth1:4
eth1:4    Link encap:Ethernet  HWaddr 08:00:27:C1:6B:A5
          inet addr:10.205.163.203  Bcast:10.205.167.255  Mask:255.255.248.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:5 Base address:0xd260
```

*These
three
ifconfig
commands
are
equivalent*

The slash notation can be used on ifconfig commands. The broadcast address is not needed on newer distros because they don't do classful calculations

Slash notation and broadcast addresses with ifconfig command

Watch out ... older distros like Red Hat 9 do classful broadcast calculations!

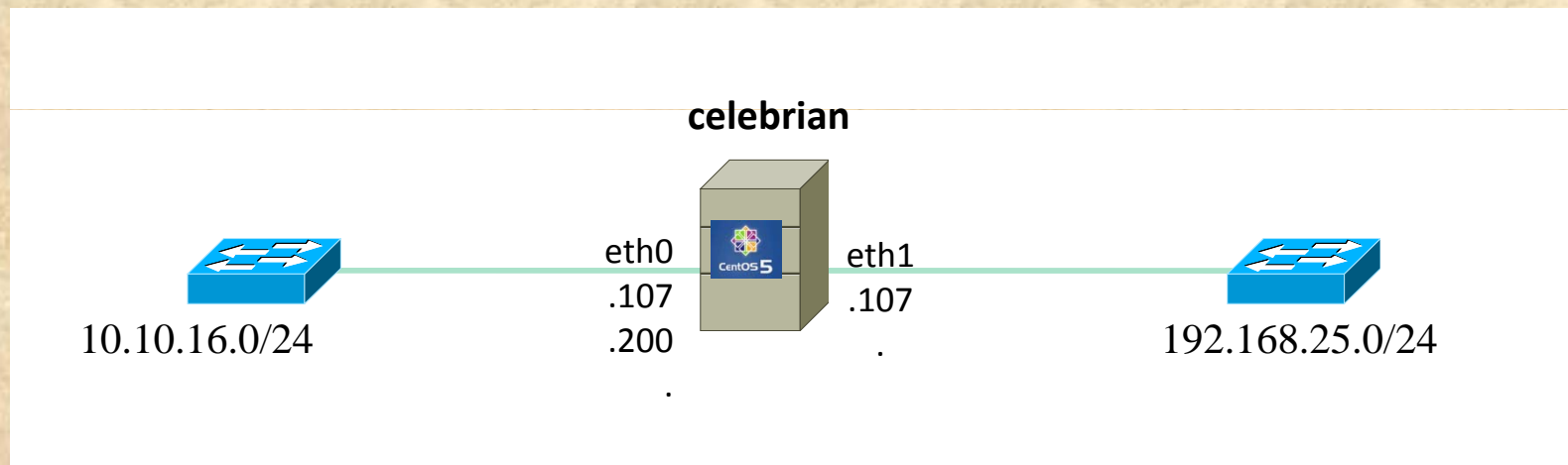
```
[root@nosmo root]# ifconfig eth0 172.30.1.221/24
[root@nosmo root]# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:46:7B:2A
          inet addr:172.30.1.221  Bcast:172.30.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:1366 (1.3 Kb)  TX bytes:838 (838.0 b)
          Interrupt:10 Base address:0xd020
```

Using class based addressing, the 172 network is a class B network yielding a 172.30.255.255 broadcast address!



On CentOS 5.4, what broadcast addresses would be calculated using the following commands:

1. **ifconfig eth0 172.30.5.227/24**
2. **ifconfig eth0 172.30.5.227 netmask 255.255.255.0**

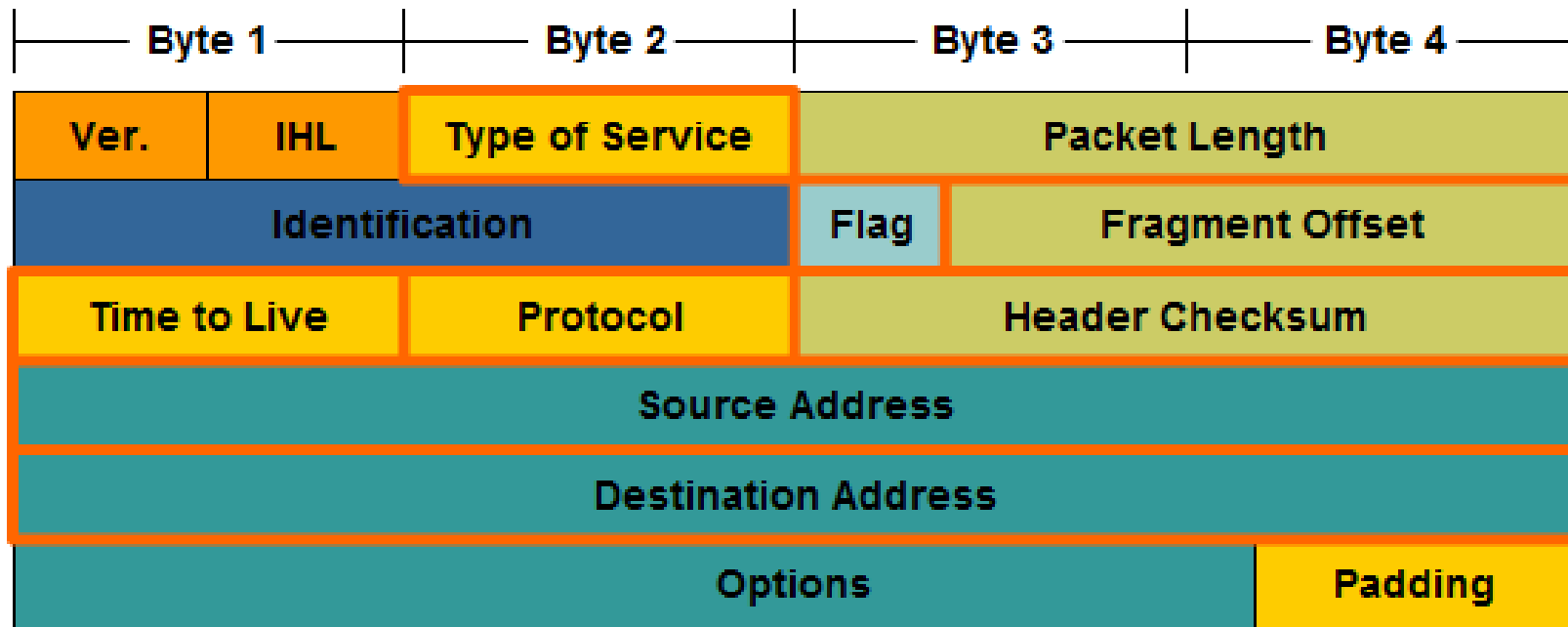


Configure Celebrian as show above then run **ifconfig**

Looking at the **ifconfig** output, what information is not displayed for eth0: 1 but is displayed for eth0?

IPv4 Addressing

IPv4 Addresses



- IPv4 addresses are 32 bit addresses

The IPv4 packet header

IPv4 Addressing

An example network, 10.0.0.0/24

```
root@frodo:~# ipcalc 10.0.0.0/24
Address: 10.0.0.0          00001010.00000000.00000000. 00000000
Netmask: 255.255.255.0 = 24 11111111.11111111.11111111. 00000000
Wildcard: 0.0.0.255      00000000.00000000.00000000. 11111111
=>
Network: 10.0.0.0/24     00001010.00000000.00000000. 00000000
HostMin: 10.0.0.1       00001010.00000000.00000000. 00000001
HostMax: 10.0.0.254     00001010.00000000.00000000. 11111110
Broadcast: 10.0.0.255   00001010.00000000.00000000. 11111111
Hosts/Net: 254          Class A, Private Internet

root@frodo:~#
```


10.0.0.0/24 network

Network Addresses
have all 0's in the
host portion.

Types of Addresses

Network Address

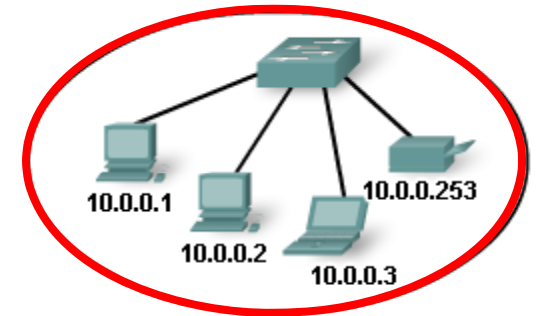
Broadcast Address

Host Address

Roll over to learn more.

Subnet Mask: 255.255.255.0

Network			Host
10	0	0	0
00001010	00000000	00000000	00000000
10	0	0	255
00001010	00000000	00000000	11111111
10	0	0	1
00001010	00000000	00000000	00000001



- **Network address** - The address by which we refer to the network
- **Broadcast address** - A special address used to send data to all hosts in the network
- **Host addresses** - The addresses assigned to the end devices in the network

Types of Addresses

Network Address

Broadcast Address

Host Address

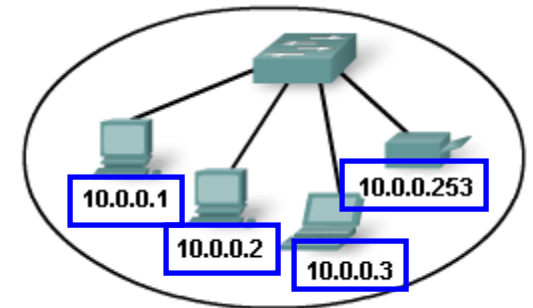
Roll over to learn more.

10.0.0.0/24 network

*Broadcast Addresses
have all 1's in the host
portion.*

Network			Host
10	0	0	0
00001010	00000000	00000000	00000000
10	0	0	255
00001010	00000000	00000000	11111111
10	0	0	1
00001010	00000000	00000000	00000001

Subnet Mask: 255.255.255.0



- **Network address** - The address by which we refer to the network
- **Broadcast address** - A special address used to send data to all hosts in the network
- **Host addresses** - The addresses assigned to the end devices in the network

Types of Addresses

Network Address

Broadcast Address

Host Address

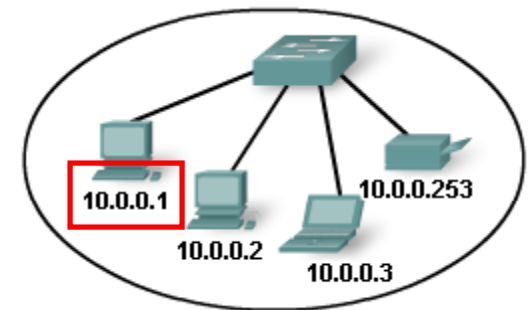
10.0.0.0/24 network

*Host Addresses can
not have all 0's or all
1's in the host portion.*

Roll over to learn more.

Subnet Mask: 255.255.255.0

Network			Host
10	0	0	0
00001010	00000000	00000000	00000000
10	0	0	255
00001010	00000000	00000000	11111111
10	0	0	1
00001010	00000000	00000000	00000001



- **Network address** - The address by which we refer to the network
- **Broadcast address** - A special address used to send data to all hosts in the network
- **Host addresses** - The addresses assigned to the end devices in the network

ipcalc command

What is the network, broadcast address and netmask for the host address 192.168.2.45/24?

```
[root@bigserver ~]# ipcalc -nbpm 192.168.2.45/24
NETMASK=255.255.255.0
PREFIX=24
BROADCAST=192.168.2.255
NETWORK=192.168.2.0
[root@bigserver ~]#
```

What is the network, broadcast address and netmask for the host address 15.20.34.2/21?

```
[root@bigserver ~]# ipcalc -nbpm 15.20.34.2/21
NETMASK=255.255.248.0
PREFIX=21
BROADCAST=15.20.39.255
NETWORK=15.20.32.0
[root@bigserver ~]#
```

Class Exercise



1. Given the following ip address and network mask, what is the broadcast address?
IP: 172.30.4.100
Netmask: 255.255.0.0
2. Given the following ip address and network mask, what is the network address?
IP: 192.168.30.100
Netmask: 255.255.240.0

ARP

TCP/IP and ARP

The TCP/IP Suite of Protocols	
Application	File Transfer: FTP, TFTP, NFS, HTTP Email: SMTP Remote Login: Telnet, rlogin Network Management: SNMP, BootP Name Management: DNS, DHCP
Transport	TCP, UDP
Internet/Network	IP, ICMP, IGMP, ARP, RARP
Network Interface (Link Layer)	Not Specified: Ethernet, 802.3, Token Ring, 802.5, FDDI, ATM,

ARP is a layer 3 protocol, one of many protocols within the TCP/IP suite of protocols.

ARP – Address Resolution Protocol

Overview

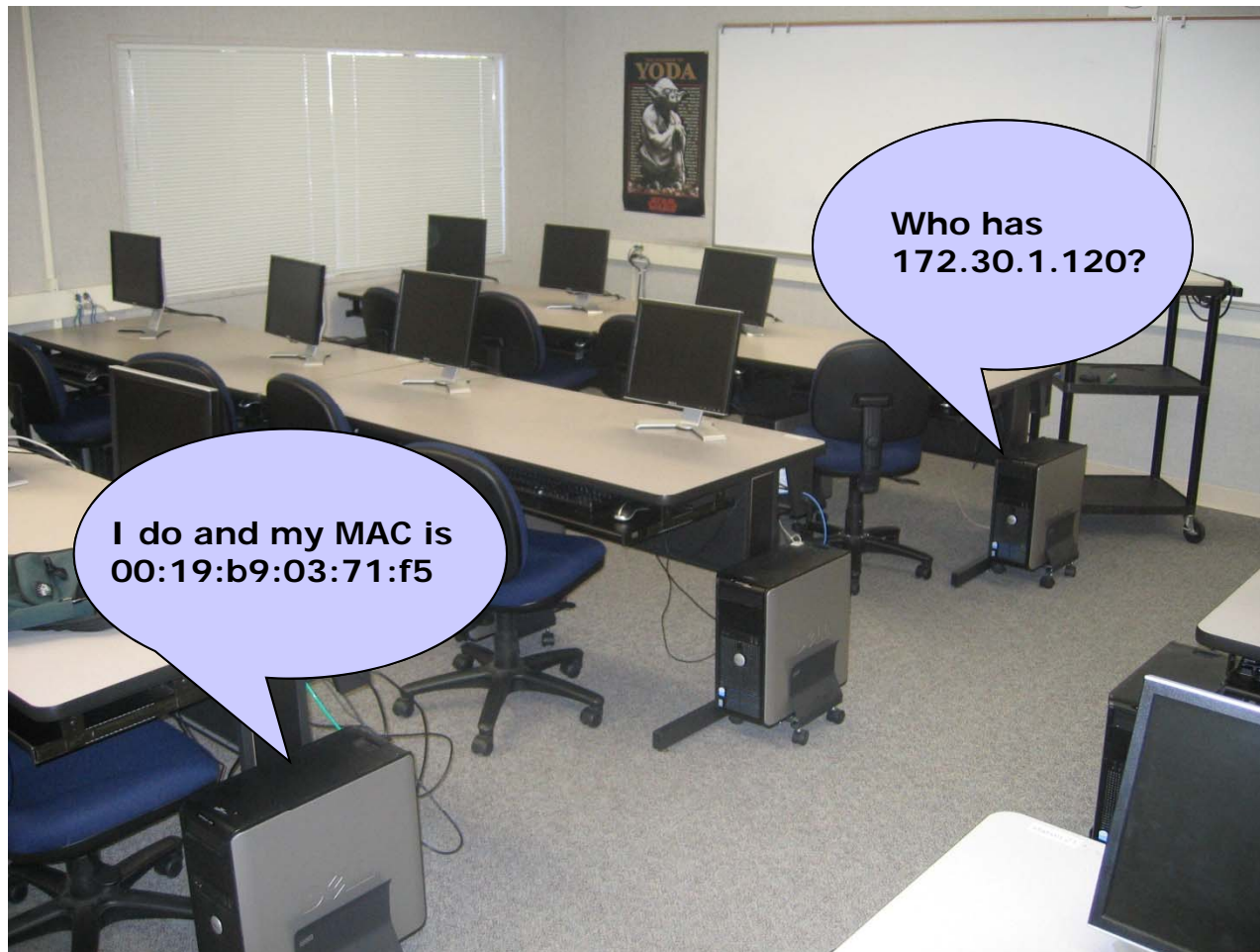
The purpose of ARP is to provide the correct destination physical address given the destination IP address.

- RFC 826 (<http://tools.ietf.org/html/rfc826>)
- Part of IPv4 (IPv6 uses Neighbor Discovery ND)
- The ARP request: broadcasts a "Who has this IP address?"
- The ARP replay: targeted to the requestor's address (unicast) – "I do and my MAC address is *xx:xx:xx:xx:xx:xx*"

ARP – Address Resolution Protocol

Overview

Station04 wants to ping Station20



ARP Example (1 of 3)

Address resolution commands

From Frodo, ping 172.30.1.109 which results in a ARP request and new arp cache entry

```
root@frodo:~# ifconfig eth0
```

```
eth0      Link encap:Ethernet  HWaddr 00:0c:29:98:c4:1d
          inet addr:172.30.1.150 Bcast:172.30.1.255 Mask:255.255.255.0
< snipped >
```

```
root@frodo:~# arp -n
```

Address	HWtype	HWaddress	Flags	Mask	Iface
172.30.1.106	ether	00:18:8b:28:ac:90	C		eth0
172.30.1.101	ether	00:19:b9:03:71:07	C		eth0
172.30.1.1	ether	00:b0:64:53:42:01	C		eth0

```
root@frodo:~# ping -c 1 172.30.1.109
```

```
PING 172.30.1.109 (172.30.1.109) 56(84) bytes of data.
64 bytes from 172.30.1.109: icmp_seq=1 ttl=128 time=3.71 ms
< snipped >
```

```
root@frodo:~# arp -n
```

Address	HWtype	HWaddress	Flags	Mask	Iface
172.30.1.109	ether	00:19:b9:03:70:d4	C		eth0
172.30.1.106	ether	00:18:8b:28:ac:90	C		eth0
172.30.1.101	ether	00:19:b9:03:71:07	C		eth0
172.30.1.1	ether	00:b0:64:53:42:01	C		eth0

ARP Example (2 of 3)

Address resolution example - request

Before Frodo can ping 172.30.1.109, it needs the destination MAC address to finish making the Ethernet frame for the ping request.

The screenshot shows a Wireshark capture of network traffic. The filter is set to `(arp || icmp) && eth.addr contains c4:1d`. The packet list shows four packets:

No.	Time	Source	Destination	Protocol	Info
204	42.970581	vmware_98:c4:1d	Broadcast	ARP	who has 172.30.1.109? Tell 172.30.1.150
205	42.970721	Dell_03:70:d4	vmware_98:c4:1d	ARP	172.30.1.109 is at 00:19:b9:03:70:d4
206	42.970820	172.30.1.150	172.30.1.109	ICMP	Echo (ping) request
207	42.970964	172.30.1.109	172.30.1.150	ICMP	Echo (ping) reply

The details pane for Frame 204 shows the following information:

- Ethernet II, Src: vmware_98:c4:1d (00:0c:29:98:c4:1d), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
- Destination: Broadcast (ff:ff:ff:ff:ff:ff)
- Source: vmware_98:c4:1d (00:0c:29:98:c4:1d)
- Type: ARP (0x0806)
- Address Resolution Protocol (request)
 - Hardware type: Ethernet (0x0001)
 - Protocol type: IP (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - opcode: request (0x0001)
 - Sender MAC address: vmware_98:c4:1d (00:0c:29:98:c4:1d)
 - Sender IP address: 172.30.1.150 (172.30.1.150)
 - Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
 - Target IP address: 172.30.1.109 (172.30.1.109)

A callout box with the text "Who has 172.30.1.109?" points to the ARP request details. The hex dump at the bottom shows the raw bytes of the frame.

Frodo's ARP request is a broadcast. Every NIC on the subnet will hear it and check to see if the requested IP address belongs to them.

ARP Example (3 of 3)

Address resolution example - reply

Filter: (arp || icmp) && eth.addr contains c4:1d

No.	Time	Source	Destination	Protocol	Info
204	42.970581	Vmware_98:c4:1d	Broadcast	ARP	who has 172.30.1.109? Tell 172.30.1.150
205	42.970721	Dell_03:70:d4	Vmware_98:c4:1d	ARP	172.30.1.109 is at 00:19:b9:03:70:d4
206	42.970820	172.30.1.150	172.30.1.109	ICMP	Echo (ping) request
207	42.970964	172.30.1.109	172.30.1.150	ICMP	Echo (ping) reply

Frame 205 (60 bytes on wire, 60 bytes captured)

- Ethernet II, Src: Dell_03:70:d4 (00:19:b9:03:70:d4), Dst: Vmware_98:c4:1d (00:0c:29:98:c4:1d)
 - Destination: Vmware_98:c4:1d (00:0c:29:98:c4:1d)
 - Source: Dell_03:70:d4 (00:19:b9:03:70:d4)
 - Type: ARP (0x0806)
 - Trailer: 00000000000000000000000000000000
- Address Resolution Protocol (reply)
 - Hardware type: Ethernet (0x0001)
 - Protocol type: IP (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - opcode: reply (0x0002)
 - Sender MAC address: Dell_03:70:d4 (00:19:b9:03:70:d4)
 - Sender IP address: 172.30.1.109 (172.30.1.109)
 - Target MAC address: Vmware_98:c4:1d (00:0c:29:98:c4:1d)
 - Target IP address: 172.30.1.150 (172.30.1.150)

```

0000  00 0c 29 98 c4 1d 00 19 b9 03 70 d4 08 06 00 01  ..).....p....
0010  08 00 06 04 00 02 00 19 b9 03 70 d4 ac 1e 01 6d  .....p....m
0020  00 0c 29 98 c4 1d ac 1e 01 96 00 00 00 00 00 00  ..).....
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
    
```

File: "C:\DOCLIME~1\CIS90~1\LOCALS~1\Temp... Packets: 257 Displayed: 6 Marked: 0 Dropped: 0 Profile: Default

Station09's ARP reply sent as a unicast directly back to Frodo.

Now Frodo knows the destination MAC address and can start pinging

ARP – Address Resolution Protocol

Overview

Managing the ARP cache

- Complete (C) 0x02

Temporary arp cached entries are aged out after several minutes

- Permanent (M) 0x04

Till next system restart

- Published (P) 0x08

The system will act as a ARP server and respond to ARP requests for IP addresses that are not its own

ARP – Address Resolution Protocol

Overview

```

root@frodo:~# ping 172.30.1.222
PING 172.30.1.222 (172.30.1.222) 56(84) bytes of data.
From 172.30.1.151 icmp_seq=2 Destination Host Unreachable
From 172.30.1.151 icmp_seq=3 Destination Host Unreachable
From 172.30.1.151 icmp_seq=4 Destination Host Unreachable
^C
--- 172.30.1.222 ping statistics ---
6 packets transmitted, 0 received, +3 errors, 100% packet loss, time 5006ms
, pipe 3
root@frodo:~# arp -n
Address                HWtype  HWaddress          Flags Mask          Iface
172.30.1.222           (incomplete)
172.30.1.125           ether    08:00:27:a6:20:b6   C                    eth0
root@frodo:~#

```

Pinging a non-existing system or a system that is powered off results in a failed arp request and an incomplete arp cache entry

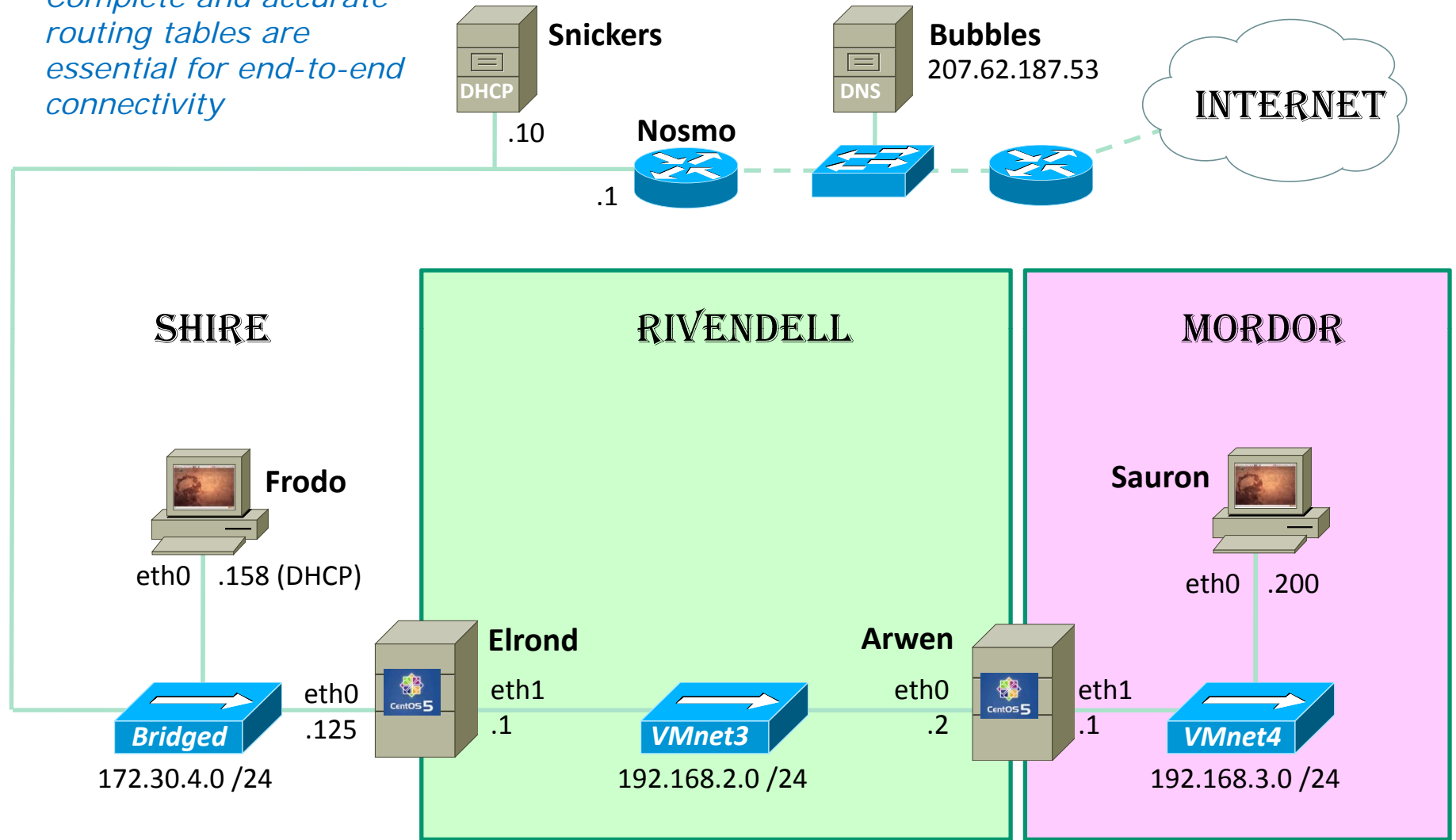
Class Exercise



1. What is the MAC address on the Nosmo router (a Cisco 2621) at 172.30.1.1?
2. Who wrote RFC 826?

Routing

Complete and accurate routing tables are essential for end-to-end connectivity



Frodo
 Default GW > 172.30.4.1
 192.168.2.0/24 > 172.30.4.125
 192.168.3.0/24 > 172.30.4.125

Elrond
 Default GW > 172.30.4.1
 192.168.3.0/24 > 192.168.2.2
 forwarding on

Arwen
 Default GW > 192.168.2.1
 forwarding on

Sauron
 Default GW > 192.168.3.1

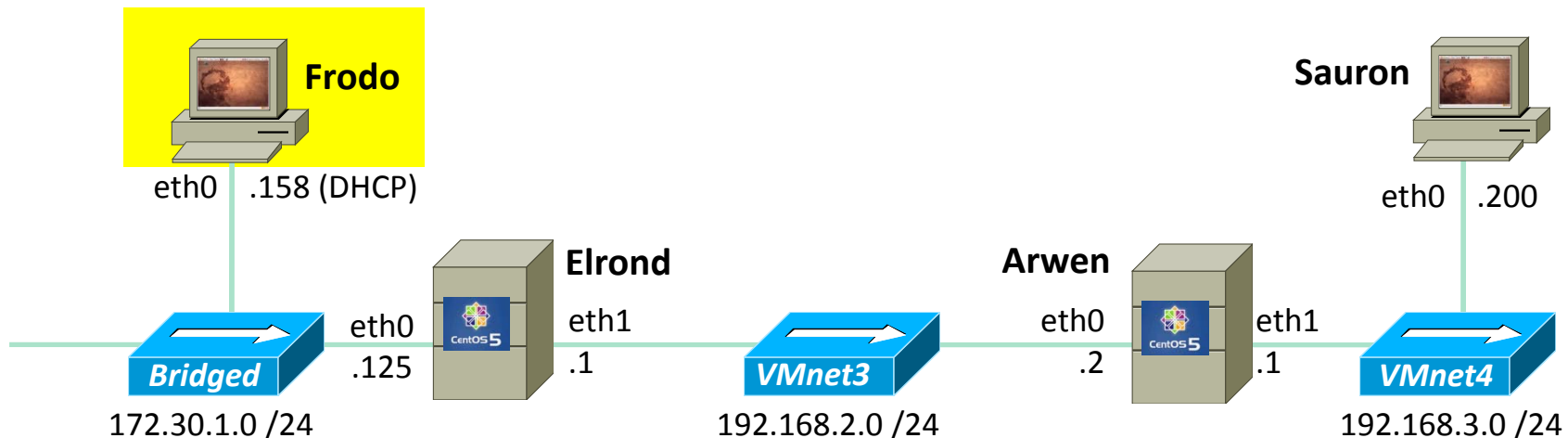
The Routing Table

This is the routing table on Frodo for Lab 3

```
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:~#
```



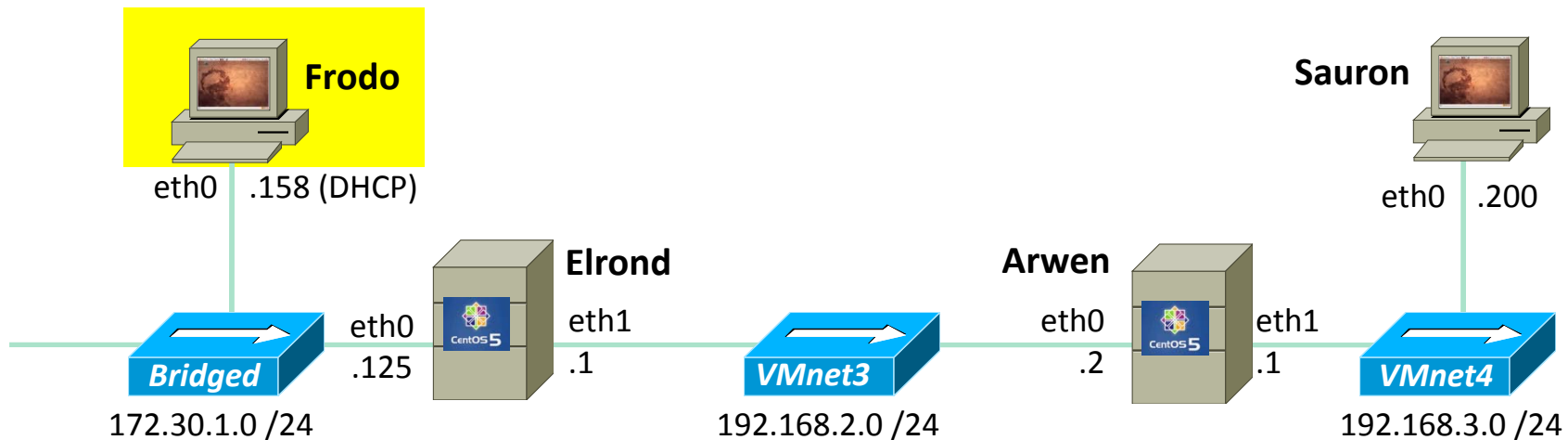
The Routing Table

Directly connected networks

```
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	U	1000	0	0	eth0	
0.0.0.0	172.30.1.1	0.0.0.0	UG	100	0	0	eth0

```
root@frodo:~#
```



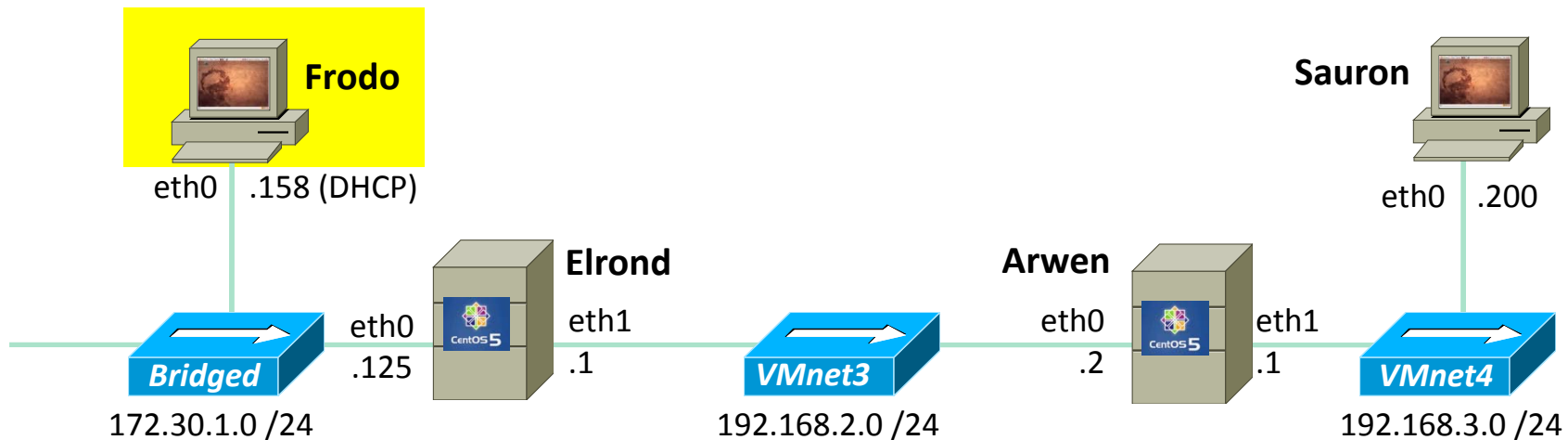
The Routing Table

Non-directly connected networks that are reachable via gateways (routers)

```
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:~#
```



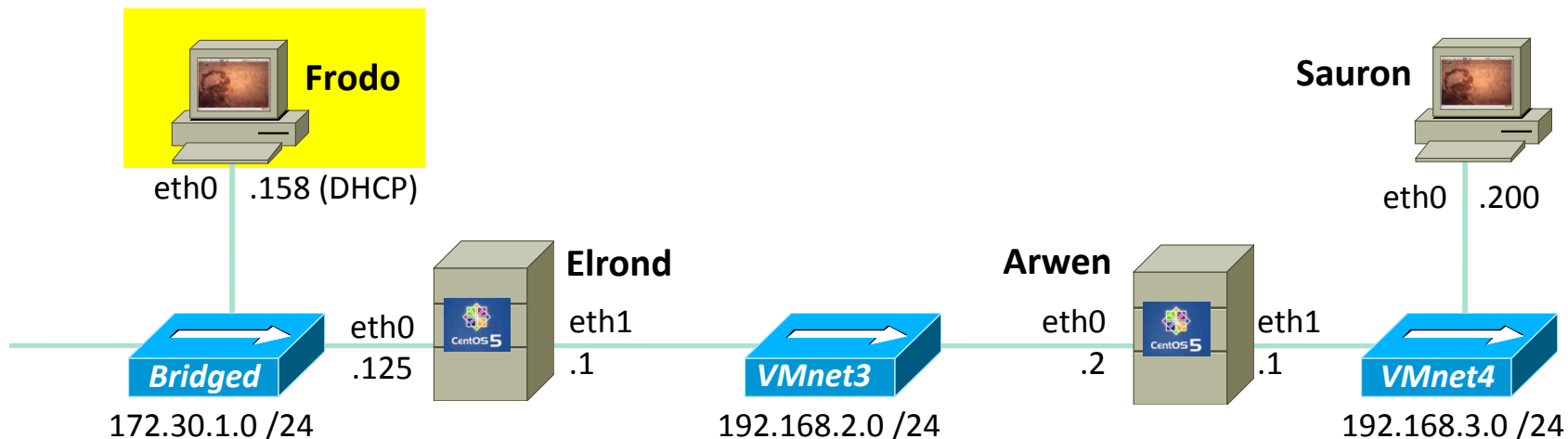
The Routing Table

The default gateway matches all networks

```
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:~#
```



The Routing Table

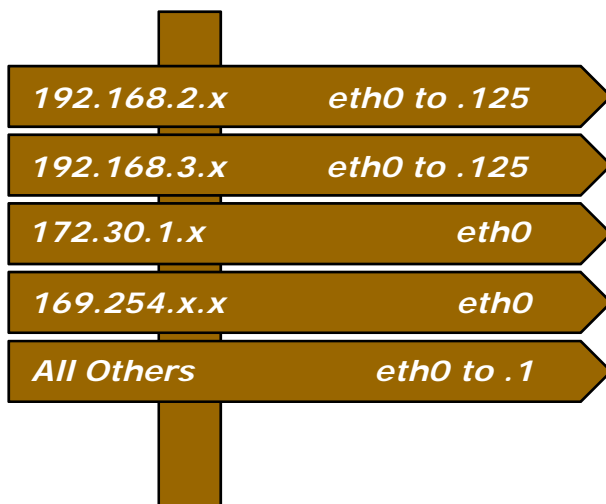
The routing table is a sign post

```
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:~#
```



In this case, all packets are forwarded out eth0. Each IP packet is placed in an Ethernet frame. The destination MAC address for the frame will be either the MAC address of the destination device if it is on a connected network, or the MAC address of the appropriate gateway.

The Routing Table

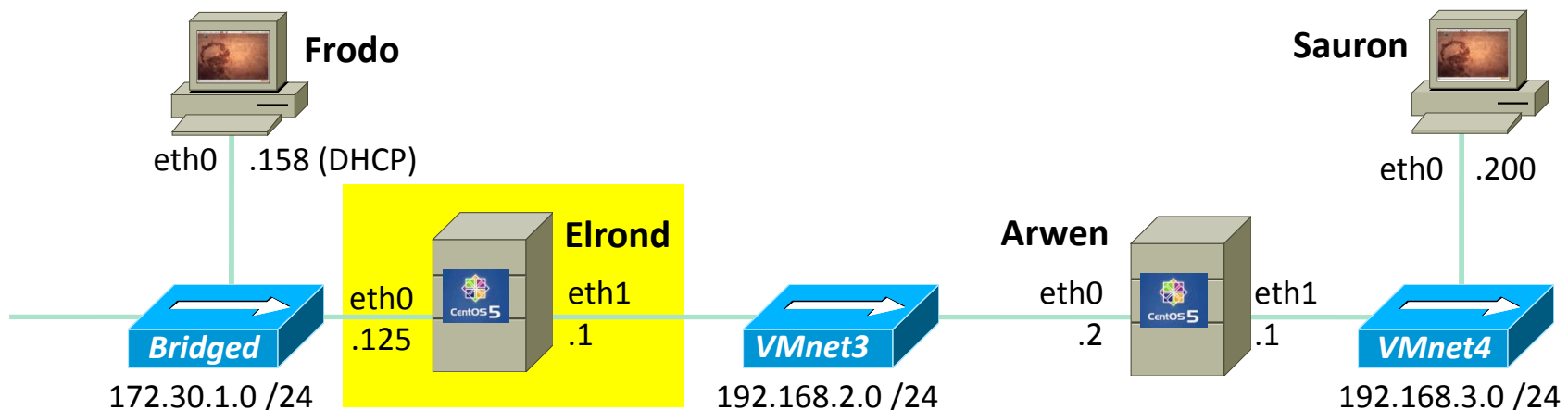
This is the routing table on Elrond for Lab 3

```
[root@elrond bin]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
172.30.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.3.0	192.168.2.2	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
0.0.0.0	172.30.1.1	0.0.0.0	UG	0	0	0	eth0

```
[root@elrond bin]#
```



The Routing Table

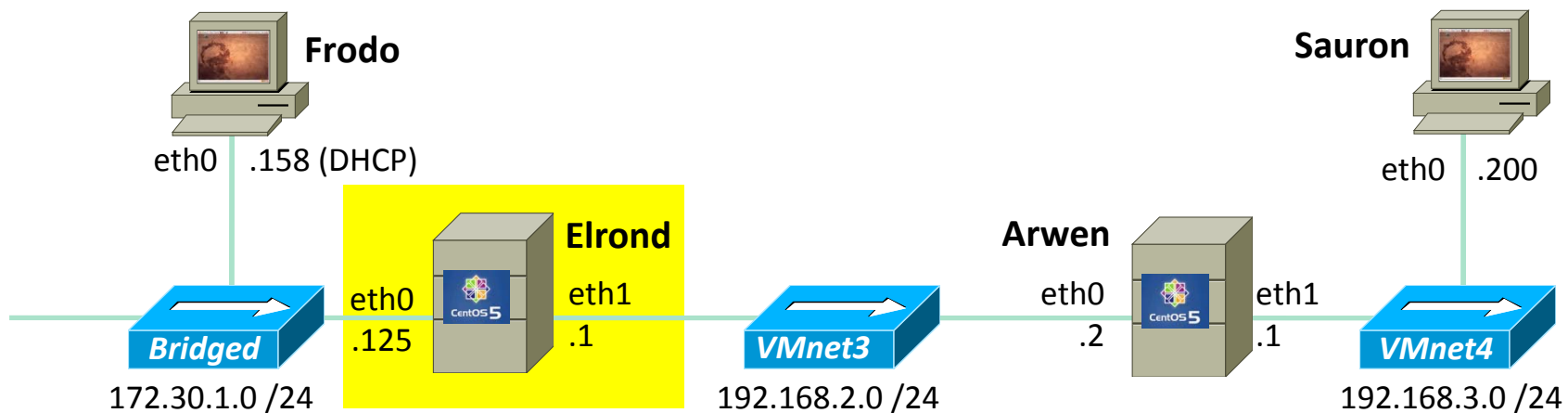
Directly connected networks

```
[root@elrond bin]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
172.30.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.3.0	192.168.2.2	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
0.0.0.0	172.30.1.1	0.0.0.0	UG	0	0	0	eth0

```
[root@elrond bin]#
```



The Routing Table

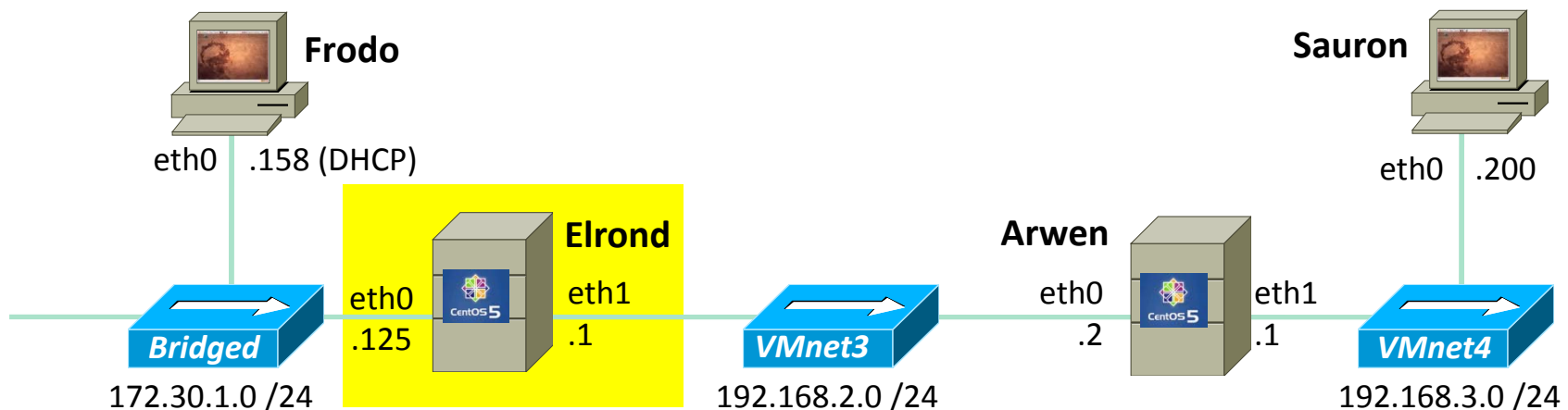
Non-directly connected networks that are reachable via gateways (routers)

```
[root@elrond bin]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
172.30.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.3.0	192.168.2.2	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
0.0.0.0	172.30.1.1	0.0.0.0	UG	0	0	0	eth0

```
[root@elrond bin]#
```



The Routing Table

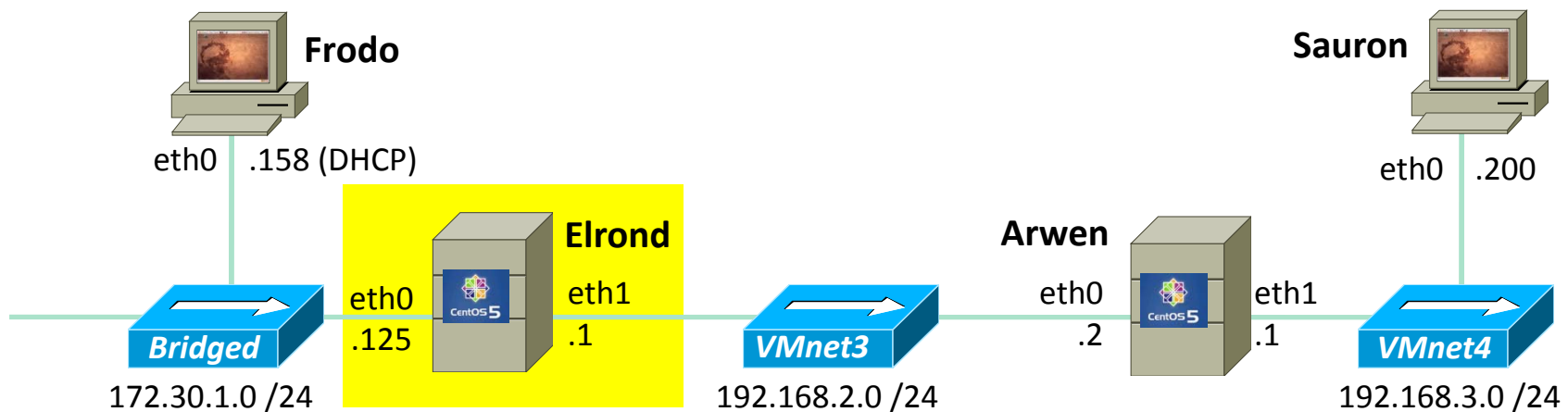
The default gateway matches all networks

```
[root@elrond bin]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
172.30.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.3.0	192.168.2.2	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
0.0.0.0	172.30.1.1	0.0.0.0	UG	0	0	0	eth0

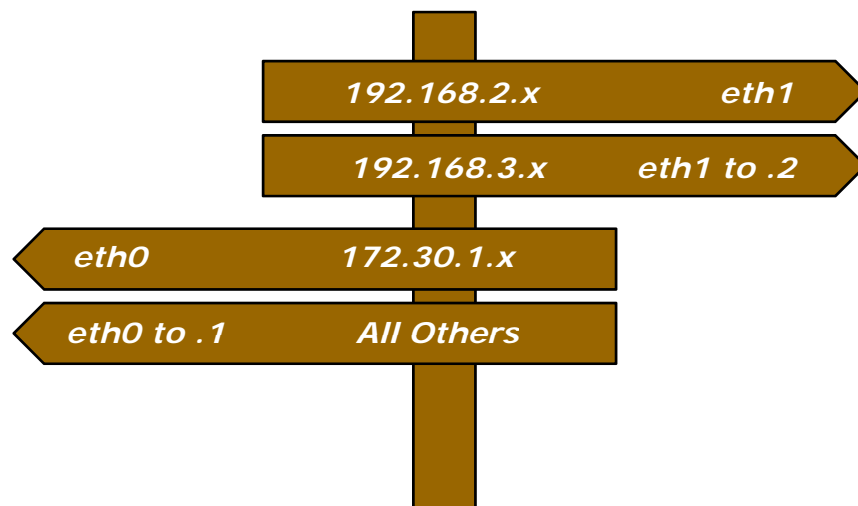
```
[root@elrond bin]#
```



The Routing Table

The routing table is a sign post

```
[root@elrond bin]# route -n
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
172.30.1.0       0.0.0.0         255.255.255.0  U         0      0      0 eth0
192.168.3.0     192.168.2.2    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
0.0.0.0         172.30.1.1     0.0.0.0        UG        0      0      0 eth0
[root@elrond bin]#
```



In this case, all packets are forwarded out either eth0 or eth1. Each IP packet is placed in an Ethernet frame. The destination MAC address for the frame will be either the MAC address of the destination device if it is on a connected network, or the MAC address of the appropriate gateway.

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 **192.168.0.0 /16**. This is super-netting (the reverse of sub-netting)*

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.

- Compute the destination network number of the destination IP address.
- Does the destination network match that on a local interface?
If so, send it out that interface
- Does the destination network match one or more listed in the routing table?
If so, send it using the best match (largest genmask) route
- Is there a default route listed in the routing table?
If so, use that gateway
Otherwise, drop the packet - "network is unreachable"

route command

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 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



```
[root@elrond bin]# route -n
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
172.30.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.3.0	192.168.2.2	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
0.0.0.0	172.30.1.1	0.0.0.0	UG	0	0	0	eth0

```
[root@elrond bin]#
```

1. What happens with an IP packet having a destination address of 207.62.187.53?
2. What happens with an IP packet having a destination address of 192.168.2.15?
3. What happens with an IP packet having a destination address of 172.30.4.1?
4. What networks are directly connected to this Linux router?
5. Are gateway addresses on directly connected networks, non-directly connected networks or both?
6. When creating an Ethernet frame for an IP packet with destination address 192.168.3.99, what destination MAC address is used?
7. What is the default gateway?



```
[root@arwen ~]# 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
172.30.1.0	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	0	0	0	eth1

```
[root@arwen ~]#
```

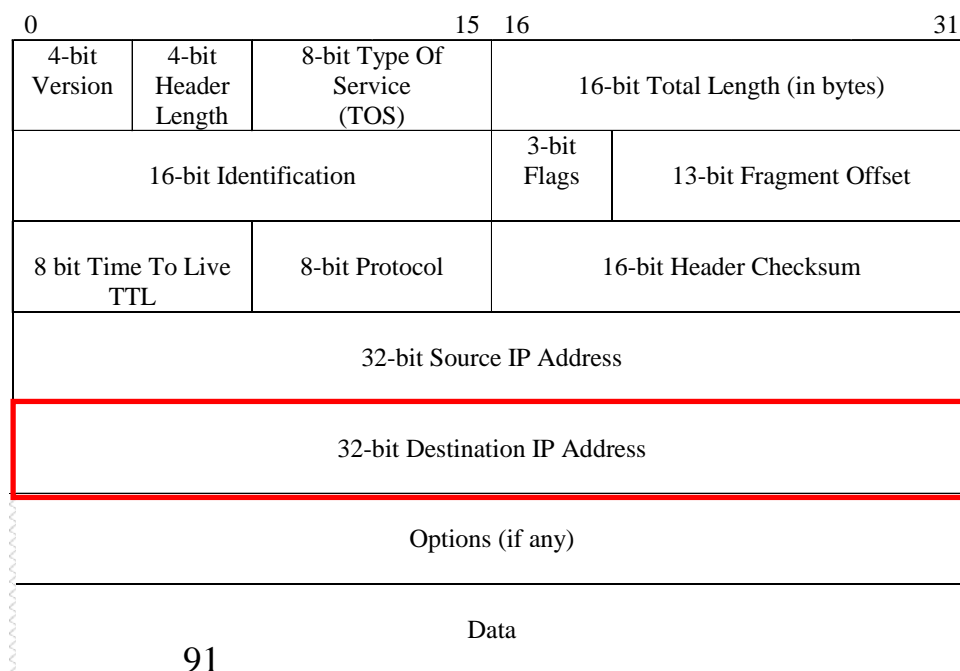
1. What does the "G" mean under flags?
2. What is the default gateway?
3. What happens with a packet with destination IP address 207.62.186.9?
4. What networks are not directly connected?
5. In what order are potential routes tested for a match?
6. When creating an Ethernet frame for an IP packet with destination address 10.10.10.200, what destination MAC address is used?
7. When configuring a static route , should the gateway be on a connected or non-connected network?

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.

IP Header



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 to implement in the Cabrillo routing and advanced routing classes.

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

Routing Types

- **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.

*A router must learn about non-directly connected networks either **statically** or **dynamically**.*

Static

Uses a programmed route that a network administrator enters into the router

Dynamic

Uses a route that a routing protocol adjusts automatically for topology or traffic changes



BGP
ARP
OSPF
RIP
IPv4
IPv6
Ethernet
DHCP

1. Which of the protocols above are routable?
2. Which are routing protocols?
3. Which are neither routing protocols or routable?

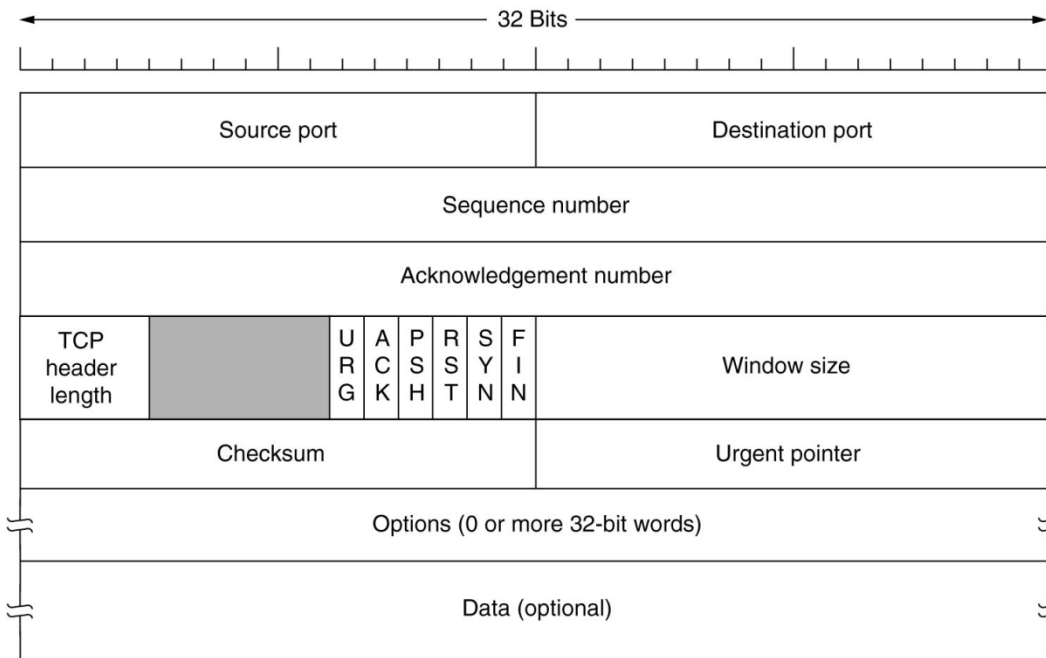
End of
Review

Transmission Control Protocol

Transport Layer

The Transmission Control Protocol

TCP Header



Sequence and acknowledgement numbers are used for flow control.

ACK, SYN and FIN flags are used for initiating connections, acknowledging data received and terminating connections

Window size is use to communicate buffer size of recipient.

Options like SACK permit selective acknowledgement

Transport Layer

The Transmission Control Protocol

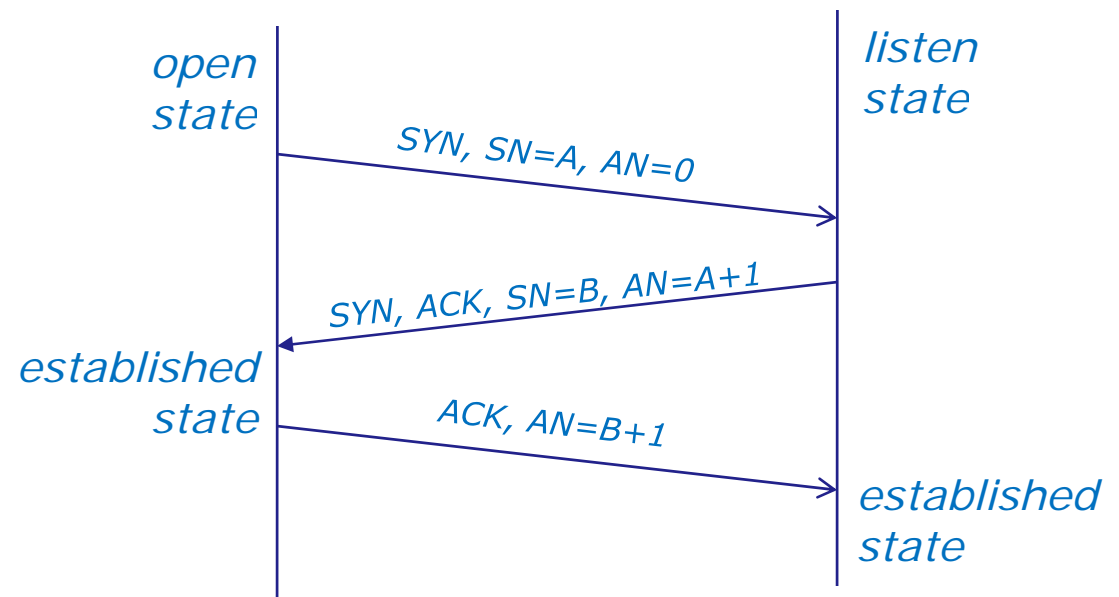
Initial Connection

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

Continuing Communications

- o The Sliding Window

Used for flow control - allows sending additional segments before an acknowledgement is received based on recipients buffer size

- o Flow Control (cumulative acknowledgment)

Recipient tells sender the size of its input buffer and sends acknowledgements when data has been received. Sequence numbers are used to detect missing segments.

- o The SACK option

Selective acknowledgement so only the dropped segments need to be retransmitted.

- o The RST Flag

Used to terminate a connection when an abnormal situation happens

Transport Layer

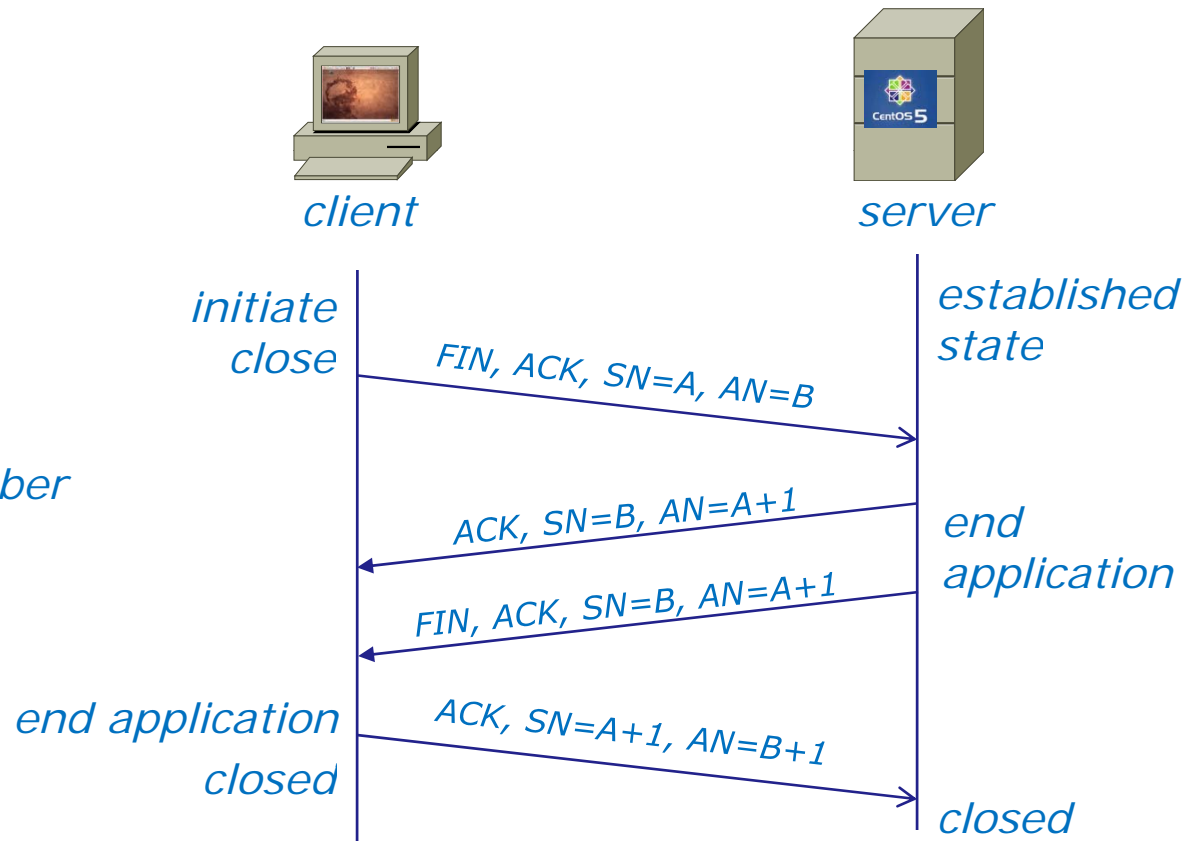
The Transmission Control Protocol

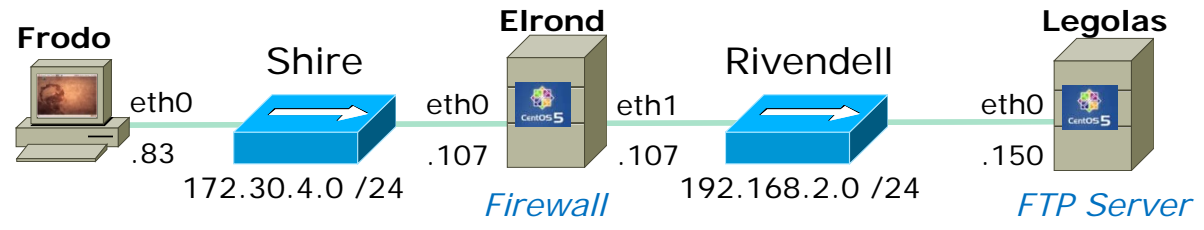
Closing a Connection

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

```
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)
```

Socket for data transfer

Client	Server
172.30.4.83	192.168.2.150
42571	20

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 PASV
172.30.4.83	42855	192.168.2.150	21	FTP	Request: RETR legolas <i>Retrieve legolas file</i>
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [SYN] Seq=0 Win=0 Len=0
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [SYN, ACK] Seq=1 Win=0 Len=0
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=1 Ack=2 Win=5888 Len=0
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 <i>File transfer</i>
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] Seq=20 Ack=20 Win=0 Len=0
172.30.4.83	42571	192.168.2.150	20	TCP	42571 > ftp-data [FIN, ACK] Seq=20 Ack=20 Win=0 Len=0
192.168.2.150	20	172.30.4.83	42571	TCP	ftp-data > 42571 [ACK] Seq=20 Ack=20 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

Tunable Kernel Parameters

Transport Layer

TCP Tunable Kernel Parameters

<code>tcp_fin_timeout</code>	<i>how long to keep in FIN-WAIT-2 state</i>
<code>tcp_keepalive_time</code>	<i>how long to keep an unused connection alive</i>
<code>tcp_sack</code>	<i>enable/disable selective acknowledgments</i>
<code>tcp_timestamps</code>	<i>enable RFC 1323 definition for round-trip measurement</i>
<code>tcp_window_scaling</code>	<i>enable RFC 1323 window scaling</i>
<code>tcp_retries1</code>	<i>how many times to retry before reporting an error</i>
<code>tcp_retries2</code>	<i>how many times to retry before killing connection</i>
<code>tcp_syn_retries</code>	<i>how many times to retransmit the SYN, ACK reply</i>

In the same directory:

<code>ip_forward</code>	<i>enable/disable selective acknowledgments</i>
-------------------------	---

Exercise

Explore the TCP, UDP and other variables in the `/proc/sys/net/ipv4` directory

```
[root@bigserver ~]# ls /proc/sys/net/ipv4
cipso_cache_bucket_size      ip_dynaddr                  tcp_dsack                   tcp_retries2
cipso_cache_enable          ip_forward                  tcp_ecn                     tcp_rfc1337
cipso_rbm_optfmt            ipfrag_high_thresh         tcp_fack                     tcp_rmem
cipso_rbm_strictvalid        ipfrag_low_thresh          tcp_fin_timeout             tcp_sack
conf                         ipfrag_max_dist            tcp_frto                     tcp_slow_start_after_idle
icmp_echo_ignore_all         ipfrag_secret_interval     tcp_keepalive_intvl         tcp_stdurg
icmp_echo_ignore_broadcasts  ipfrag_time                 tcp_keepalive_probes        tcp_synack_retries
icmp_errors_use_inbound_ifaddr ip_local_port_range         tcp_keepalive_time          tcp_syncookies
icmp_ignore_bogus_error_responses ip_nonlocal_bind            tcp_low_latency              tcp_syn_retries
icmp_ratelimit               ip_no_pmtu_disc             tcp_max_orphans              tcp_timestamps
icmp_ratemask                 neigh                        tcp_max_syn_backlog          tcp_tso_win_divisor
igmp_max_memberships         netfilter                   tcp_max_tw_buckets          tcp_tw_recycle
igmp_max_msf                 route                       tcp_mem                      tcp_tw_reuse
inet_peer_gc_maxtime         tcp_abc                     tcp_moderate_rcvbuf          tcp_window_scaling
inet_peer_gc_mintime         tcp_abort_on_overflow       tcp_mtu_probing              tcp_wmem
inet_peer_maxttl             tcp_adv_win_scale           tcp_no_metrics_save
tcp_workaround_signed_windows
inet_peer_minttl             tcp_app_win                 tcp_orphan_retries          udp_mem
inet_peer_threshold          tcp_base_mss                 tcp_reordering               udp_rmem_min
ip_contrack_max              tcp_congestion_control       tcp_retrans_collapse         udp_wmem_min
ip_default_ttl               tcp_dma_copybreak           tcp_retries1
```

```
[root@bigserver ~]# cat /proc/sys/net/ipv4/tcp_sack
1
[root@bigserver ~]# cat /proc/sys/net/ipv4/tcp_syn_retries
5
[root@bigserver ~]#
```

Exercise

Google linux tcp variables

The screenshot shows two overlapping browser windows. The background window is a Google search results page for 'linux tcp variables'. The foreground window is a Mozilla Firefox browser displaying a tutorial page from 'Ipsysctl tutorial 1.0.4', specifically 'Chapter 3. IPv4 variable reference'. The tutorial page contains the following content:

3.3. TCP Variables

This section will take a brief look at the variables that changes the behaviour of the TCP variables. These variables are normally set to a pretty good value per default and most of them should never ever be touched, except when asked by authoritative developers! They are mainly described here, only for those who are curious about their basic meaning.

3.3.1. tcp_abort_on_overflow

The tcp_abort_on_overflow variable tells the kernel to reset new connections if the system is currently overflowed with new connection attempts that the daemon(s) can not handle. What this means, is that if the system is overflowed with 1000 large requests in a burst, connections may be reset since we can not handle them if this variable is turned on. If it is not set, the system will try to recover and handle all requests.

This variable takes an boolean value (ie, 1 or 0) and is per default set to 0 or FALSE. Avoid enabling this option except as a last resort since it most definitely harm your clients. Before considering using this variable you should try to tune up your daemons to accept connections faster.

3.3.2. tcp_adv_win_scale

This variable is used to tell the kernel how much of the socket buffer space should be used for TCP window size, and how much to save for an application buffer. If tcp_adv_win_scale is negative, the following equation is used to calculate the buffer overhead for window scaling:

$$\text{bytes} = \frac{\text{bytes}}{2^{(-\text{tcp_adv_win_scale})}}$$

At the bottom of the browser window, there is a search bar with the text 'Find:' and a 'Done' button. The 'Match case' checkbox is checked.

Security Issues

Transport Layer

Security Issues

Resource: *www.securityfocus.org*

- SYN Flooding

" ... Bombarding a system with, say, dozens of falsified connection requests a minute can seriously degrade its ability to give service to legitimate connection requests. This is why the attack is said to "deny service" to the system's users. ..."

Source: <http://www.securityfocus.com/advisories/141>

- Falsifying TCP Communications

"... In IP spoofing, an attacker gains unauthorized access to a computer or a network by making it appear that a malicious message has come from a trusted machine by "spoofing" the IP address of that machine. ..."

Source: <http://www.securityfocus.com/infocus/1674>

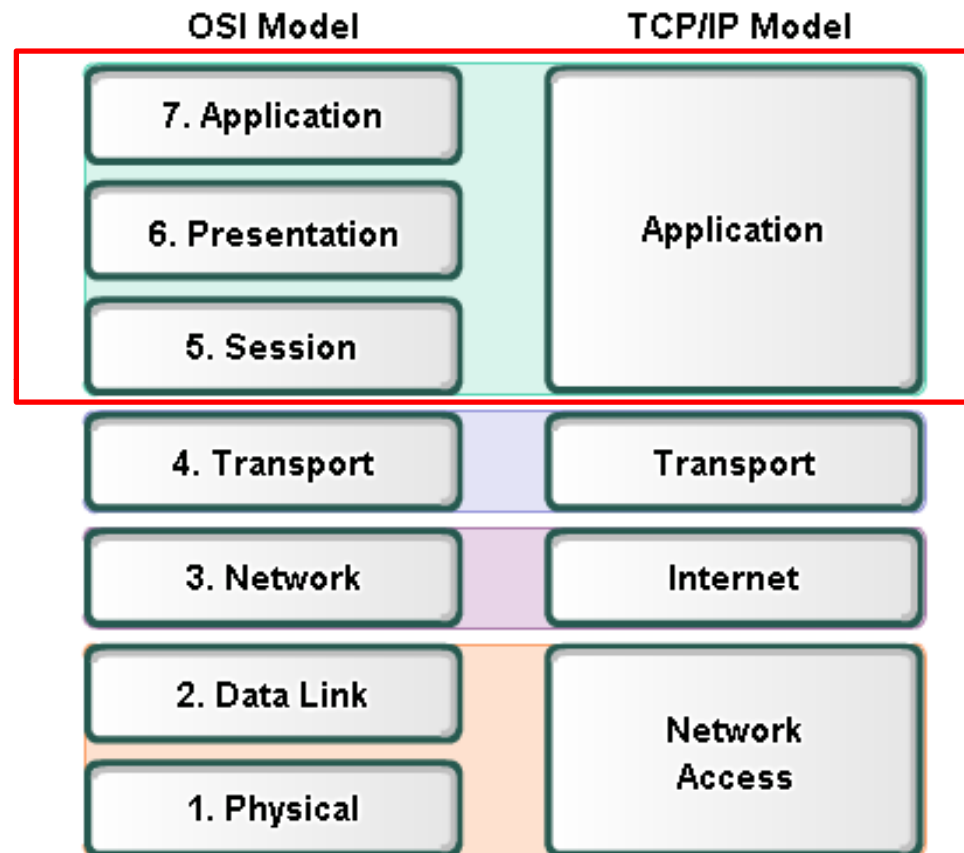
- Hijacking connections

"... Another consequence, specific to TCP, is sequence number prediction, which can lead to session hijacking or host impersonating. This method builds on IP spoofing, since a session, albeit a false one, is built. ..."

source: <http://www.securityfocus.com/infocus/1674>

Application Layer

Protocol and Reference Models



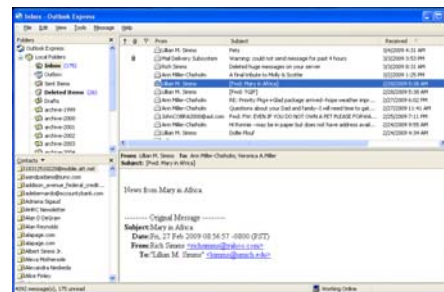
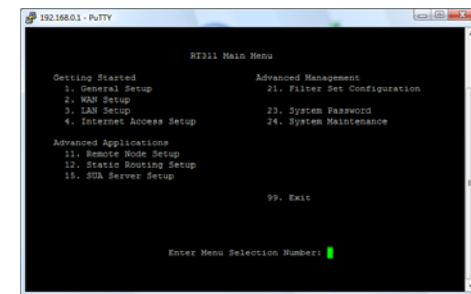
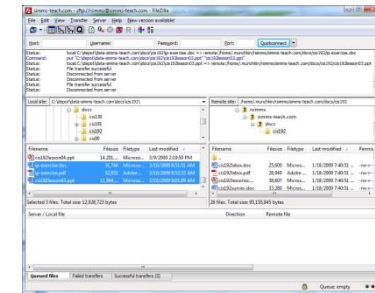
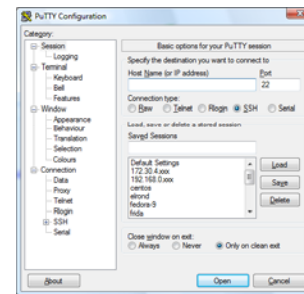
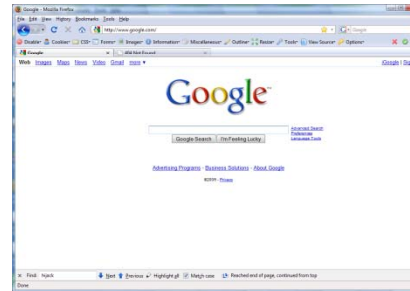
- The **Open Systems Interconnection (OSI)** model is the *most widely known internetwork reference model*.

Application Layer

Applications

Examples:

- Web servers
- FTP servers
- SSH daemon
- Telnet server
- email



Application Layer

Responsibilities of Applications

Network connections, routing, and transfer of data are all taken care of by the lower layers of the protocol stack. What must applications do?

- Authenticate users
- Control access
- Log important information
- Format data (compress/encrypt)
- Provide whatever functionality is desired.

Service Ports

Last week we talked about Layer 4 ports . Ports are used to direct requests to the appropriate service/application

< 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

```
lmtpl       24/tcp          # LMTP Mail Delivery
lmtpl       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 >

Application Layer

The Client-Server Model

Clients

Programs that are generally run on demand, and initiate the network connection to the server.

Examples: telnet, ftp, ssh, browsers, email clients.

Servers

Programs (services/daemons) that are constantly running in the background waiting for client connections.

- Services and Ports: */etc/services*
- Architecture:
 - Direct or iterative servers – listens to a particular port and directly responds to requests
 - Indirect or concurrent servers (e.g. super daemons) – listens to a particular port and then starts up another server program to process the request

Application Layer

The Super Daemons

- There are three primary super-daemons controlling server services.
- Super daemons spawn other daemons to handle specific client requests.
 1. `inetd` - From early UNIX days, this was the primary daemon for handling tcp application services. It is being replaced by `xinetd`.
 2. `portmap` - portmapper operates with Remote Procedure Call (RCP) applications.
 3. `xinetd` - Extended Internet Services Daemon: used by modern distributions of Linux.

telnet
(via port 23)

Installing and Configuring Telnet (Red Hat Family)

Is it installed?

```
[root@bigserver ~]# rpm -qa | grep telnet  
telnet-0.17-39.e15  
telnet-server-0.17-39.e15  
[root@bigserver ~]#
```

client

server

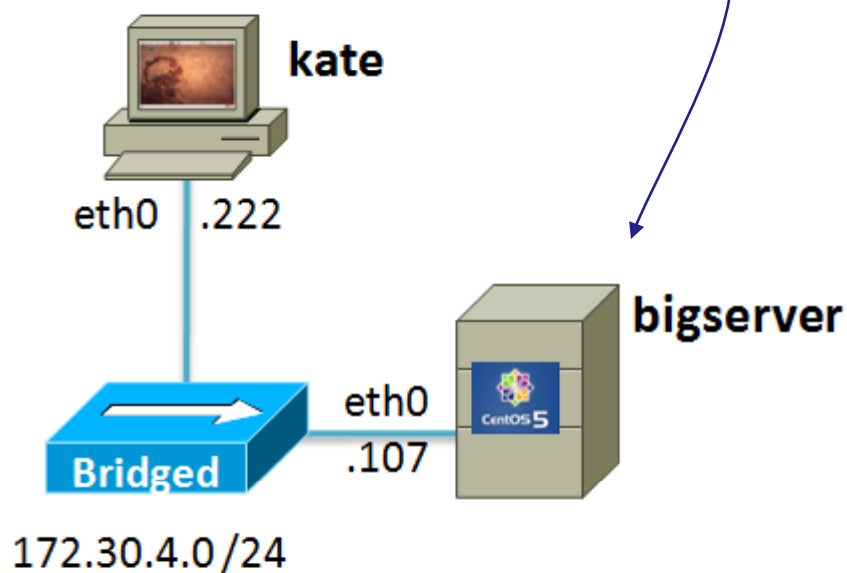
No response means it is not installed

*Use **dpkg -l | grep telnet** on the Debian family*

Installing and Configuring Telnet

Installing Telnet

yum install telnet-server



Installing and Configuring Telnet

```
[root@bigserver ~]# yum install telnet-server
Loading "fastestmirror" plugin
Loading mirror speeds from cached hostfile
 * base: centos.mirrors.redwire.net
 * updates: centos.mirrors.redwire.net
 * addons: centos.mirrors.redwire.net
 * extras: mirrors.usc.edu
Setting up Install Process
Parsing package install arguments
Resolving Dependencies
--> Running transaction check
---> Package telnet-server.i386 1:0.17-39.el5 set to be updated
--> Processing Dependency: xinetd for package: telnet-server
--> Running transaction check
---> Package xinetd.i386 2:2.3.14-10.el5 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

Note that the telnet server uses xinetd

Installing and Configuring Telnet

Dependencies Resolved

```
=====
Package                Arch      Version      Repository    Size
=====
Installing:
telnet-server          i386     1:0.17-39.el5  base          35 k
Installing for dependencies:
xinetd                 i386     2:2.3.14-10.el5 base          124 k
=====
```

Transaction Summary

```
=====
Install      2 Package(s)
Update      0 Package(s)
Remove      0 Package(s)
=====
```

Total download size: 159 k

Is this ok [y/N]: y

Downloading Packages:

```
(1/2): xinetd-2.3.14-10.e 100% |=====| 124 kB    00:00
(2/2): telnet-server-0.17 100% |=====|  35 kB    00:00
```

Running rpm_check_debug

Running Transaction Test

Finished Transaction Test

Transaction Test Succeeded

Running Transaction

```
Installing: xinetd                ##### [1/2]
```

```
Installing: telnet-server         ##### [2/2]
```

Installed: telnet-server.i386 1:0.17-39.el5

Dependency Installed: xinetd.i386 2:2.3.14-10.el5

Complete!

[root@bigserver ~]#

Installing and Configuring Telnet

Edit 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           Sets the TCP/IP socket to be reusable (after restarts)
    socket_type          = stream
    wait                 = no              Starts a daemon for each request
    user                  = root           Sets UID for daemon
    server                = /usr/sbin/in.telnetd
    log_on_failure        += USERID       Use HOST for IP address and RECORD for terminal type
    disable               = no            This enables Telnet service
}
```

Great reference is "LINUX TCP/IP Network Administration" by Scott Mann

Installing and Configuring Telnet

Note: there may be multiple telnetd daemons

```
[root@arwen ~]# ls /etc/xinetd.d/*tel*  
/etc/xinetd.d/ekrb5-telnet  /etc/xinetd.d/telnet  
/etc/xinetd.d/krb5-telnet  /etc/xinetd.d/telnet.rpmsave
```

```
[root@arwen ~]# cat /etc/xinetd.d/krb5-telnet  
# default: off  
# description: The kerberized telnet server accepts normal telnet sessions, \  
#               but can also use Kerberos 5 authentication.  
service telnet  
{  
    flags          = REUSE  
    socket_type    = stream  
    wait          = no  
    user          = root  
    server         = /usr/kerberos/sbin/telnetd  
    log_on_failure += USERID  
    disable       = yes  
}  
[root@arwen ~]#
```

Disable this telnet daemon

Installing and Configuring Telnet

Note: there may be multiple xinetd daemons

```
[root@arwen ~]# ls /etc/xinetd.d/*tel*
/etc/xinetd.d/ekrb5-telnet  /etc/xinetd.d/telnet
/etc/xinetd.d/krb5-telnet  /etc/xinetd.d/telnet.rpmsave
```

```
[root@arwen ~]# cat /etc/xinetd.d/ekrb5-telnet
# default: off
# description: The kerberized telnet server accepts only telnet sessions, \
#               which use Kerberos 5 authentication and encryption.
service telnet
{
    flags                = REUSE
    socket_type          = stream
    wait                 = no
    user                 = root
    server               = /usr/kerberos/sbin/telnetd
    server_args          = -e
    log_on_failure      += USERID
    disable              = yes
}
[root@arwen ~]#
```

Disable this telnet daemon

Installing and Configuring Telnet (Red Hat Family)

Start or restart service

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

Or have xinetd reread it's configuration file

```
[root@arwen ~]# killall -1 xinetd
```

Automatically start at system boot

```
[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  
  eklogin:        off  
  ekrb5-telnet:   off  
  gssftp:         off  
  klogin:         off  
  krb5-telnet:   off  
  kshell:         off  
  rsync:          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

Multiple telnet daemons



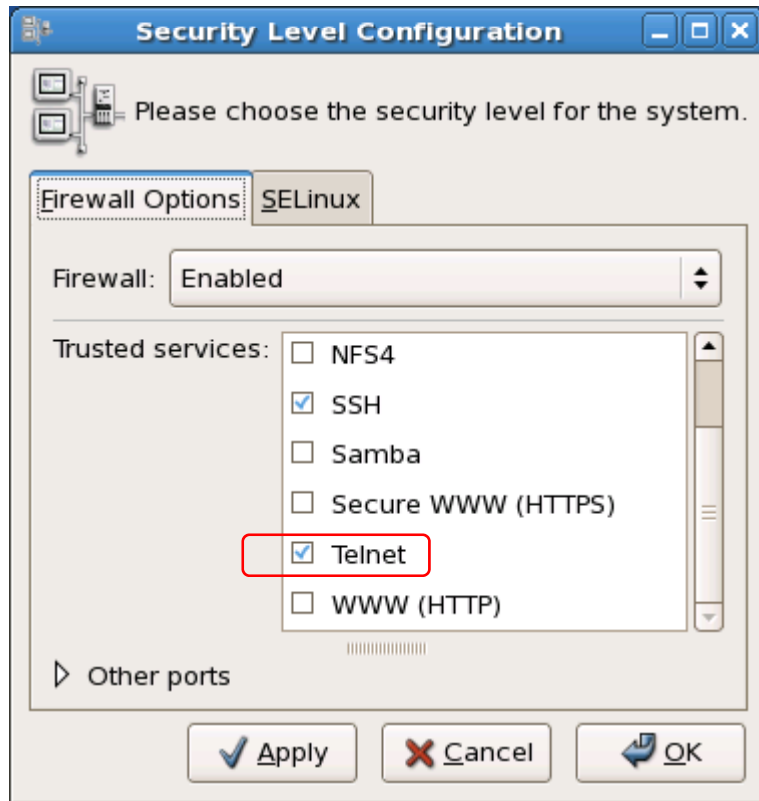
Firewall for Telnet

CentOS Default

```
[root@bigserver ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -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 ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@bigserver ~]#
```

*Only the SSH port (22) is open
for new incoming connections*

Firewall for Telnet (CentOS)



Open the Telnet port by placing check next to Telnet in the Trusted service list

*Later we will learn how to make these changes using the **iptables** command*

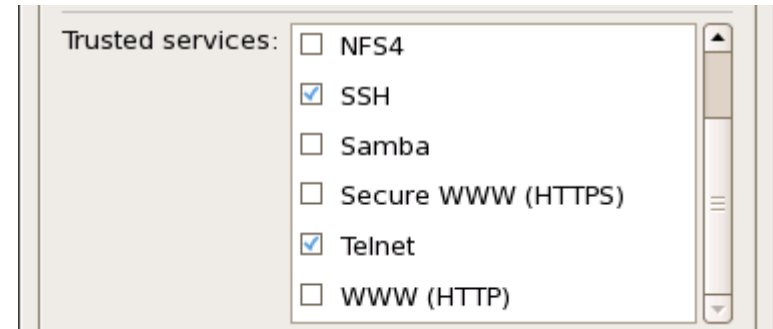
Firewall for Telnet

CentOS Modified

```

root@bigserver ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -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 ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 23 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@bigserver ~]#

```



Telnet and SSH port are open for incoming connections

Installing and Configuring Telnet

netstat

```
[root@bigserver ~]# netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 127.0.0.1:2208         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:111          0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:6000         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:21           0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:23           0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:631        0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:792          0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:25         0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:2207       0.0.0.0:*               LISTEN
tcp      0      0 :::6000              :::*                    LISTEN
tcp      0      0 :::22                :::*                    LISTEN
[root@bigserver ~]#
```

*Use **netstat -tln** command to see what port numbers your system is listening for requests on*

Installing and Configuring Telnet

netstat

```
[root@bigserver ~]# netstat -tl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 r1.localdomain:2208    *:*                    LISTEN
tcp      0      0 *:sunrpc               *:*                    LISTEN
tcp      0      0 *:x11                  *:*                    LISTEN
tcp      0      0 *:ftp                  *:*                    LISTEN
tcp      0      0 *:telnet               *:*                    LISTEN
tcp      0      0 r1.localdomain:ipp     *:*                    LISTEN
tcp      0      0 *:792                  *:*                    LISTEN
tcp      0      0 r1.localdomain:smtp    *:*                    LISTEN
tcp      0      0 r1.localdomain:2207    *:*                    LISTEN
tcp      0      0 *:x11                  *:*                    LISTEN
tcp      0      0 *:ssh                  *:*                    LISTEN
[root@bigserver ~]#
```

*Use **netstat -tl** command to see what port names your system is listening for requests on*

Installing and Configuring Telnet

telnetd processes

```
[cis192@bigserver ~]$ ps -ef | grep telnet
root      6156   6118   0 07:52 ?          00:00:00 in.telnetd: kate
root      6268   6118   0 07:53 ?          00:00:00 in.telnetd: 192.168.0.27
root      6299   6118   0 07:56 ?          00:00:00 in.telnetd: 192.168.0.23
cis192    6325   6270   0 07:56 pts/2    00:00:00 grep telnet
[cis192@bigserver ~]$
```

Individual telnetd daemons are run for each session

Installing and Configuring Telnet

Edit 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            = 192.168.0.23
    server               = /usr/sbin/in.telnetd
    log_on_failure       += USERID
    disable              = no
}
[root@arwen ~]#
```

Use `only_from` to restrict clients that can access the Telnet service

Installing and Configuring Telnet

Only_from examples

only_from = arwen *hostname*

only_from = arwen legolas *multiple hostnames*

only_from = 192.168.3.12 192.168.3.14 *or IP addresses*

only_from = 192.168.3.{12, 14} *same as above*

only_from = 192.168.0.0 *0's are wildcards*

only_from = sauron 172.30.4.0 10.10.10.{1, 200} *mixes*

Installing and Configuring Telnet

The screenshot shows a Wireshark capture of network traffic. The main pane displays a list of packets:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	Vmware_6f:53:d9	Broadcast	ARP	who has 172.30.4.107? Tell 172.30.4.222
2	0.000159	Vmware_12:50:1e	Vmware_6f:53:d9	ARP	172.30.4.107 is at 00:0c:29:12:50:1e
3	0.000199	172.30.4.222	172.30.4.107	TCP	52389 > telnet [SYN] Seq=0 Win=5840 Len=0 MSS=1460 WS=5
4	0.002030	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 WS=5
5	0.002537	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=1 Ack=1 Win=5856 Len=0
6	0.005580	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...

The packet details pane for Frame 4 (66 bytes on wire, 66 bytes captured) shows the following information:

- Ethernet II, Src: Vmware_12:50:1e (00:0c:29:12:50:1e), Dst: Vmware_6f:53:d9 (00:0c:29:6f:53:d9)
- Internet Protocol, Src: 172.30.4.107 (172.30.4.107), Dst: 172.30.4.222 (172.30.4.222)
- Transmission Control Protocol, Src Port: telnet (23), Dst Port: 52389 (52389), Seq: 0, Ack: 1, Len: 0
 - Source port: telnet (23)
 - Destination port: 52389 (52389)
 - Sequence number: 0 (relative sequence number)
 - Acknowledgement number: 1 (relative ack number)
 - Header length: 32 bytes
 - Flags: 0x12 (SYN, ACK)
 - Window size: 5840
 - Checksum: 0x121a [correct]
 - Options: (12 bytes)
 - [SEQ/ACK analysis]

} 3-way
handshake
that initiates
TCP
connection

Installing and Configuring Telnet

No.	Time	Source	Destination	Protocol	Info
1	0.000000	Vmware_6f:53:d	Broadcast	ARP	Who has 172.30.4.107? Tell 172.30.4.222
2	0.000159	Vmware_12:50:1	Vmware_6f:53:d	ARP	172.30.4.107 is at 00:0c:29:12:50:1e
3	0.000199	172.30.4.222	172.30.4.107	TCP	52389 > telnet [SYN] Seq=0 Win=5840 Len=0 MSS=1460 WS=5
4	0.002030	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 WS=5
5	0.002537	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=1 Ack=1 Win=5856 Len=0
6	0.005580	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
7	0.005682	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [ACK] Seq=1 Ack=25 Win=5888 Len=0
8	0.042520	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
9	0.042604	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=25 Ack=13 Win=5856 Len=0
10	0.042658	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
11	0.044574	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
12	0.044683	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [ACK] Seq=28 Ack=28 Win=5888 Len=0
13	0.046971	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
14	0.047065	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
15	0.049608	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
16	0.071170	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
17	0.071258	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
18	0.071982	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
19	0.074900	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
20	0.087610	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
21	0.126004	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=77 Ack=125 Win=5856 Len=0
22	1.910924	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
23	1.911326	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...

TCP acknowledgments (ACKS) of data received

Installing and Configuring Telnet

No.	Time	Source	Destination	Protocol	Info
1	0.000000	Vmware_6f:53:d1: Broadcast		ARP	who has 172.30.4.107? Tell 172.30.4.222
2	0.000159	Vmware_12:50:11: Vmware_6f:53:d1: Broadcast		ARP	172.30.4.107 is at 00:0c:29:12:50:11
3	0.000199	172.30.4.222	172.30.4.107	TCP	52389 > telnet [SYN] Seq=0 Win=5840 Len=0 MSS=5440
4	0.002030	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=5440
5	0.002537	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=1 Ack=1 Win=5856 Len=0
6	0.005580	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
7	0.005682	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [ACK] Seq=1 Ack=25 Win=5888 Len=0
8	0.042520	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
9	0.042604	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=25 Ack=13 Win=5856 Len=0
10	0.042658	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
11	0.044574	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
12	0.044683	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [ACK] Seq=28 Ack=28 Win=5888 Len=0
13	0.046971	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
14	0.047065	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
15	0.049608	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
16	0.071170	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
17	0.071258	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
18	0.071982	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
19	0.074900	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
20	0.087610	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
21	0.126004	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=77 Ack=125 Win=5856 Len=0
22	1.910924	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
23	1.911326	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...

Telnet data sent

Installing and Configuring Telnet

Layer 2 Link frame
with MAC addresses

Layer 3 Network
packet with IP
addresses

Layer 4 Transport
segment with port
numbers

Layer 5 Application
data

(Untitled) - Wireshark

File Edit View Go Capture Analyze Statistics Help

Go to the packet with number...

Filter: Expression... Clear Apply

No.	Time	Source	Destination	Protocol	Info
19	0.074900	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
20	0.087610	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
21	0.126004	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=77 Ack=125 Win=5856 Len=0
22	1.910924	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
23	1.911326	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
24	1.912310	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=78 Ack=126 Win=5856 Len=0
25	2.158030	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
26	2.158083	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
27	2.159485	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=79 Ack=127 Win=5856 Len=0
28	2.416456	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...

Frame 20 (61 bytes on wire, 61 bytes captured)

- Ethernet II, Src: Vmware_12:50:1e (00:0c:29:12:50:1e), Dst: Vmware_6f:53:d9 (00:0c:29:6f:53:d9)
- Internet Protocol, Src: 172.30.4.107 (172.30.4.107), Dst: 172.30.4.222 (172.30.4.222)
- Transmission Control Protocol, Src Port: telnet (23), Dst Port: 52389 (52389), Seq: 118, Ack: 77, Len: 7
- Telnet
 - Data: login:

"login: " prompt sent to client

File: "/tmp/etherXXXXSAopg3" 91... Packets: 117 Displayed: 117 Marked: 0 Dropped: 0 Profile: Default

Installing and Configuring Telnet

The image shows a Wireshark network traffic capture window titled "(Untitled) - Wireshark". The main pane displays a list of network packets. Packet 116 is selected and highlighted in blue. The packet list pane shows the following data:

No.	Time	Source	Destination	Protocol	Info
108	16.404278	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=108 Ack=276 Win=5856 Len=0
109	16.618129	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
110	16.618441	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
111	16.618699	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=109 Ack=277 Win=5856 Len=0
112	17.261976	172.30.4.222	172.30.4.107	TELNET	Telnet Data ...
113	17.262354	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
114	17.278804	172.30.4.107	172.30.4.222	TELNET	Telnet Data ...
115	17.314309	172.30.4.222	172.30.4.107	TCP	52389 > telnet [ACK] Seq=111 Ack=279 Win=5856 Len=0
116	17.314356	172.30.4.222	172.30.4.107	TCP	52389 > telnet [FIN, ACK] Seq=111 Ack=294 Win=5856 Len=0
117	17.314378	172.30.4.107	172.30.4.222	TCP	telnet > 52389 [ACK] Seq=294 Ack=112 Win=5888 Len=0

The packet details pane for Frame 116 (60 bytes on wire, 60 bytes captured) shows the following structure:

- Ethernet II, Src: Vmware_6f:53:d9 (00:0c:29:6f:53:d9), Dst: Vmware_12:50:1e (00:0c:29:12:50:1e)
- Internet Protocol, Src: 172.30.4.222 (172.30.4.222), Dst: 172.30.4.107 (172.30.4.107)
- Transmission Control Protocol, Src Port: 52389 (52389), Dst Port: telnet (23), Seq: 111, Ack: 294, Len: 0
 - Source port: 52389 (52389)
 - Destination port: telnet (23)
 - Sequence number: 111 (relative sequence number)
 - Acknowledgement number: 294 (relative ack number)
 - Header length: 20 bytes
 - Flags: 0x11 (FIN, ACK)
 - Window size: 5856 (scaled)

The status bar at the bottom indicates: Frame (frame), 60 bytes | Packets: 117 Displayed: 117 Marked: 0 Dropped: 0 | Profile: Default

Connection terminated

Installing and Configuring Telnet

Troubleshooting

```
root@sun:~# telnet 172.30.4.107
Trying 172.30.4.107...
telnet: Unable to connect to remote host: Connection refused
```

Open the firewall on the Telnet sever to accept incoming Telnet connections

```
root@sun:~# telnet 172.30.4.107
Trying 172.30.4.107...
Connected to 172.30.4.107.
Escape character is '^]'.
getaddrinfo: localhost Name or service not known
Connection closed by foreign host.
```

On the Telnet server, insure that the server's own hostname is configured in /etc/hosts

Installing and Configuring Telnet

Troubleshooting (continued)

```
root@kate:~# telnet 172.30.4.107
Trying 172.30.4.107...
Connected to 172.30.4.107.
Escape character is '^]'.
Connection closed by foreign host.
root@kate:~# telnet 172.30.4.107
```

Edit /etc/xinetd.d/telnet file to allow connections from this client

vsftpd

vsftpd

- vsftpd = Very Secure FTP Daemon
- Licensed under the GNU General Public License
- <http://vsftpd.beasts.org/>

The screenshot shows a web browser window with the URL <http://vsftpd.beasts.org/>. The page title is "vsftpd" and the main heading is "Probably the most secure and fastest FTP server for UNIX-like systems." The page is divided into several sections:

- Main index:** Contains links for "About vsftpd", "Features", "Online source / docs", "Download vsftpd", "Who recommends vsftpd", "vsftpd security", and "vsftpd performance". A "PayPal Donate" button is located below these links.
- News:** Includes a sub-section "Other links you may be looking for" with links to a security blog and advisories. It also features two news items:
 - Nov 2009 - vsftpd-2.2.2 released:** Announces the release of vsftpd-2.2.2, highlighting a fix for a regression and mentioning a new PayPal donation button.
 - Sept. 2003 - Is any server other than vsftpd safe?:** Lists security vulnerabilities in ProFTPD, wu-ftpd, and lukemftpd.
- Logos and mentions:** At the bottom, there are logos for Red Hat and OpenBSD, with text stating that ftp.redhat.com and ftp.openbsd.org are powered by vsftpd.

Installing and Configuring vsftpd (Red Hat Family)

Is it installed?

```
[root@bigserver ~]# rpm -qa | grep vsftpd  
vsftpd-2.0.5-12.el5
```

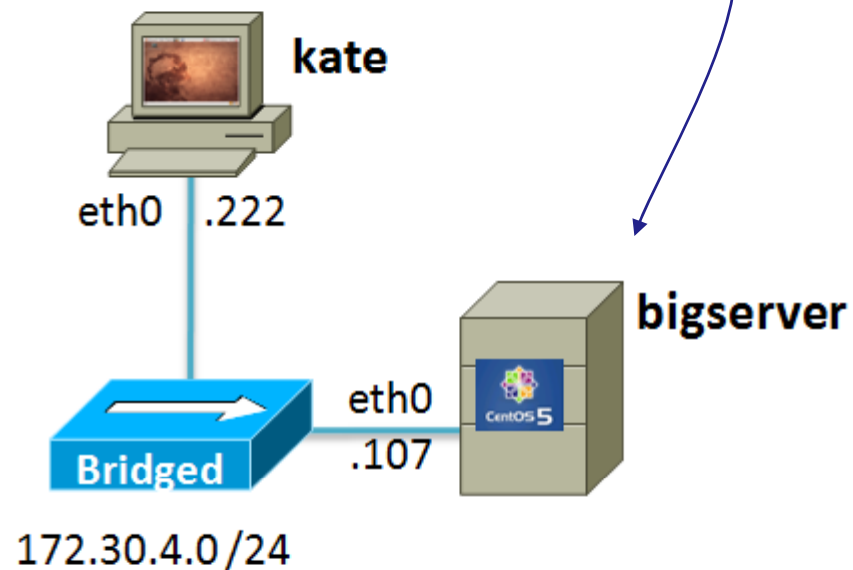
No response means it is not installed

Use `dpkg -I | grep vsftpd` on the Debian family

vsftpd

Installing vsftpd

yum install vsftpd



vsftpd

```
[root@bigserver ~]# yum install vsftpd
Loading "fastestmirror" plugin
Loading mirror speeds from cached hostfile
 * base: mirror.hmc.edu
 * updates: mirrors.easynews.com
 * addons: mirrors.cat.pdx.edu
 * extras: centos.cogentcloud.com
Setting up Install Process
Parsing package install arguments
Resolving Dependencies
--> Running transaction check
---> Package vsftpd.i386 0:2.0.5-12.el5 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

vsftpd

Dependencies Resolved

```
=====
Package                Arch      Version      Repository    Size
=====
Installing:
vsftpd                 i386     2.0.5-12.el5  base          137 k
=====
```

Transaction Summary

```
=====
Install      1 Package(s)
Update      0 Package(s)
Remove      0 Package(s)
=====
```

Total download size: 137 k

Is this ok [y/N]: y

Downloading Packages:

```
(1/1): vsftpd-2.0.5-12.el 100% |=====| 137 kB    00:00
```

Running rpm_check_debug

Running Transaction Test

Finished Transaction Test

Transaction Test Succeeded

Running Transaction

```
Installing: vsftpd                ##### [1/1]
```

Installed: vsftpd.i386 0:2.0.5-12.el5

Complete!

[root@bigserver ~]#

Installing and Configuring vsftpd

Does it use TCP Wrappers?

```
[root@bigserver ~]# type vsftpd
vsftpd is /usr/sbin/vsftpd
[root@bigserver ~]# ldd /usr/sbin/vsftpd
linux-gate.so.1 => (0x0074c000)
libssl.so.6 => /lib/libssl.so.6 (0x0012a000)
libwrap.so.0 => /usr/lib/libwrap.so.0 (0x005cb000)
libnsl.so.1 => /lib/libnsl.so.1 (0x00913000)
libpam.so.0 => /lib/libpam.so.0 (0x00b11000)
libcap.so.1 => /lib/libcap.so.1 (0x0084a000)
libdl.so.2 => /lib/libdl.so.2 (0x00110000)
libc.so.6 => /lib/libc.so.6 (0x0016f000)
libcrypto.so.6 => /lib/libcrypto.so.6 (0x002b2000)
libgssapi_krb5.so.2 => /usr/lib/libgssapi_krb5.so.2 (0x00bb4000)
libkrb5.so.3 => /usr/lib/libkrb5.so.3 (0x003e5000)
libcom_err.so.2 => /lib/libcom_err.so.2 (0x0092c000)
libk5crypto.so.3 => /usr/lib/libk5crypto.so.3 (0x0054c000)
libresolv.so.2 => /lib/libresolv.so.2 (0x00114000)
libz.so.1 => /usr/lib/libz.so.1 (0x00478000)
libaudit.so.0 => /lib/libaudit.so.0 (0x004c5000)
/lib/ld-linux.so.2 (0x0085a000)
libkrb5support.so.0 => /usr/lib/libkrb5support.so.0 (0x00fb5000)
libkeyutils.so.1 => /lib/libkeyutils.so.1 (0x00961000)
libselinux.so.1 => /lib/libselinux.so.1 (0x0048b000)
libsepol.so.1 => /lib/libsepol.so.1 (0x004da000)
[root@bigserver ~]#
```

yes it does

Installing and Configuring vsftpd

Edit the configuration file

```
[root@arwen ~]# cat /etc/vsftpd/vsftpd.conf
[root@bigserver ~]# cat /etc/vsftpd/vsftpd.conf
# Example config file /etc/vsftpd/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.

< snipped >

# You may fully customise the login banner string:
ftpd_banner=Welcome to the Simms FTP service.

< snipped >

tcp_wrappers=YES
[root@bigserver ~]#
```

Installing and Configuring vsftpd (Red Hat Family)

Start or restart service

```
[root@bigserver ~]# service vsftpd start  
Starting vsftpd for vsftpd: [ OK ]  
[root@bigserver ~]#
```

Automatically start at system boot

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

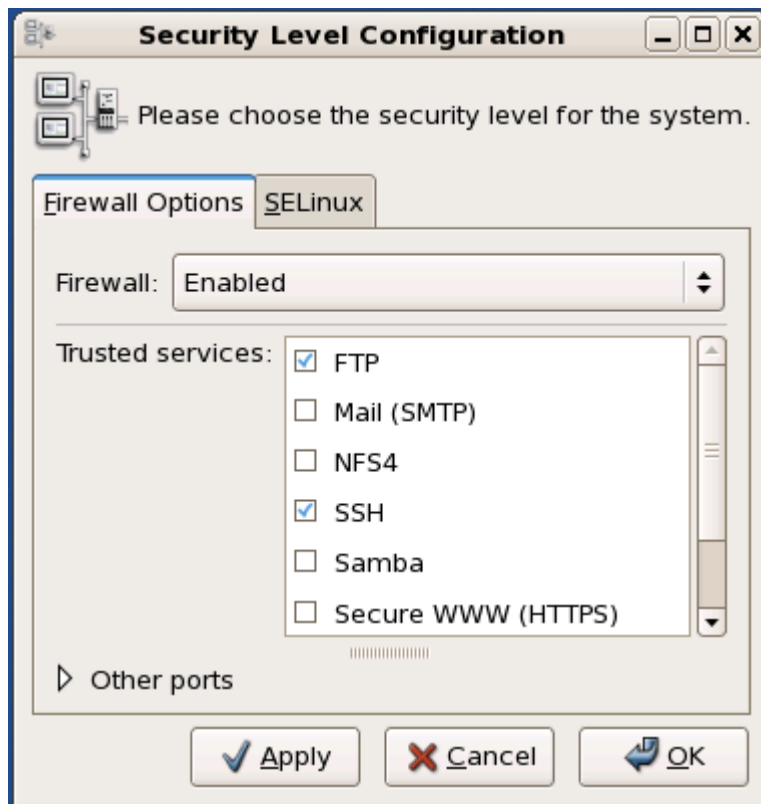
Firewall for FTP

CentOS Default

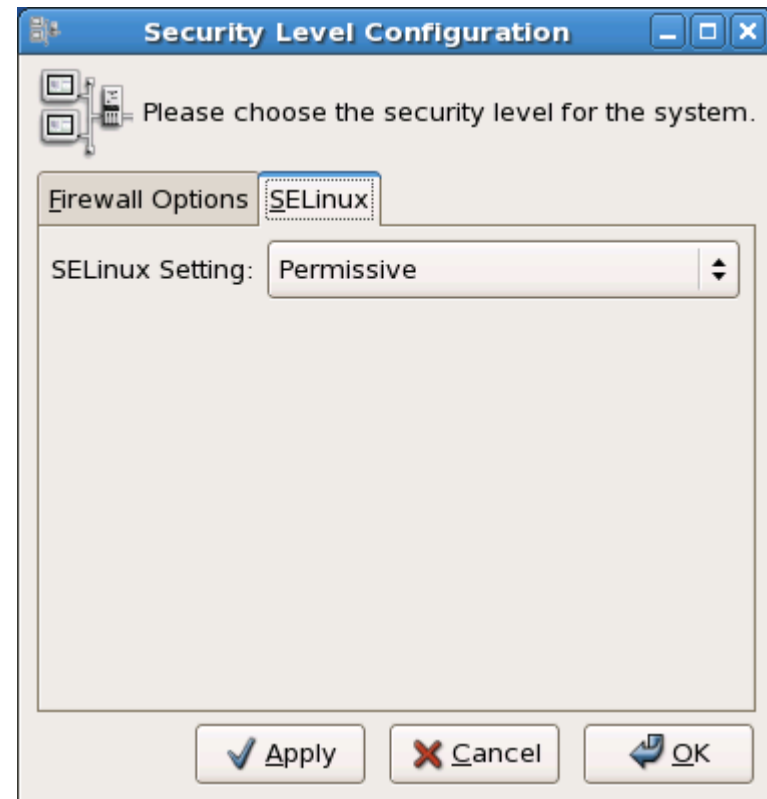
```
[root@bigserver ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -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 ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@bigserver ~]#
```

*FTP port is not open, FTP clients
will not be able to connect*

Firewall for FTP (CentOS)



Open FTP port



SELinux set to Permissive

Firewall for FTP

CentOS Modified

```

root@bigserver ~]# cat /etc/sysconfig/iptables
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -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 ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 23 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
[root@bigserver ~]#

```

FTP port is now open

Installing and Configuring vsftpd

netstat

```
[root@bigserver ~]# netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 127.0.0.1:2208         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:111          0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:6000         0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:21          0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:23          0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:631        0.0.0.0:*               LISTEN
tcp      0      0 0.0.0.0:792         0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:25        0.0.0.0:*               LISTEN
tcp      0      0 127.0.0.1:2207       0.0.0.0:*               LISTEN
tcp      0      0 :::6000             :::*                   LISTEN
tcp      0      0 :::22              :::*                   LISTEN
[root@bigserver ~]#
```

Use netstat command to see what ports your system is listening for requests on

Installing and Configuring vsftpd

netstat

```
[root@bigserver ~]# netstat -tl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 r1.localdomain:2208    *:*                     LISTEN
tcp      0      0 *:sunrpc                *:*                     LISTEN
tcp      0      0 *:x11                   *:*                     LISTEN
tcp      0      0 *:ftp                   *:*                     LISTEN
tcp      0      0 *:telnet                *:*                     LISTEN
tcp      0      0 r1.localdomain:ipp     *:*                     LISTEN
tcp      0      0 *:792                   *:*                     LISTEN
tcp      0      0 r1.localdomain:smtp    *:*                     LISTEN
tcp      0      0 r1.localdomain:2207    *:*                     LISTEN
tcp      0      0 *:x11                   *:*                     LISTEN
tcp      0      0 *:ssh                   *:*                     LISTEN
[root@bigserver ~]#
```

Use netstat command to see what ports your system is listening for requests on

Installing and Configuring vsftpd

vsftpd processes

```
[root@bigserver ~]# ps -ef | grep vsftpd
root      10541      1  0 00:29 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
nobody    10806  10541  0 00:55 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
cis192    10808  10806  0 00:55 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
root      10812  9988  0 00:57 pts/0    00:00:00 grep vsftpd
[root@bigserver ~]#
```

Individual vsftpd daemons are run for each session

Installing and Configuring vsftpd

The image shows two overlapping windows. The top window is a terminal session on a host named 'cis192@kate'. The user has connected to an FTP server at 172.30.4.107. The terminal output is as follows:

```

cis192@kate:~$ ftp 172.30.4.107
Connected to 172.30.4.107.
220 Welcome to the Simms FTP service.
Name (172.30.4.107:root): cis192
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> get myfile
local: myfile remote: myfile
No control connection for command: Success
ftp> bye
cis192@kate:~$
  
```

The bottom window is a network packet capture tool. It shows a list of captured packets. The selected packet is:

```

> ftp [ACK] Seq=1 Ack=1 Win=5856 Len=0
  
```

The packet details pane shows the following information:

- Frame 4 (93 bytes on wire, 93 bytes captured)
- Ethernet II, Src: Vmware_12:50:1e (00:0c:29:12:50:1e), Dst: Vmware_6f:53:d9 (00:0c:29:6f:53:d9)
- Internet Protocol, Src: 172.30.4.107 (172.30.4.107), Dst: 172.30.4.222 (172.30.4.222)
- Transmission Control Protocol, Src Port: ftp (21), Dst Port: 43773 (43773), Seq: 1, Ack: 1, Len: 39
- File Transfer Protocol (FTP)
 - 220 Welcome to the Simms FTP service.\r\n

An arrow points from the text 'FTP use port 21 for commands and messages' to the 'Src Port: ftp (21)' field in the packet details.

*3-way
handshake*

*Login is
transmitted in
clear text*

*Data (L5)
encapsulated in
TCP segment (L4),
which is
encapsulated in IP
packet (L3), which
is encapsulated in
Ethernet frame
(L2) which is
encapsulated in L1
frame on the wire*

Installing and Configuring vsftpd

The image shows a Wireshark capture of network traffic between 172.30.4.222 and 172.30.4.107. The capture shows a 3-way TCP handshake for port 21 (FTP) and the start of an FTP session. The first three packets are the handshake: SYN, SYN-ACK, and ACK. The fourth packet is the FTP 220 response. The fifth packet is the USER command, and the sixth is the ACK. The seventh packet is the 331 password prompt, and the eighth is the PASS command. The ninth packet is the ACK for the password.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	172.30.4.222	172.30.4.107	TCP	43773 > ftp [SYN] Seq=0 Win=5840 Len=0 MSS=1460 WS=5
2	0.000047	172.30.4.107	172.30.4.222	TCP	ftp > 43773 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 WS=5
3	0.000088	172.30.4.222	172.30.4.107	TCP	43773 > ftp [ACK] Seq=1 Ack=1 Win=5856 Len=0
4	0.024980	172.30.4.107	172.30.4.222	FTP	Response: 220 Welcome to the Simms FTP service.
5	0.025530	172.30.4.222	172.30.4.107	TCP	43773 > ftp [ACK] Seq=1 Ack=40 Win=5856 Len=0
6	4.864213	172.30.4.222	172.30.4.107	FTP	Request: USER cis192
7	4.864313	172.30.4.107	172.30.4.222	TCP	ftp > 43773 [ACK] Seq=40 Ack=14 Win=5888 Len=0
8	4.864343	172.30.4.107	172.30.4.222	FTP	Response: 331 Please specify the password.
9	4.889841	172.30.4.222	172.30.4.107	TCP	43773 > ftp [ACK] Seq=14 Ack=74 Win=5856 Len=0
10	8.731806	172.30.4.222	172.30.4.107	FTP	Request: PASS Cabrillo

Frame 4 (93 bytes on wire, 93 bytes captured)

- Ethernet II, Src: Vmware_12:50:1e (00:0c:29:12:50:1e), Dst: Vmware_6f:53:d9 (00:0c:29:6f:53:d9)
- Internet Protocol, Src: 172.30.4.107 (172.30.4.107), Dst: 172.30.4.222 (172.30.4.222)
- Transmission Control Protocol, Src Port: ftp (21), Dst Port: 43773 (43773), Seq: 1, Ack: 1, Len: 39
- File Transfer Protocol (FTP)
 - 220 Welcome to the Simms FTP service.\r\n

FTP use port 21 for commands and messages

3-way handshake

Login is transmitted in clear text

Data (L5) encapsulated in TCP segment (L4), which is encapsulated in IP packet (L3), which is encapsulated in Ethernet frame (L2) which is encapsulated in L1 frame on the wire

Installing and Configuring vsftpd

The image shows a terminal window on the left and a Wireshark window on the right. The terminal window displays the following commands and output:

```

cis192@kate: ~
cis192@kate:~$ ftp 172.30.4.107
Connect to 172.30.4.107:21.
220 Welcome to the vsftpd 2.3.4-10ubuntu1 control connection.
Name (172.30.4.107:~): kate
331 Please use the directory listing command (dir) to list files.
Password:
230 Login successful.
Remote system type is local.
Using binary mode to transfer files.
No connection to directory. Assume local files.
ftp> get myfile
local: myfile.
No connection to directory. Assume local files.
ftp> bye
cis192@kate:~$

```

The Wireshark window shows a list of captured packets. Packet 28 is highlighted, showing FTP data transfer:

No.	Time	Source	Destination	Protocol	Info
22	13.149468	172.30.4.107	172.30.4.222	FTP	Response: 200 PORT command successful. Consider using PA
23	13.149519	172.30.4.222	172.30.4.107	FTP	Request: RETR myfile
24	13.153406	172.30.4.107	172.30.4.222	TCP	ftp-data > 35677 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV
25	13.153496	172.30.4.222	172.30.4.107	TCP	35677 > ftp-data [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 M
26	13.153511	172.30.4.107	172.30.4.222	TCP	ftp-data > 35677 [ACK] Seq=1 Ack=1 Win=5888 Len=0
27	13.153540	172.30.4.107	172.30.4.222	FTP	Response: 150 Opening BINARY mode data connection for my
28	13.153807	172.30.4.107	172.30.4.222	FTP-DATA	FTP Data: 12 bytes
29	13.154286	172.30.4.107	172.30.4.222	TCP	ftp-data > 35677 [FIN, ACK] Seq=13 Ack=1 Win=5888 Len=0
30	13.186151	172.30.4.222	172.30.4.107	TCP	35677 > ftp-data [ACK] Seq=1 Ack=13 Win=5856 Len=0

The packet details pane for Frame 28 shows the following structure:

- Frame 28 (66 bytes on wire, 66 bytes captured)
- Ethernet II, Src: Vmware_12:50:1e (00:0c:29:12:50:1e), Dst: Vmware_6f:53:d9 (00:0c:29:6f:53:d9)
- Internet Protocol, Src: 172.30.4.107 (172.30.4.107), Dst: 172.30.4.222 (172.30.4.222)
- Transmission Control Protocol, Src Port: ftp-data (20), Dst Port: 35677 (35677), Seq: 1, Ack: 1, Len: 12
- FTP Data
 - FTP Data: Linux Rules\n

A blue arrow points from the text "FTP uses port 20 to transfer data" to the "FTP Data" field in the packet details pane.

Wrap

New commands, daemons and files:

service
chconfig
killall
netstat

Daemons and related configuration files

inetd	/etc/inetd.conf
portmap	/etc/etc/rpc
xinetd	/etc/etc/xinetd.d
service	/etc//etc/init.d
chconfig	/etc/rc.d/rc* .d
tcpd	/etc/hosts.allow,hosts.deny
killall	

Next Class

Assignment: Check Calendar Page

<http://simms-teach.com/cis192calendar.php>

Lab 4 due

Quiz questions for next class:

- How do you find out if vsftpd is installed?
- What two ports does FTP use?
- What command shows the ports on your system that are open and listening for requests?



Test 1



Backup

IP addresses for VM's in the classroom

Station	IP	Static 1
Instructor	172.30.1.100	172.30.1.125
Station-01	172.30.1.101	172.30.1.126
Station-02	172.30.1.102	172.30.1.127
Station-03	172.30.1.103	172.30.1.128
Station-04	172.30.1.104	172.30.1.129
Station-05	172.30.1.105	172.30.1.130
Station-06	172.30.1.106	172.30.1.131
Station-07	172.30.1.107	172.30.1.132
Station-08	172.30.1.108	172.30.1.133
Station-09	172.30.1.109	172.30.1.134
Station-10	172.30.1.110	172.30.1.135
Station-11	172.30.1.111	172.30.1.136
Station-12	172.30.1.112	172.30.1.137

Station	IP	Static 1
Station-13	172.30.1.113	172.30.1.138
Station-14	172.30.1.114	172.30.1.139
Station-15	172.30.1.115	172.30.1.140
Station-16	172.30.1.116	172.30.1.141
Station-17	172.30.1.117	172.30.1.142
Station-18	172.30.1.118	172.30.1.143
Station-19	172.30.1.119	172.30.1.144
Station-20	172.30.1.120	172.30.1.145
Station-21	172.30.1.121	172.30.1.146
Station-22	172.30.1.122	172.30.1.147
Station-23	172.30.1.123	172.30.1.148
Station-24	172.30.1.124	172.30.1.149



Note the static IP address for your station to use in the next class exercise

SSH

super
daemons

Application Layer

inet Daemon

- */etc/inetd.conf*
- */etc/services*
- */etc/protocols*

Application Layer

xinetd Daemon

Advantages

1. provides access control for TCP, UDP, and RPC services
2. Access limitations based on time
3. Extensive logging capabilities
4. Implements RFC 1413 username retrievals
5. Provides for hard reconfiguration
6. Provides numerous mechanisms to prevent denial of service attacks
7. Allows compiled in TCP_Wrappers through libwrap
8. Services may be bound to specific interfaces
9. Services may be forwarded (proxied) to another system
10. Supports ipv6

Application Layer

xinetd Daemon

Syntax:

```
service service_name
{
    attribute operator value value ...
}
```

Application Layer

xinetd Daemon

Required Attributes

1. socket_type
2. wait
3. user
4. server
5. port
6. protocol
7. rpc_version - only for RPC services
8. rpc_number - only for RPC services

Application Layer

xinetd Daemon

- Access Attributes
 1. only_from
 2. no_access
- The bind Attribute
- The redirect Attribute
- Incorporating TCP_Wrappers

Application Layer

xinetd Daemon

The xinetd Daemon command line options

1. -d
2. -syslog
3. -loop rate
4. -reuse
5. -limit
6. -logproc