Lesson Module Status

- Wall updated and emailed

- Slides –
- Properties -
- Flashcards -
- 1st minute quiz –
- Web Calendar summary –
- Web book pages –
- Commands –
- Howtos –

- Lab tested –
- Lab template in depot -
- Youtube Videos uploaded –

- VM (Classroom PC) –
- VMs (VLab) - extra gondor and arnor switches made for each pod

- Headset charged –

- Special – test published/locked
- Bring MikroTik router - done

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/

[ ] Has the phone bridge been added?

[ ] Is phone being used for voice input?

[ ] Is recording on?

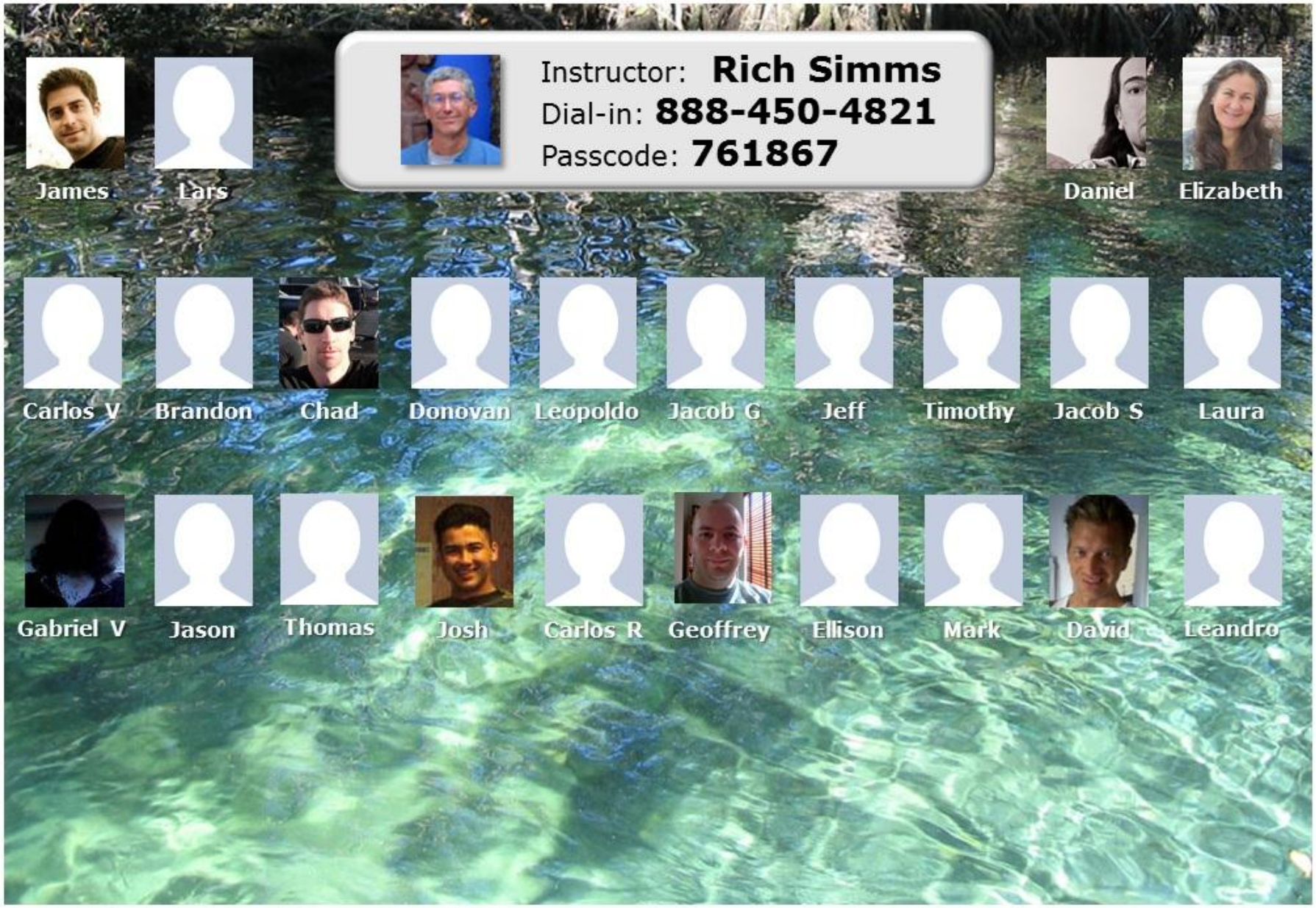
[ ] Share slides, multiple Putties started, Chrome, vlab192.rdp, VMware Workstation, wireshark

[ ] Disable spelling on PowerPoint

[ ] Repeat all ?'s for remote students

[ ] Remote student proxy

CIS 192 – Lesson 5



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

First Minute Quiz

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

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

Firewalls and NAT

Objectives	Agenda
• 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. • Use iptables to build a permissive firewall by selectively filtering packets based on protocol type. • Use Network Address Translation (NAT) to allow hosts on a private network to access the Internet.	• Quiz • Questions on previous material • Housekeeping • Wrap up transport layer • Application Layer • Super daemons • Telnet • FTP • Example firewalls and NAT • Netfilter • Lab 5 Prep • Wrap

Questions on previous material

Questions?

- Test 1?
- Previous lesson material?
- Lab assignment?
- How this class works?

Taming the Beast

(Lab 4)

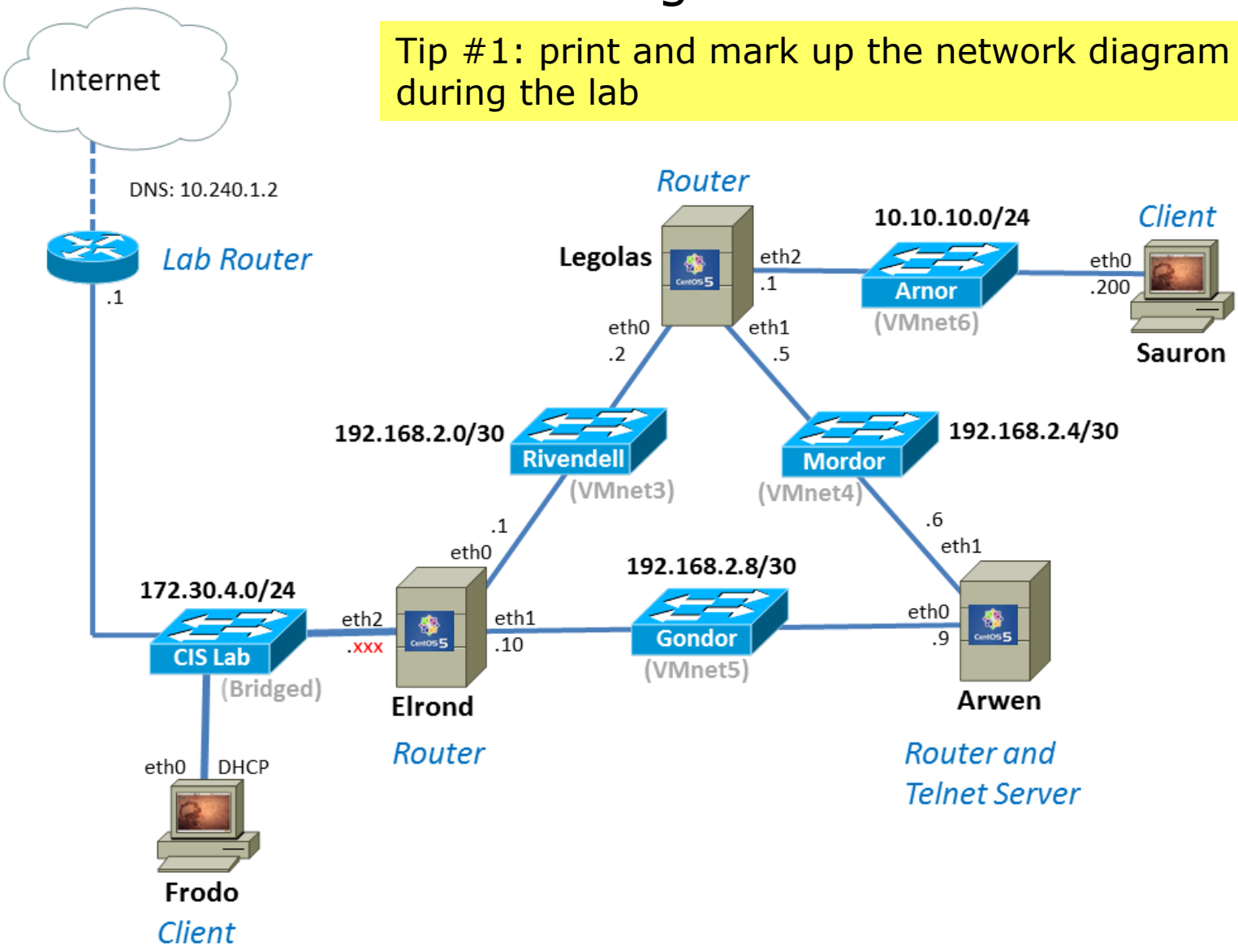
Brainteasers

1. NIC order vs eth*n* order – watch out!
   - Observed mismatch on Pods 1, 4, 6
   - Check MAC address on NIC (VM Settings) with interface (ifconfig)
   - Compare with: /etc/udev/rules.d/70-persistent-net.rules

2. Can't ping a systems "far interface" when the return route is different
   - Replaced RIP with static routes = same behavior
   - Set SELinux to permissive = same behavior
   - Some kind of reflexive DOS prevention? ... TBD
   - /etc/resolv.conf (10.240.1.2 vs 192.168.0.8) … TBD
   - Try using older distros, like good ole RH9 ... TBD
   - Try physical computers … TBD

Lab 4 – Taming with the Beast

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



Lab 4 – Taming the Beast

Tip #2: 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

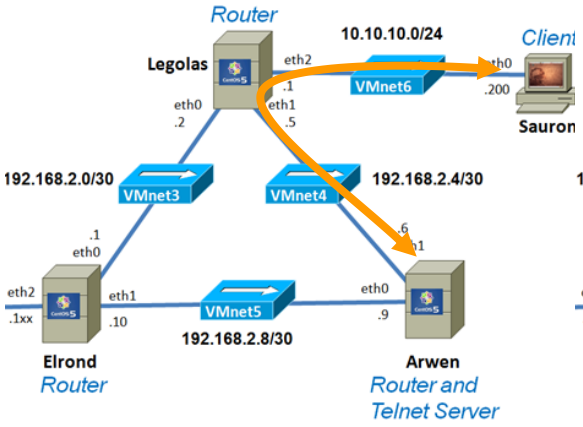
Tip #3: Create, in a one text file, key commands and all configuration files before doing lab then use scp, Copy & Paste or as a reference to configure systems.

Playing with the Beast

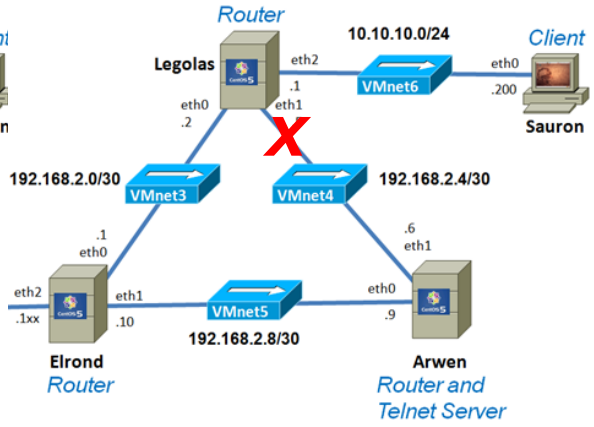
(Lab 4)

Lab 4 – Playing with the Beast

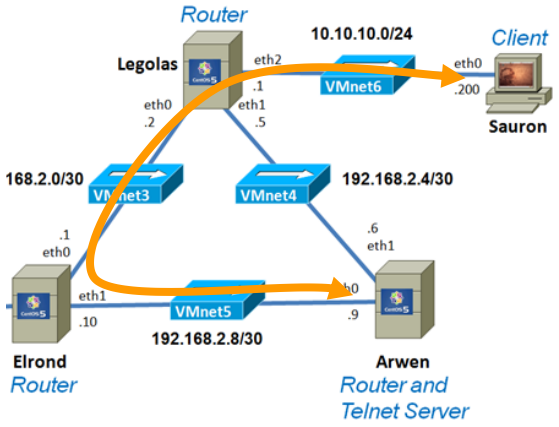
Playing #1: Force routing table to adapt to network changes you make



Pinging Arwen from Sauron via Legolas



Making trouble: The eth1 interface on Legolas is brought down with ***ifconfig eth0 down***



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

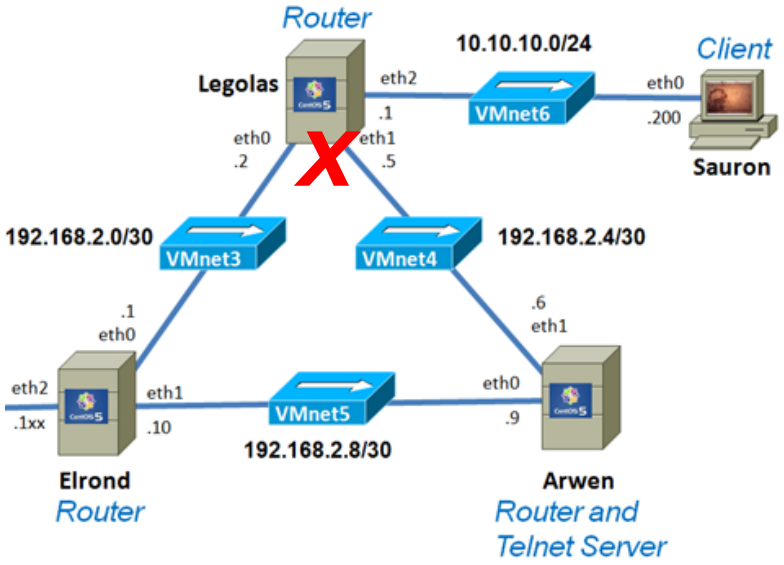
In Lab 4 you can observe routing tables update themselves as the network changes

Lab 4 – Playing with the Beast

```
root@sauron:~# while true; do ping -Rc2 arwen; sleep 10; done
PING arwen (192.168.2.6) 56(124) bytes of data.
64 bytes from arwen (192.168.2.6): icmp_req=1 ttl=63 time=2.01 ms
RR:         sauron.local (10.10.10.200)
            192.168.2.5
            arwen (192.168.2.6)
            arwen (192.168.2.6)
            legolas (10.10.10.1)
            sauron.local (10.10.10.200)
64 bytes from arwen (192.168.2.6): icmp_req=2 ttl=63 time=1.12 ms
--- arwen ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 65198ms
rtt min/avg/max/mdev = 1.127/1.572/2.018/0.447 ms

< snipped >

PING arwen (192.168.2.6) 56(124) bytes of data.
--- arwen ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1007ms

< snipped >

PING arwen (192.168.2.6) 56(124) bytes of data.
64 bytes from arwen (192.168.2.6): icmp_req=1 ttl=62 time=1.39 ms
RR:         sauron.local (10.10.10.200)
            192.168.2.2
            192.168.2.10
            arwen (192.168.2.6)
            arwen (192.168.2.6)
            elrond (192.168.2.1)
            legolas (10.10.10.1)
            sauron.local (10.10.10.200)
64 bytes from arwen (192.168.2.6): icmp_req=2 ttl=62 time=5.90 ms
--- arwen ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 90181ms
rtt min/avg/max/mdev = 1.395/3.649/5.904/2.255 ms
```

Pinging Arwen from Sauron

Trouble: Legolas eth1 is brought down



After a number of failed pings, routing tables adjust and now use longer route via Legolas and Elrond

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
[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 the debug command to enable debugging

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

Housekeeping

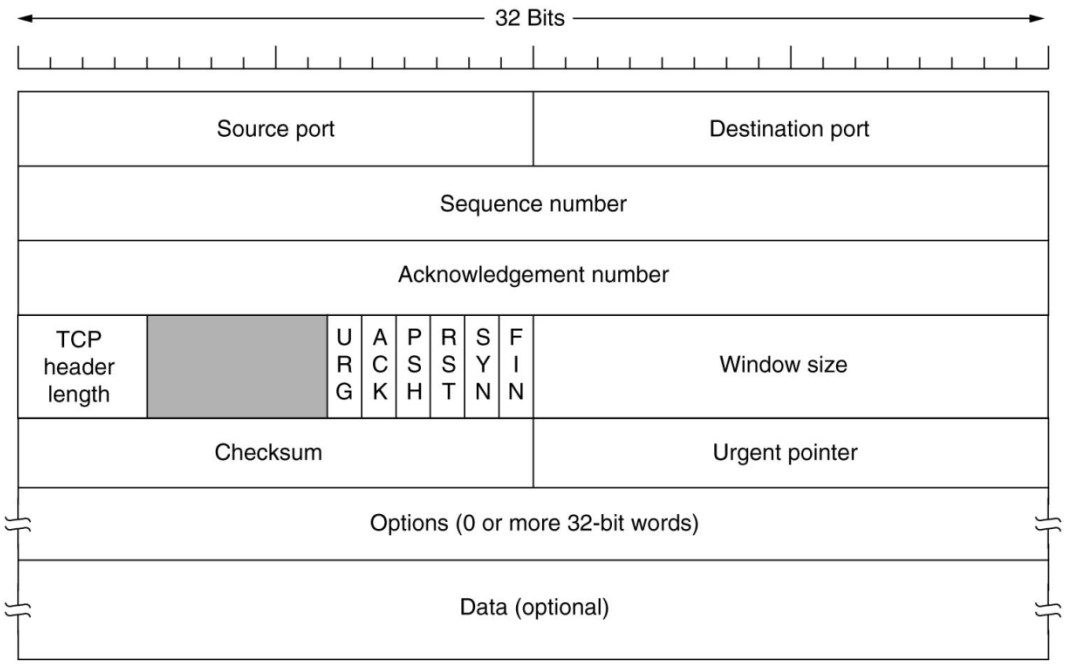
- Corrected tests in your home directory
- Lab 4 due midnight tonight
- Lab X1 due by last day of course

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.

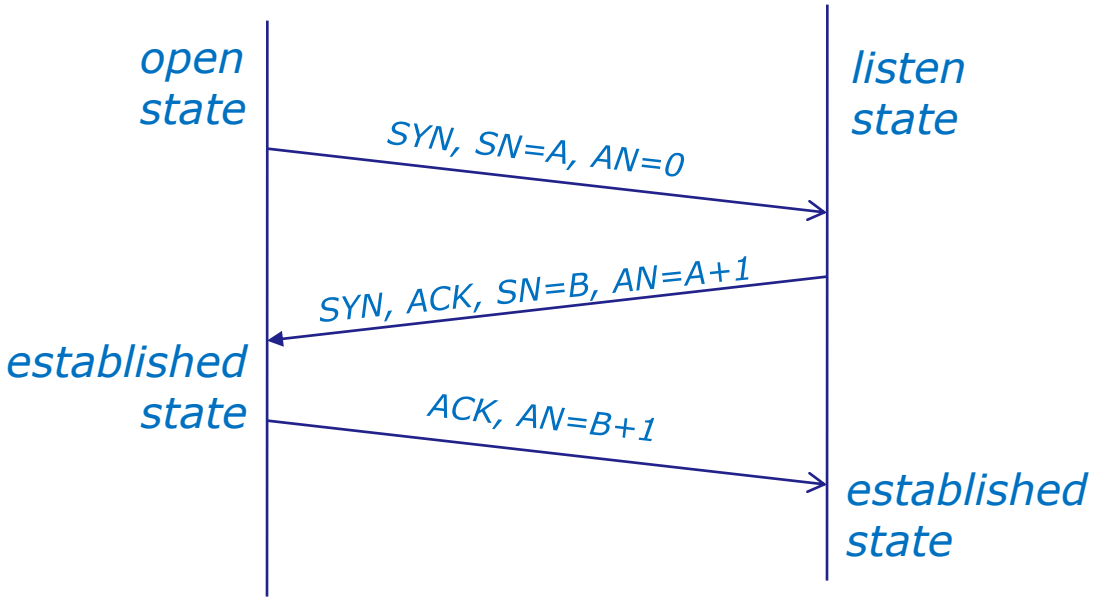
Options like SACK permit selective acknowledgement

Transport Layer

3-Way Handshake

Initiating a new TCP Connection

1. SYN
2. SYN-ACK
3. ACK



AN=Acknowledgment Number
SN=Sequence Number
ACK=ACK flag set
SYN=SYN 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 (TCP)

Continuing communications on an established connection

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

- 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.

- The SACK option
  Selective acknowledgement so only the dropped segments need to be retransmitted.

- The RST Flag
  Used to terminate a connection when an abnormal situation happens

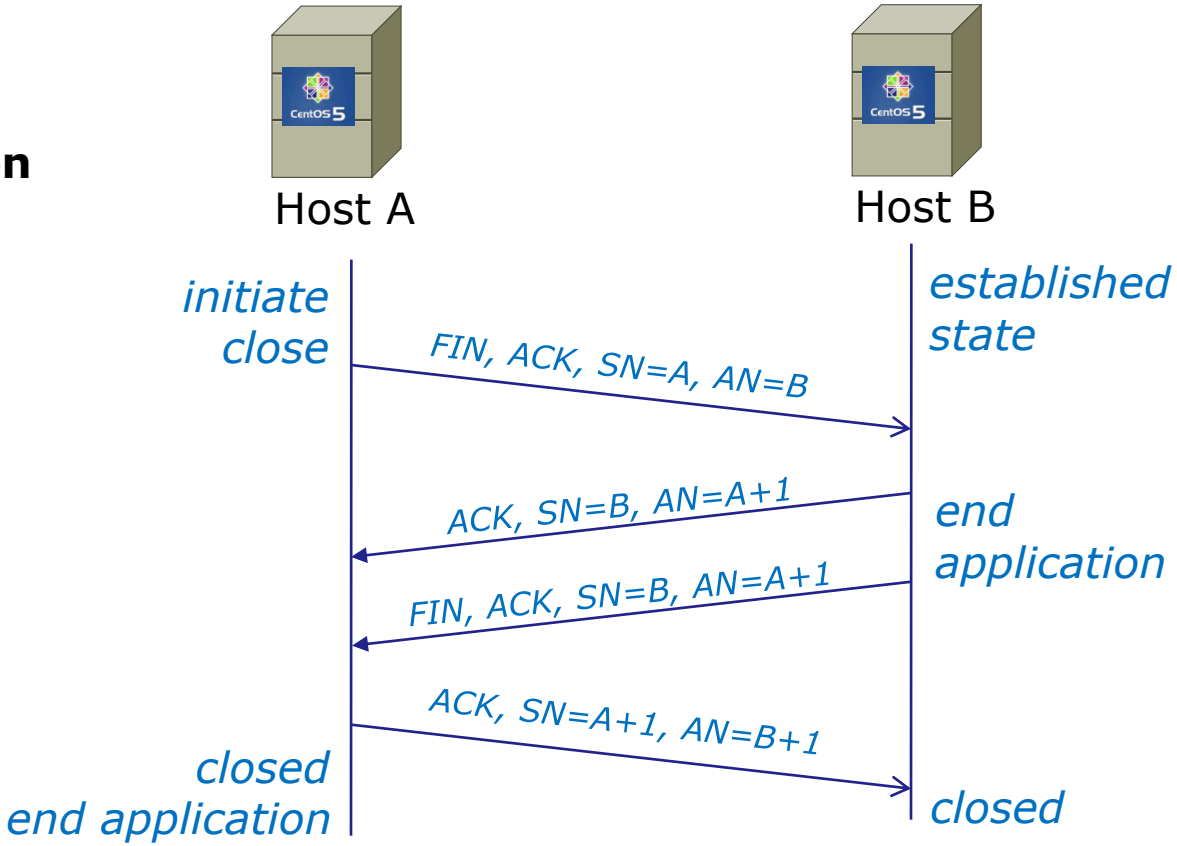
Transport Layer

Closing a TCP Connection

Four-Way Handshake

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

Closing with a shorter three-way handshake is also possible, where the Host A sends a FIN and Host B replies with a FIN & ACK (combining two steps into one) and Host A replies with an ACK.



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

Tunable Kernel Parameters

Transport Layer

Example TCP Tunable Kernel Parameters

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

In the same directory:

ip_forward	*enable/disable selective acknowledgments*
icmp_echo_ignore_broadcasts	*enable/disable responding to broadcast pings*

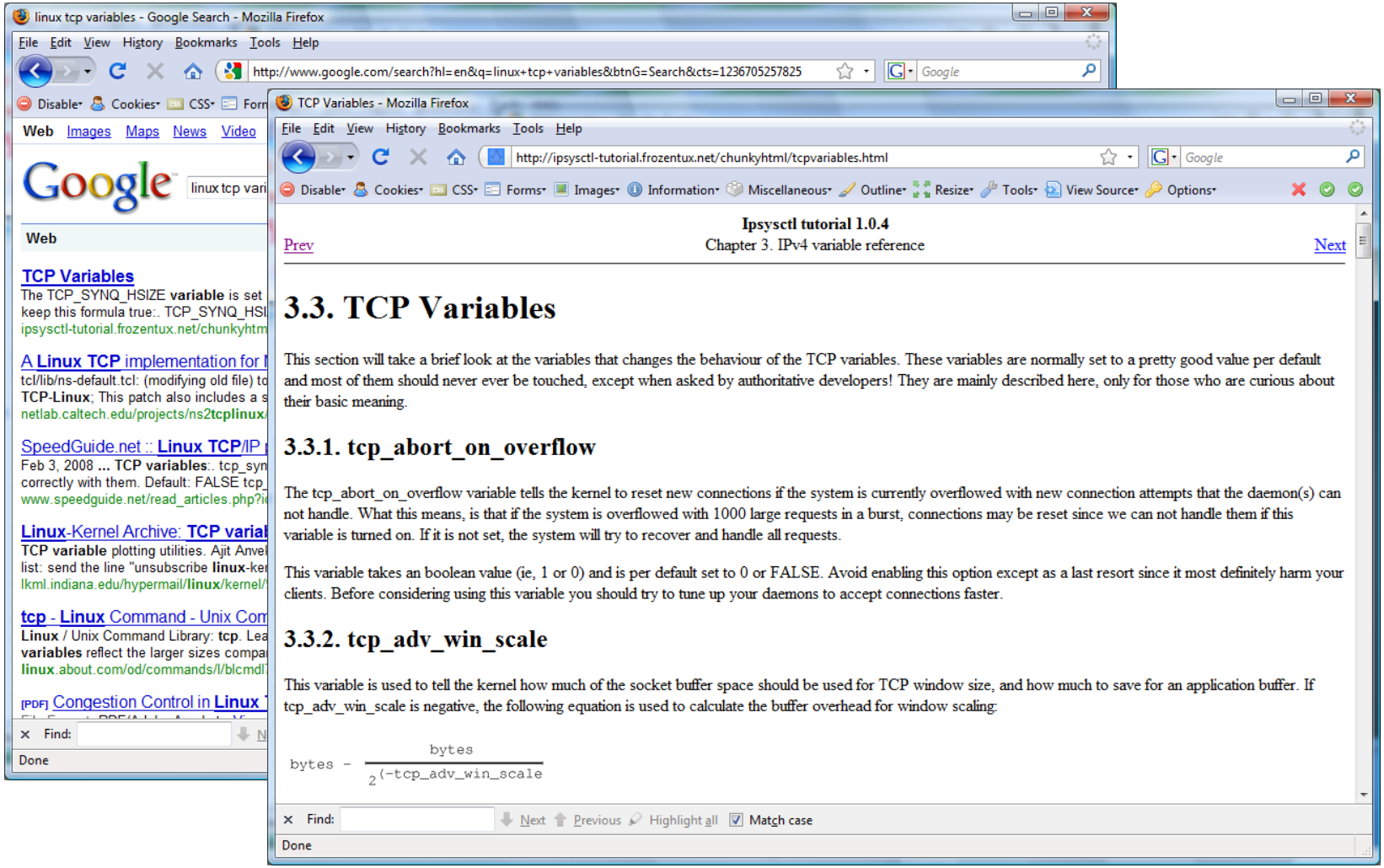
Exercise

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_conntrack_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



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

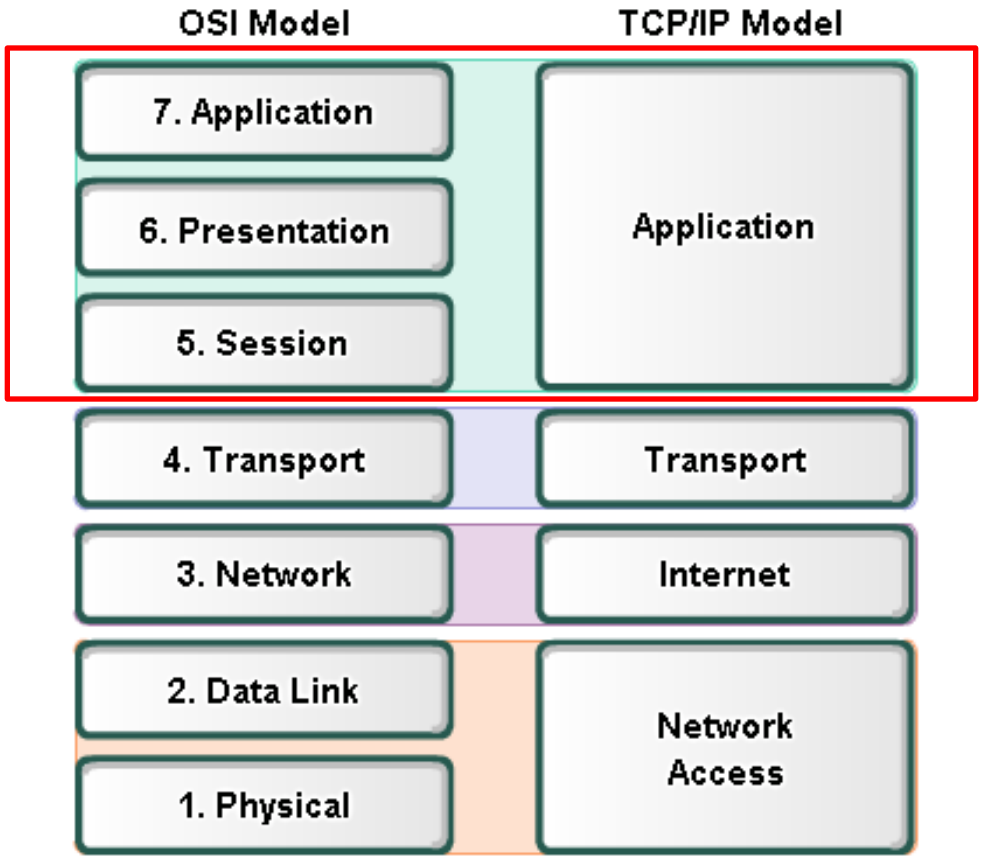
- Hijacking connections

  "… 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



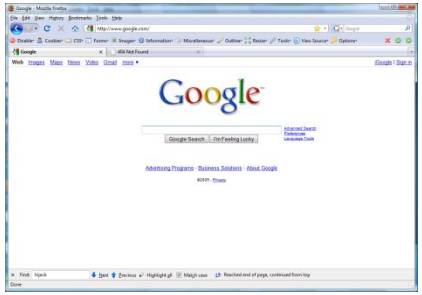
- The **Open Systems Interconnection (OSI)** model is the *most widely known internetwork reference model*.

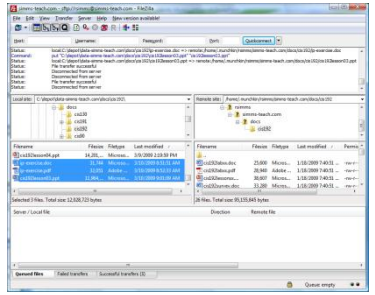
Application Layer

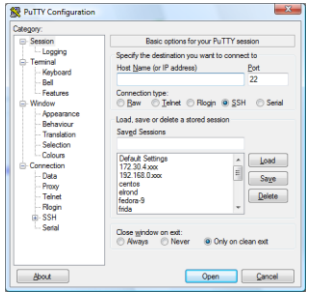
Applications

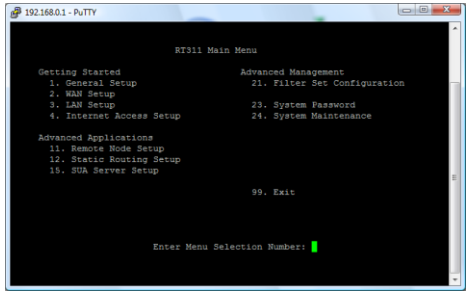
Examples:

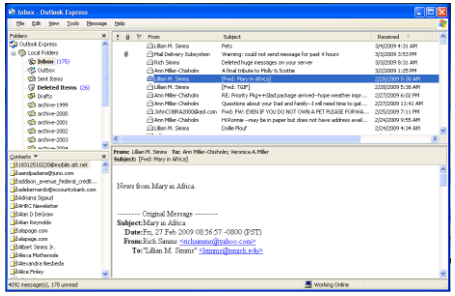
- Web servers
- FTP servers
- SSH daemon
- Telnet server
- email











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
lmtp            24/tcp                          # LMTP Mail Delivery
lmtp            24/udp                          # LMTP Mail Delivery
smtp            25/tcp          mail
smtp            25/udp          mail
```

< snipped >

```
domain          53/tcp                          # name-domain server
domain          53/udp
whois++         63/tcp
whois++         63/udp
bootps          67/tcp                          # BOOTP server
bootps          67/udp
bootpc          68/tcp          dhcpc           # BOOTP client
bootpc          68/udp          dhcpc
tftp            69/tcp
tftp            69/udp
finger          79/tcp
finger          79/udp
http            80/tcp          www www-http    # WorldWideWeb HTTP
http            80/udp          www www-http    # HyperText Transfer Protocol
kerberos        88/tcp          kerberos5 krb5  # Kerberos v5
```

< snipped >

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.

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

Using Telnet module

Using Telnet

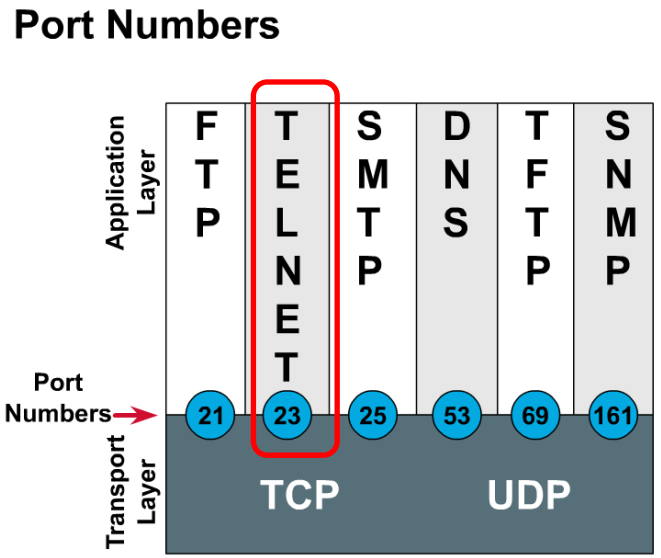
Example telnet session

Telnet

- Provides command line interface to a remote host
- Client-server model
- Uses port 23
- Not secure, uses clear text over the network that can be sniffed

Telnet uses port 23

```
[root@elrond bin]# cat /etc/services
< snipped >
telnet          23/tcp
telnet          23/udp
< snipped >
[root@elrond bin]#
```



Application Layer

The Client-Server Model

Clients

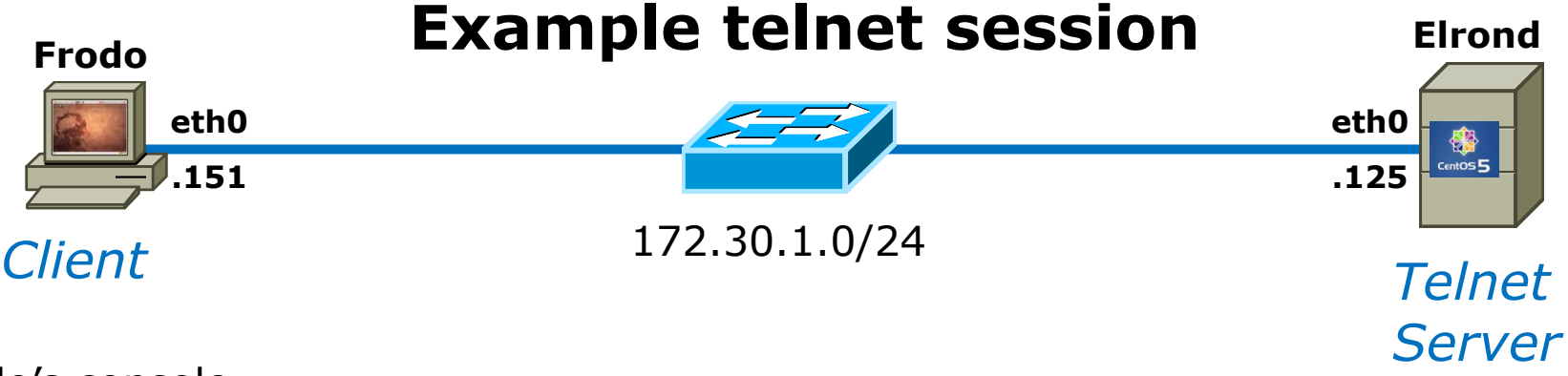
Programs that are generally run on demand, and initiate the network connection to the server.
Examples: telnet, ftp, ssh, browsers, email clients.

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

Example telnet session



Frodo's console

```
root@frodo:~# telnet 172.30.1.125
Trying 172.30.1.125...
Connected to 172.30.1.125.
Escape character is '^]'.
CentOS Linux release 6.0 (Final)
Kernel 2.6.32-71.el6.i686 on an i686
login: cis192
Password:
Last login: Sat Nov 19 17:45:01 from 172.30.1.151
[cis192@elrond ~]$ who
root     tty1         2011-11-19 15:44
root     pts/0        2011-11-19 15:54 (172.30.1.199)
cis192   pts/1        2011-11-19 18:15 (172.30.1.151)
[cis192@elrond ~]$ exit
logout
Connection closed by foreign host.
root@frodo:~#
```

The telnet client is installed on Frodo.

The telnet server is installed on Elrond.

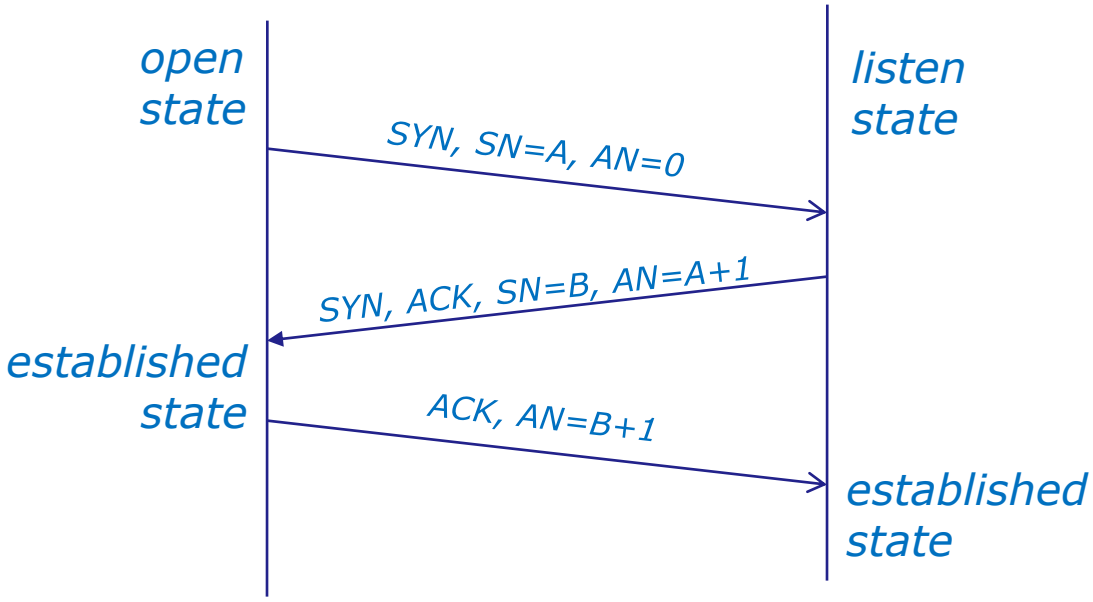
In this example, Telnet is used to login to Elrond from Frodo

Transport Layer

3-Way Handshake

Initiating a new TCP Connection

1. SYN
2. SYN-ACK
3. ACK



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

Example telnet session

No.	Time	Protocol	Source	SP	Destination	DP	Info
445	15.708754	ICMP	172.30.1.155		172.30.1.125		Echo (ping) request (id=0x196e, seq(be/le)=1/256, ttl=64)
447	15.709344	ICMP	172.30.1.125		172.30.1.155		Echo (ping) reply (id=0x196e, seq(be/le)=1/256, ttl=64)
518	16.707423	ICMP	172.30.1.155		172.30.1.125		Echo (ping) request (id=0x196e, seq(be/le)=2/512, ttl=64)
519	16.707991	ICMP	172.30.1.125		172.30.1.155		Echo (ping) reply (id=0x196e, seq(be/le)=2/512, ttl=64)
699	24.479236	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSV=5
702	24.480523	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PE
703	24.480552	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=1 Ack=1 Win=14624 Len=0 TSV=5914718 TSER=178
704	24.480978	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
705	24.481524	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [ACK] Seq=1 Ack=25 Win=5792 Len=0 TSV=1781289 TSER=591
719	24.624371	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
720	24.624470	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=25 Ack=13 Win=14624 Len=0 TSV=5914754 TSER=17
721	24.624812	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
722	24.624951	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
723	24.625134	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [ACK] Seq=28 Ack=28 Win=5792 Len=0 TSV=1781432 TSER=591
724	24.625506	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
725	24.625750	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
726	24.625924	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
727	24.627266	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
728	24.627422	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
729	24.630212	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
730	24.630413	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
733	24.643413	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...[Malformed Packet]

```
▸ Frame 1737: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
▸ Ethernet II, Src: Vmware_10:4f:d8 (00:0c:29:10:4f:d8), Dst: Vmware_db:1d:64 (00:0c:29:db:1d:64)
▸ Internet Protocol, Src: 172.30.1.125 (172.30.1.125), Dst: 172.30.1.155 (172.30.1.155)
▸ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 40192 (40192), Seq: 403, Ack: 124, Len: 0
```

3-way handshake that initiates TCP connection

Connection established

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

A socket is uniquely defined by the source IP address, source port, destination IP address, and destination port

Example telnet session

No.	Time	Protocol	Source	SP	Destination	DP	Info
445	15.708754	ICMP	172.30.1.155		172.30.1.125		Echo (ping) request (id=0x196e, seq(be/le)=1/256, ttl=64)
447	15.709344	ICMP	172.30.1.125		172.30.1.155		Echo (ping) reply (id=0x196e, seq(be/le)=1/256, ttl=64)
518	16.707423	ICMP	172.30.1.155		172.30.1.125		Echo (ping) request (id=0x196e, seq(be/le)=2/512, ttl=64)
519	16.707991	ICMP	172.30.1.125		172.30.1.155		Echo (ping) reply (id=0x196e, seq(be/le)=2/512, ttl=64)
699	24.479236	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSV=5
702	24.480523	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PE
703	24.480552	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=1 Ack=1 Win=14624 Len=0 TSV=5914718 TSER=178
704	24.480978	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
705	24.481524	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [ACK] Seq=1 Ack=25 Win=5792 Len=0 TSV=1781289 TSER=591
719	24.624371	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
720	24.624470	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=25 Ack=13 Win=14624 Len=0 TSV=5914754 TSER=17
721	24.624812	TELNET	172.30.1.155	40192	172.30.1.125	23	Tel
722	24.624951	TELNET	172.30.1.125	23	172.30.1.155	40192	Tel
723	24.625134	TCP	172.30.1.125	23	172.30.1.155	40192	tel
724	24.625506	TELNET	172.30.1.155	40192	172.30.1.125	23	Tel
725	24.625750	TELNET	172.30.1.125	23	172.30.1.155	40192	Tel
726	24.625924	TELNET	172.30.1.155	40192	172.30.1.125	23	Tel
727	24.627266	TELNET	172.30.1.125	23	172.30.1.155	40192	Tel
728	24.627422	TELNET	172.30.1.155	40192	172.30.1.125	23	Tel
729	24.630212	TELNET	172.30.1.125	23	172.30.1.155	40192	Tel
730	24.630413	TELNET	172.30.1.155	40192	172.30.1.125	23	Tel
733	24.643413	TELNET	172.30.1.125	23	172.30.1.155	40192	Tel

▸ Frame 1737: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
▸ Ethernet II, Src: Vmware_10:4f:d8 (00:0c:29:10:4f:d8), Dst: Vmware_db:1d:64 (00:0c:29:db:1d:64)
▸ Internet Protocol, Src: 172.30.1.125 (172.30.1.125), Dst: 172.30.1.155 (172.30.1.155)
▸ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 40192 (40192), Seq: 403, Ack: 124, Len: 0

Socket

Client	Server
172.30.1.155	172.30.1.125
40192	23

The socket used for the Telnet session

Transport Layer

The Transmission Control Protocol (TCP)

Continuing communications on an established connection

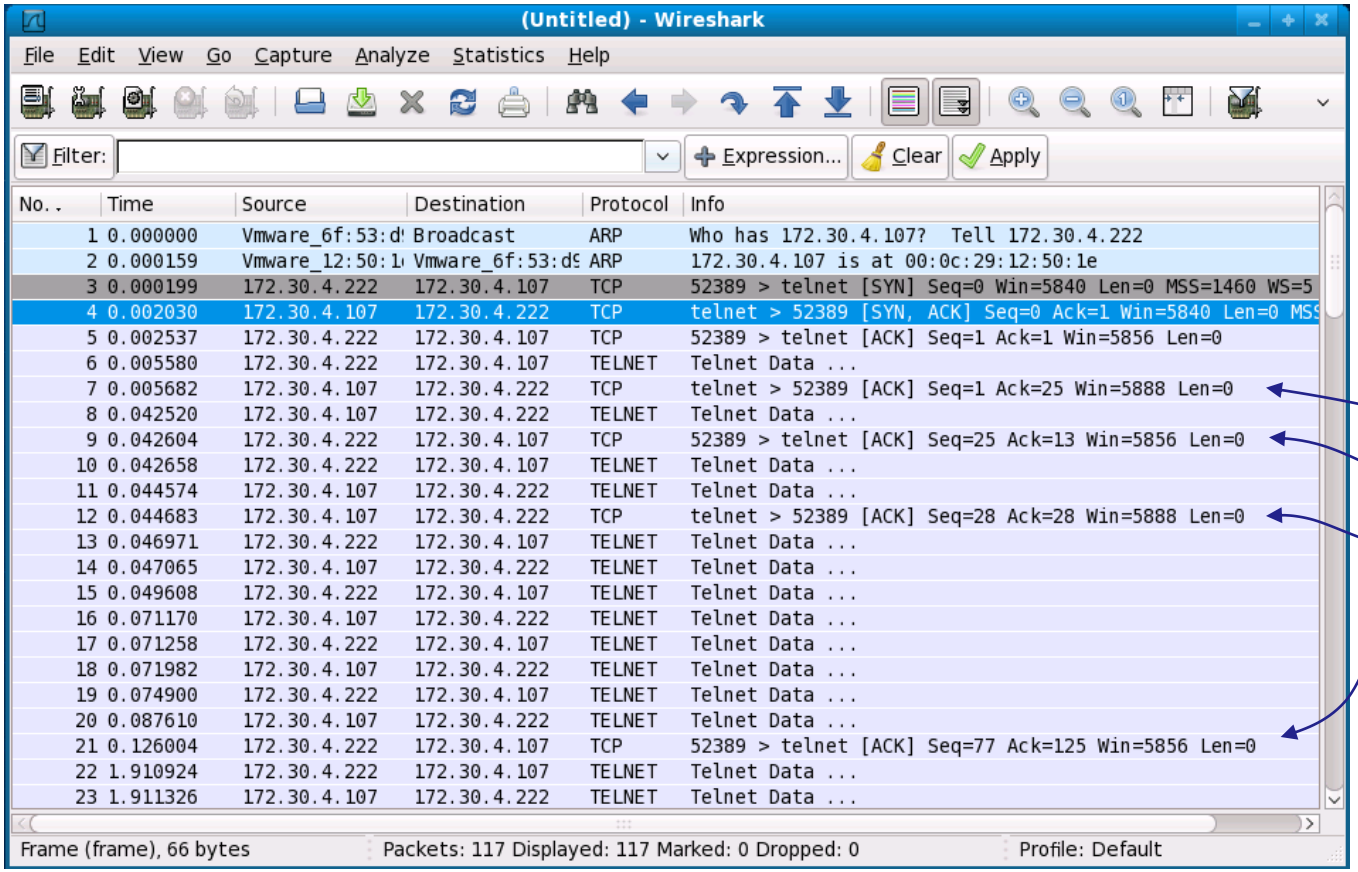
- o The Sliding Window
  Used for flow control - allows sending additional segments before an acknowledgement is received based on recipients buffer size

- 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

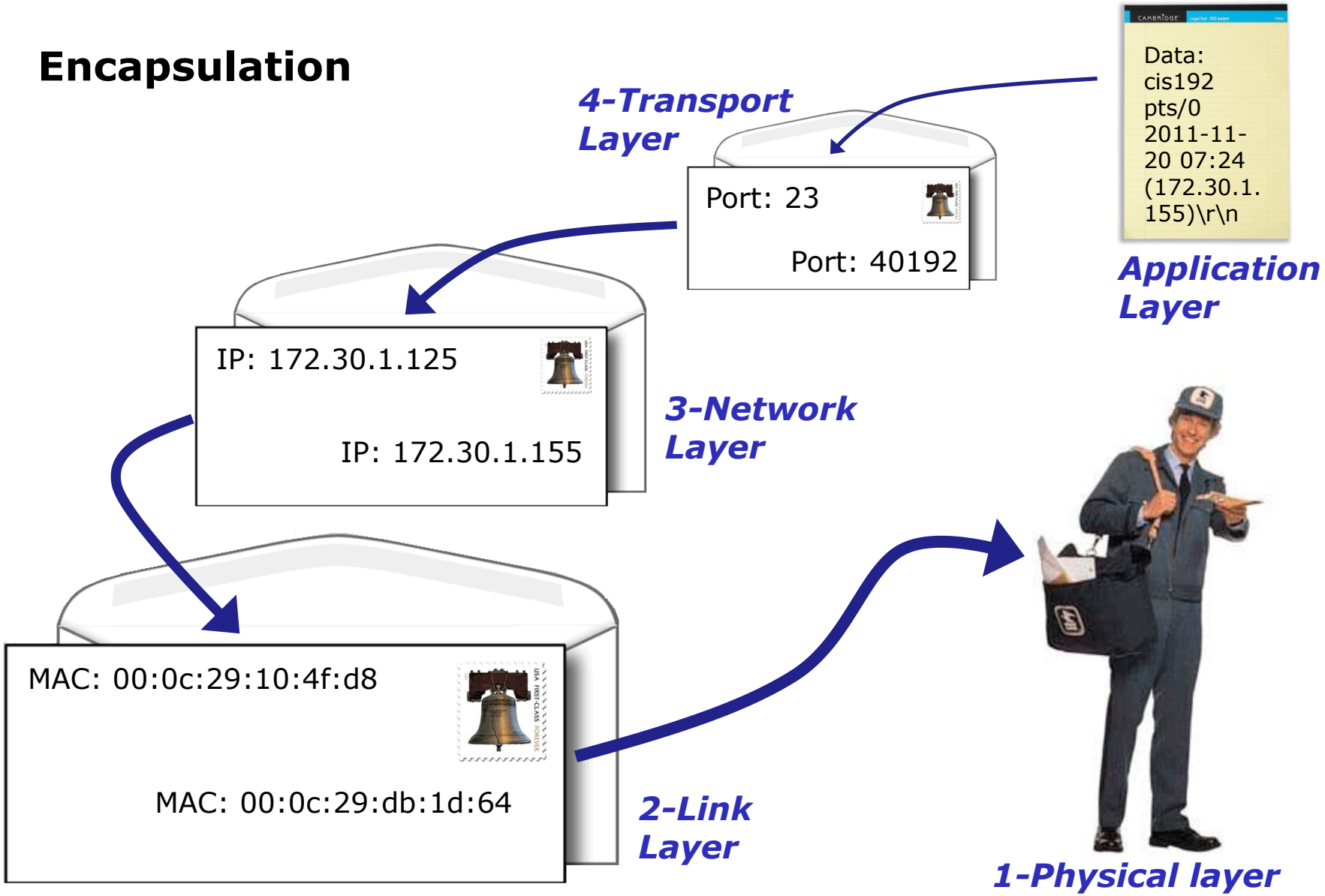
Example telnet session



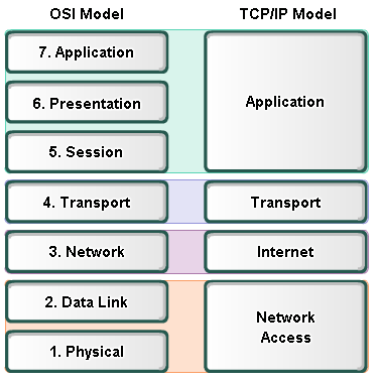
TCP acknowledgments (ACKS) of data received

Observing TCP acknowledgements sent as data is received

Encapsulation



Example telnet session



Data Link
Layer 2
(MAC addresses)

Internet
Layer 3
(IP addresses)

Network
Layer 4
(ports)

Application
Layer 5
(application data)

No.	Time	Protocol	Source	SP	Destination	DP	Info
1439	42.251893	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1440	42.254779	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1441	42.254841	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=114 Ack=252 Win=14624 Len=0 TSV=5919161 TSER=
1445	42.491914	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1446	42.494966	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1447	42.495006	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=115 Ack=253 Win=14624 Len=0 TSV=5919221 TSER=
1450	42.699982	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1451	42.703234	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1452	42.703292	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=116 Ack=254 Win=14624 Len=0 TSV=5919273 TSER=
1456	43.052011	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1457	43.056641	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1458	43.056759	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=256 Win=14624 Len=0 TSV=5919362 TSER=
1460	43.071222	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1461	43.071257	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=296 Win=14624 Len=0 TSV=5919365 TSER=
1462	43.072513	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1463	43.072545	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=351 Win=14624 Len=0 TSV=5919366 TSER
1464	43.074543	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1465	43.074568	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=390 Win=14624 Len=0 TSV=5919366 TSER=
1544	46.603941	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...

```
▸ Frame 1462: 121 bytes on wire (968 bits), 121 bytes captured (968 bits)
▸ Ethernet II, Src: Vmware_10:4f:d8 (00:0c:29:10:4f:d8), Dst: Vmware_db:1d:64 (00:0c:29:db:1d:64)
▸ Internet Protocol, Src: 172.30.1.125 (172.30.1.125), Dst: 172.30.1.155 (172.30.1.155)
▸ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 40192 (40192), Seq: 296, Ack: 118, Len: 55
▾ Telnet
    Data: cis192   pts/0        2011-11-20 07:24 (172.30.1.155)\r\n
```

Observing the network layers of encapsulation in the Telnet session

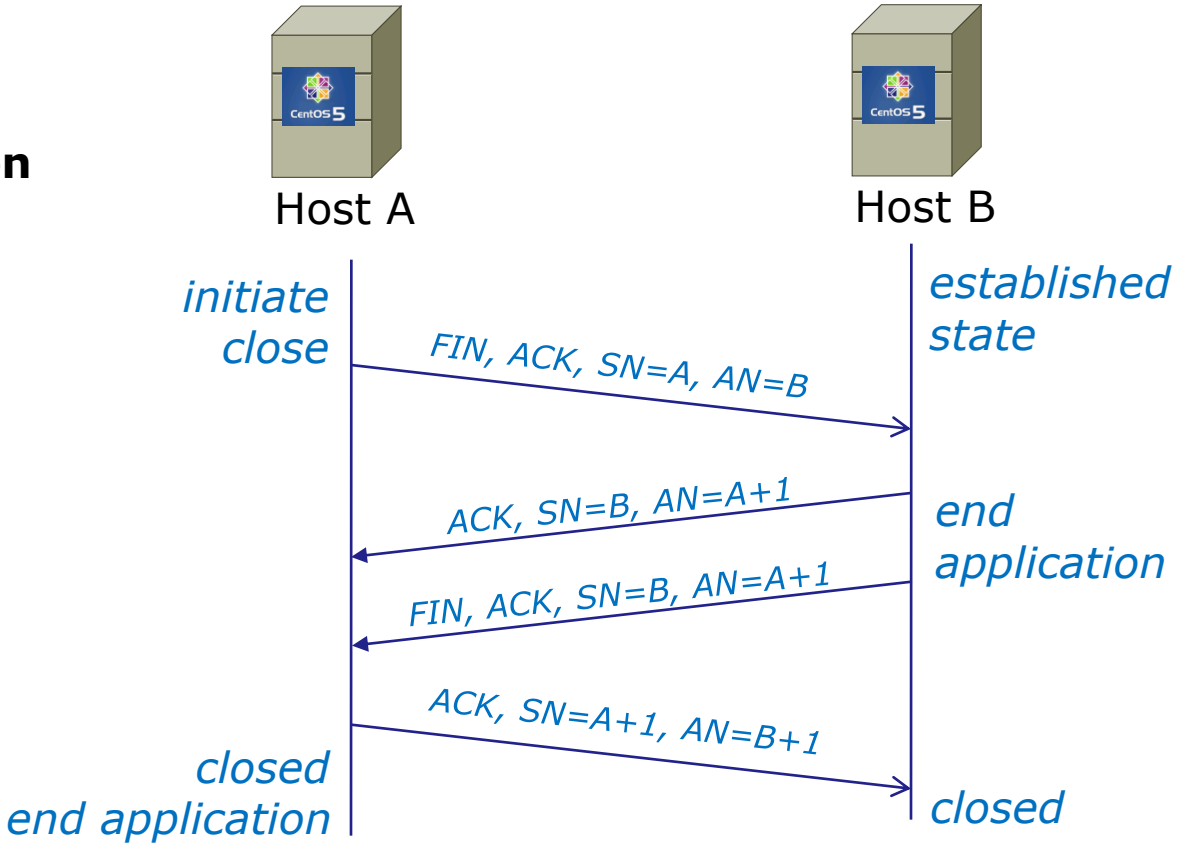
Transport Layer

Closing a TCP Connection

Four-Way Handshake

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

Closing with a shorter three-way handshake is also possible, where the Host A sends a FIN and Host B replies with a FIN & ACK (combining two steps into one) and Host A replies with an ACK.



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

Example telnet session

No.	Time	Protocol	Source	SP	Destination	DP	Info
1462	43.072513	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1463	43.072545	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=351 Win=14624 Len=0 TSV=5919366 TSER=
1464	43.074543	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1465	43.074568	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=118 Ack=390 Win=14624 Len=0 TSV=5919366 TSER=
1544	46.603941	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1545	46.607095	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1546	46.607185	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=119 Ack=391 Win=14624 Len=0 TSV=5920249 TSER=
1550	46.875997	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1551	46.879250	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1552	46.879306	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=120 Ack=392 Win=14624 Len=0 TSV=5920317 TSER=
1567	47.116046	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1568	47.118922	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1569	47.118961	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=121 Ack=393 Win=14624 Len=0 TSV=5920377 TSER=
1575	47.243526	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1576	47.245599	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1577	47.245631	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=122 Ack=394 Win=14624 Len=0 TSV=5920409 TSER=
1734	51.724011	TELNET	172.30.1.155	40192	172.30.1.125	23	Telnet Data ...
1735	51.728312	TELNET	172.30.1.125	23	172.30.1.155	40192	Telnet Data ...
1736	51.728359	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [ACK] Seq=124 Ack=403 Win=14624 Len=0 TSV=5921530 TSER=
1737	51.730616	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [FIN, ACK] Seq=403 Ack=124 Win=5792 Len=0 TSV=1808538
1738	51.730822	TCP	172.30.1.155	40192	172.30.1.125	23	40192 > telnet [FIN, ACK] Seq=124 Ack=404 Win=14624 Len=0 TSV=5921530
1739	51.731072	TCP	172.30.1.125	23	172.30.1.155	40192	telnet > 40192 [ACK] Seq=404 Ack=125 Win=5792 Len=0 TSV=1808538 TSER=

```
▶ Frame 1735: 75 bytes on wire (600 bits), 75 bytes captured (600 bits)
▶ Ethernet II, Src: Vmware_10:4f:d8 (00:0c:29:10:4f:d8), Dst: Vmware_db:1d:64 (00:0c:29:db:1d:64)
▶ Internet Protocol, Src: 172.30.1.125 (172.30.1.125), Dst: 172.30.1.155 (172.30.1.155)
▶ Transmission Control Protocol, Src Port: telnet (23), Dst Port: 40192 (40192), Seq: 394, Ack: 124, Len: 9
▼ Telnet
    Data: \n
    Data: logout\r\n
```

Handshake to close connection

Connection closed

Class Activity

Can you ping 172.30.4.253 ?

Can you ssh into 172.30.4.253 ?

Can you Telnet to 172.30.4.253 ?

Installing Telnet module

Telnet Server Installation

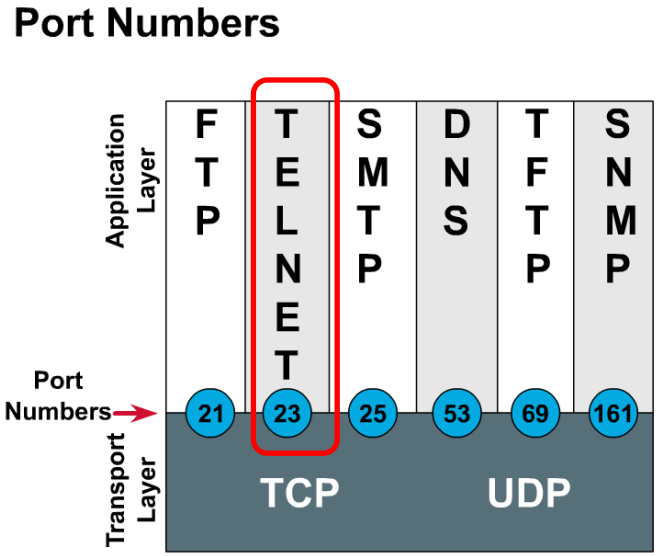
Installing and Configuring Telnet (Red Hat Family)

Telnet

- Provides command line interface to a remote host
- Client-server model
- Uses port 23
- Not secure, uses clear text over the network that can be sniffed

Telnet uses port 23

```
[root@elrond bin]# cat /etc/services
< snipped >
telnet          23/tcp
telnet          23/udp
< snipped >
[root@elrond bin]#
```



Is it installed?

Step 1 *Install software*

```
[root@elrond ~]# rpm -qa | grep telnet
telnet-0.17-46.el6.i686
telnet-server-0.17-46.el6.i686
[root@elrond ~]#
```

client

server

No response means it is not installed

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

Installing Telnet

Step 1 *Install software*

client

[root@elrond ~]# **yum install telnet**

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

server

Installing Telnet

Step 1 *Install software (continued)*

```
[root@elrond ~]# yum install telnet-server
Loading mirror speeds from cached hostfile
 * base: mirrors.sonic.net
 * extras: mirrors.xmission.com
 * updates: mirror.nwresd.org
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package telnet-server.i686 1:0.17-46.el6 set to be updated
--> Processing Dependency: xinetd for package: 1:telnet-server-0.17-46.el6.i686
--> Running transaction check
---> Package xinetd.i686 2:2.3.14-29.el6 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved
```

Note that the telnet server uses xinetd

Installing Telnet

Step 1 *Install software (continued)*

```
Dependencies Resolved

================================================================================
 Package                Arch         Version                    Repository    Size
================================================================================
Installing:
 telnet-server          i686         1:0.17-46.el6              base           36 k
Installing for dependencies:
 xinetd                 i686         2:2.3.14-29.el6            base          121 k

Transaction Summary
================================================================================
Install       2 Package(s)
Upgrade       0 Package(s)

Total download size: 156 k
Installed size: 307 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): telnet-server-0.17-46.el6.i686.rpm                 |  36 kB     00:00
(2/2): xinetd-2.3.14-29.el6.i686.rpm                      | 121 kB     00:00
--------------------------------------------------------------------------------
Total                                          109 kB/s | 156 kB     00:01
```

Note, that xinetd, the super daemon, is also installed because it is a dependency of the telnet server

Installing Telnet

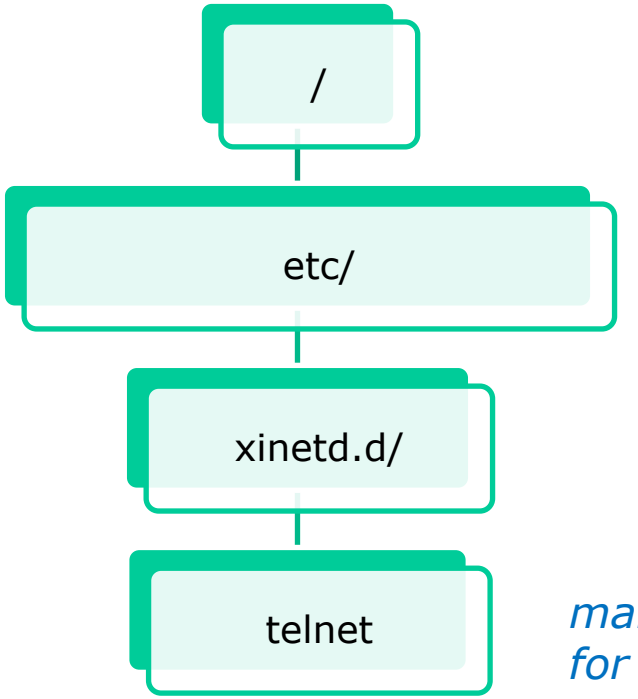
Step 1 *Install software (continued)*

```
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing     : 2:xinetd-2.3.14-29.el6.i686                              1/2
  Installing     : 1:telnet-server-0.17-46.el6.i686                         2/2

Installed:
  telnet-server.i686 1:0.17-46.el6

Dependency Installed:
  xinetd.i686 2:2.3.14-29.el6

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

Note, that xinetd, the super daemon, is also installed because it is a dependency of the telnet server

Configuring Telnet

Step 2 *Customize the configuration files*



main configuration file for telnet

Configuring Telnet

Step 2 *Customize the configuration file*

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

Change to no to enable service

Configuring Telnet

Step 2 *Customize the configuration file*

Attribute	Description
flags	Sets any of a number of attributes for the connection. *REUSE* instructs xinetd to reuse the socket for a Telnet connection.
socket_type	Sets the network socket type to *stream*.
wait	Defines whether the service is single-threaded (*yes*) or multi-threaded (*no*).
user	Defines what user *ID* the process runs under.
server	Defines the binary executable to be launched.
log_on_failure	Defines logging parameters for *log_on_failure* in addition to those already defined in xinetd.conf.
disable	Defines whether the service is active.

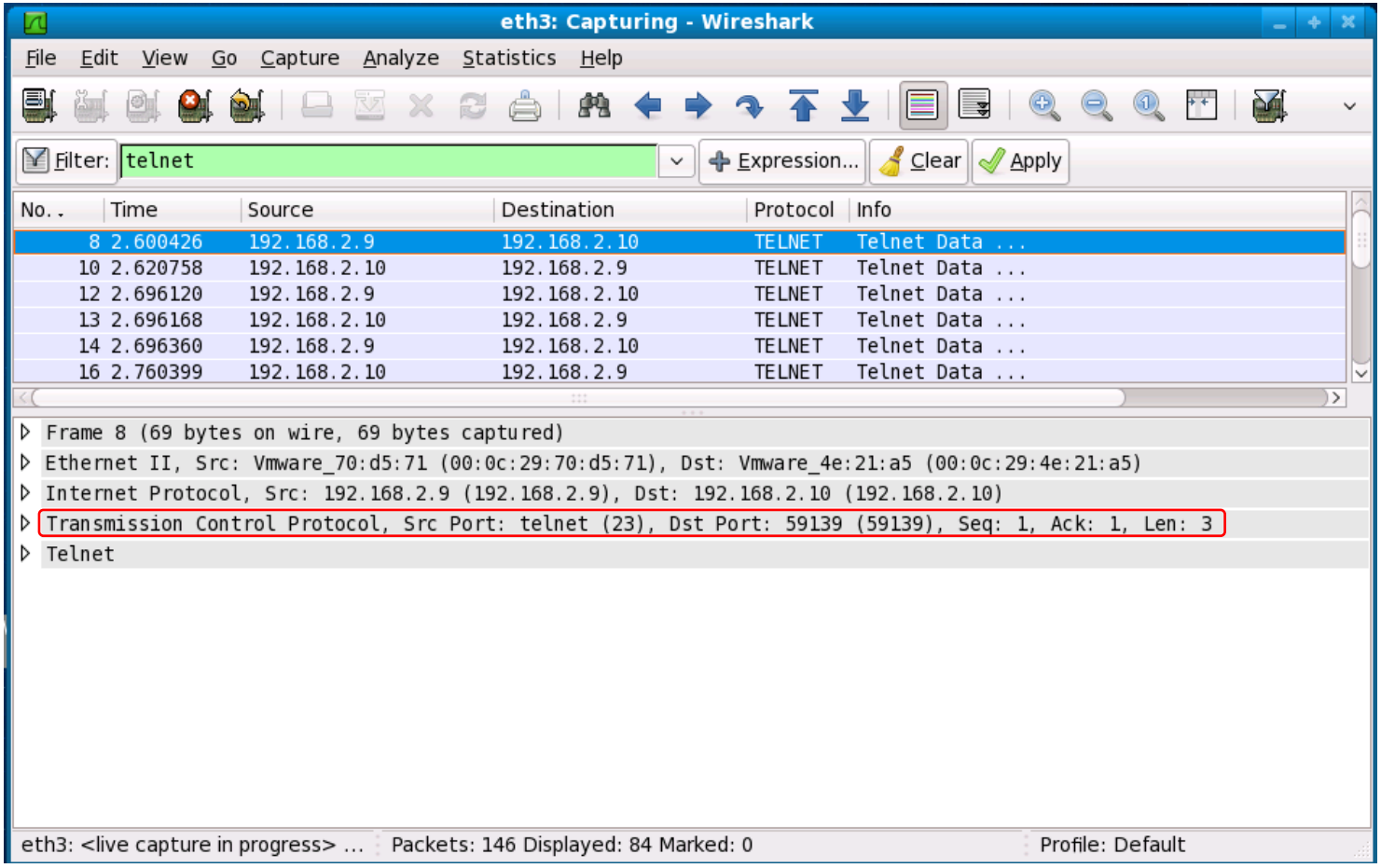
Great reference is "LINUX TCP/IP Network Administration" by Scott Mann

or use: man xinetd.conf

Firewall for Telnet

Step 3 *Modify the firewall*

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



Firewall for Telnet

Step 3 *Modify the firewall*

Show the firewall rules with line numbers

iptables -L --line-numbers

Insert rule to allow new incoming telnet connections

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

Line number (varies) to insert new rule

Verify

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

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

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

SELinux for Telnet

Step 4 *Configure SELinux*

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

Leave as enforcing

Starting Telnet service manually

Step 5 *Start the service*

```
[root@elrond ~]# service xinetd start
Starting xinetd:                                           [   OK   ]
[root@elrond ~]#
```

Starting Telnet service manually

Step 5 *Start the service*

If service is already running use the following to reread configuration files:

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

or

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

hangup signal

Starting Telnet service automatically

Step 6

To automatically start service at system boot use:

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

To later not start service at system boot use:

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

Note telnet runs under the superdaemon xinetd umbrella

Starting Telnet service automatically

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

< snipped >

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

[root@elrond ~]#  chkconfig --list | grep telnet
        telnet:         on
```

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

Monitor Telnet service

Step 7 *Verify service is running*

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

telnet daemons listens on TCP port 23

Monitor Telnet service

Step 7 *Verify service is running*

telnetd processes

```
[cis192@elrond ~]$ 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@elrond ~]$
```

Individual telnetd daemons are run for each session

Monitor Telnet service

Step 7 *Verify service is running*

netstat

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

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

Monitor Telnet service

Step 7 *Verify service is running*

netstat

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

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

Troubleshooting Telnet

Step 8 *Troubleshooting*

```
root@frodo:~# telnet 172.30.1.125
Trying 172.30.1.125...
telnet: Unable to connect to remote host: No route to host
root@frodo:~#
```

Check routing tables (route –n) and connectivity (ping).

Check firewall and make sure TCP port 23 on the Telnet sever will accept new incoming Telnet connections.

Troubleshooting Telnet

Step 8 *Troubleshooting (continued)*

```
root@frodo:~# telnet 172.30.1.125
Trying 172.30.1.125...
Connected to 172.30.1.125.
Escape character is '^]'.
Connection closed by foreign host.
root@frodo:~#
```

Check:

1. /etc/xinetd.d/telnet attributes may be blocking access:
   - only_from
   - no_access
   - access-times

2. TCP wrappers files may be blocking access:
   - /etc/hosts.allow
   - /etc/hosts.deny

Telnet Logs

Step 9 *Monitor log files*

```
[root@elrond ~]# cat /var/log/messages | grep xinetd
Nov 20 07:24:20 elrond xinetd[1391]: START: telnet pid=1855
from=::ffff:172.30.1.155
Nov 20 07:24:47 elrond xinetd[1391]: EXIT: telnet status=0 pid=1855
duration=27(sec)
Nov 20 13:33:14 elrond xinetd[1391]: Starting reconfiguration
Nov 20 13:33:14 elrond xinetd[1391]: Swapping defaults
Nov 20 13:33:14 elrond xinetd[1391]: readjusting service telnet
Nov 20 13:33:14 elrond xinetd[1391]: Reconfigured: new=0 old=1 dropped=0
(services)
Nov 20 14:22:08 elrond xinetd[1391]: START: telnet pid=3676
from=::ffff:172.30.1.155
Nov 20 14:22:16 elrond xinetd[1391]: EXIT: telnet status=0 pid=3676
duration=8(sec)
Nov 20 15:36:17 elrond xinetd[1391]: START: telnet pid=4008
from=::ffff:172.30.1.155
Nov 20 15:36:29 elrond xinetd[1391]: EXIT: telnet status=0 pid=4008
duration=12(sec)
```

Record of xinetd service stop, start, or errors

Telnet Logs

Step 9 *Monitor log files*

```
[root@elrond ~]# cat /var/log/messages | grep telnet
Nov 20 07:24:20 elrond xinetd[1391]: START: telnet pid=1855
from=::ffff:172.30.1.155
Nov 20 07:24:47 elrond xinetd[1391]: EXIT: telnet status=0 pid=1855
duration=27(sec)
Nov 20 13:33:14 elrond xinetd[1391]: readjusting service telnet
Nov 20 14:22:08 elrond xinetd[1391]: START: telnet pid=3676
from=::ffff:172.30.1.155
Nov 20 14:22:16 elrond xinetd[1391]: EXIT: telnet status=0 pid=3676
duration=8(sec)
Nov 20 15:36:17 elrond xinetd[1391]: START: telnet pid=4008
from=::ffff:172.30.1.155
Nov 20 15:36:29 elrond xinetd[1391]: EXIT: telnet status=0 pid=4008
duration=12(sec)
Nov 20 15:50:29 elrond xinetd[1391]: START: telnet pid=4096
from=::ffff:172.30.1.155
Nov 20 15:51:40 elrond xinetd[1391]: START: telnet pid=4121 from=::1
```

Record of logins by IP address

Telnet additional security

Step 10 *Configure additional security*

Attribute	Description
only_from	Allows only the specified hosts to use the service.
no_access	Blocks listed hosts from using the service.
access_times	Specifies the time range when a particular service may be used. The time range must be stated in 24-hour format notation, HH:MM-HH:MM. Example: 08:00-18:00 means the service is available from 8AM to 6PM.

Additional security attributes can be added to /etc/xinetd.d/telnet

Telnet additional security

Step 10 *Configure additional security (continued)*

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

Use only_from to restrict clients that can access the Telnet service

Telnet additional security

Step 10 *Configure additional security (continued)*

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
only_from = 192.168.16.0/22          network/prefix
```

Telnet additional security

Step 10 *Configure additional security (continued)*

TCP Wrappers

xinetd, which invokes telnet, is compiled with TCP wrappers

```
[root@elrond ~]# type xinetd
xinetd is /usr/sbin/xinetd
[root@elrond ~]# ldd /usr/sbin/xinetd
        linux-gate.so.1 =>  (0x00d00000)
        libselinux.so.1 => /lib/libselinux.so.1 (0x002fe000)
        libwrap.so.0 => /lib/libwrap.so.0 (0x005cb000)
        libnsl.so.1 => /lib/libnsl.so.1 (0x005e4000)
        libm.so.6 => /lib/libm.so.6 (0x00ed3000)
        libcrypt.so.1 => /lib/libcrypt.so.1 (0x00a7c000)
        libc.so.6 => /lib/libc.so.6 (0x00130000)
        libdl.so.2 => /lib/libdl.so.2 (0x006e9000)
        /lib/ld-linux.so.2 (0x00110000)
        libfreebl3.so => /lib/libfreebl3.so (0x0031d000)
[root@elrond ~]#
```

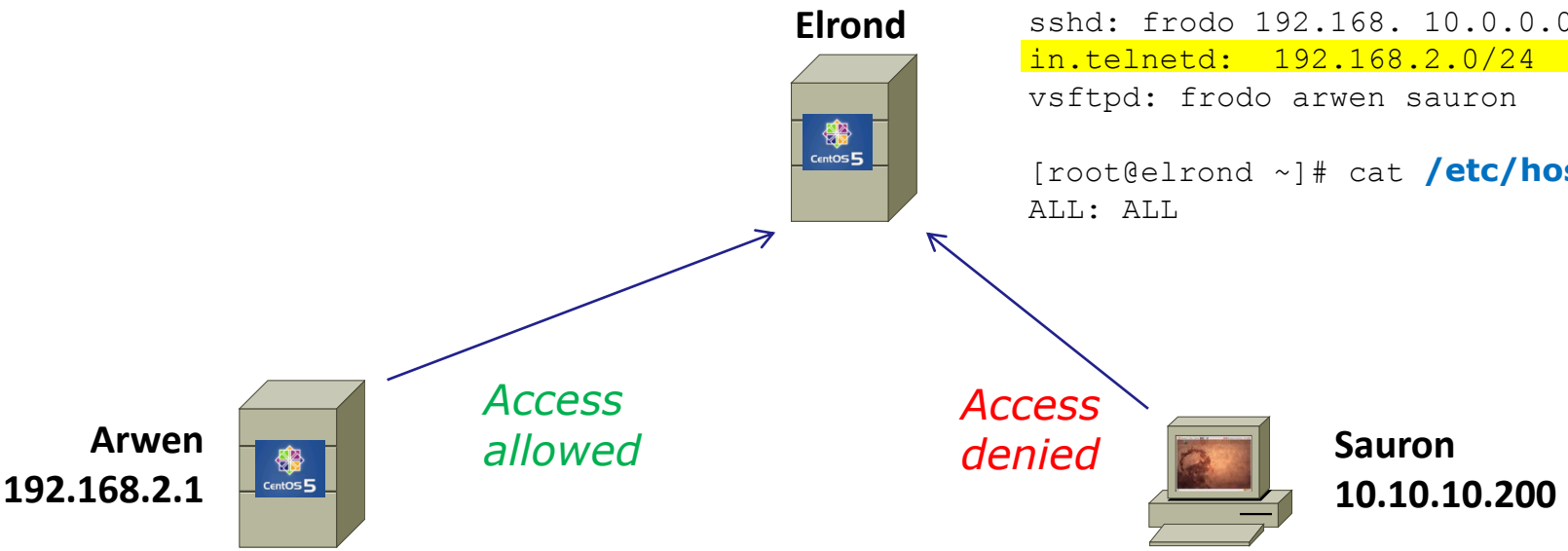
- Use **/etc/hosts.allow** for permitted hosts
- Use **/etc/hosts.deny** to ban hosts

Telnet additional security

Step 10 *Configure additional security (continued)*

TCP Wrappers

```
[root@elrond ~]# cat /etc/hosts.allow
sshd: frodo 192.168. 10.0.0.0/255.0.0.0
in.telnetd:  192.168.2.0/24  127.0.0.1
vsftpd: frodo arwen sauron

[root@elrond ~]# cat /etc/hosts.deny
ALL: ALL
```




```
[root@elrond ~]# telnet arwen
Trying 192.168.2.9...
Connected to arwen (192.168.2.9).
Escape character is '^]'.
CentOS release 5.2 (Final)
Kernel 2.6.18-92.1.22.el5 on an i686
login: cis192
Password:
Last login: Mon Mar 16 00:03:58 from arwen
[cis192@arwen ~]$
```

```
root@sauron:~# telnet arwen
Trying 192.168.2.9...
Connected to arwen.
Escape character is '^]'.
Connection closed by foreign host.
```

Class Activity

Work in teams to build a telnet server

When finished let me know your IP address so I can test logging into it

vsftpd module

vsftpd

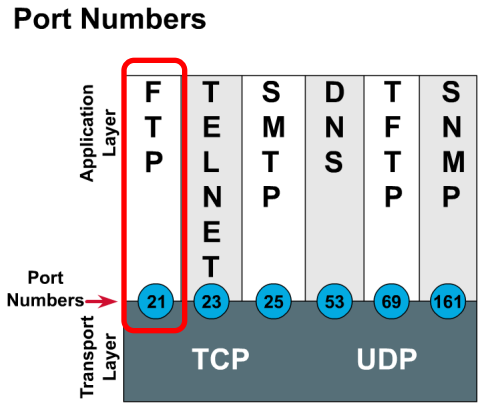
Installing and Configuring Telnet (Red Hat Family)

FTP

- File transfer protocol
- Client-server model
- Uses port 20 (for data) and 21 (for commands)
- Not secure, uses clear text over the network that can be sniffed

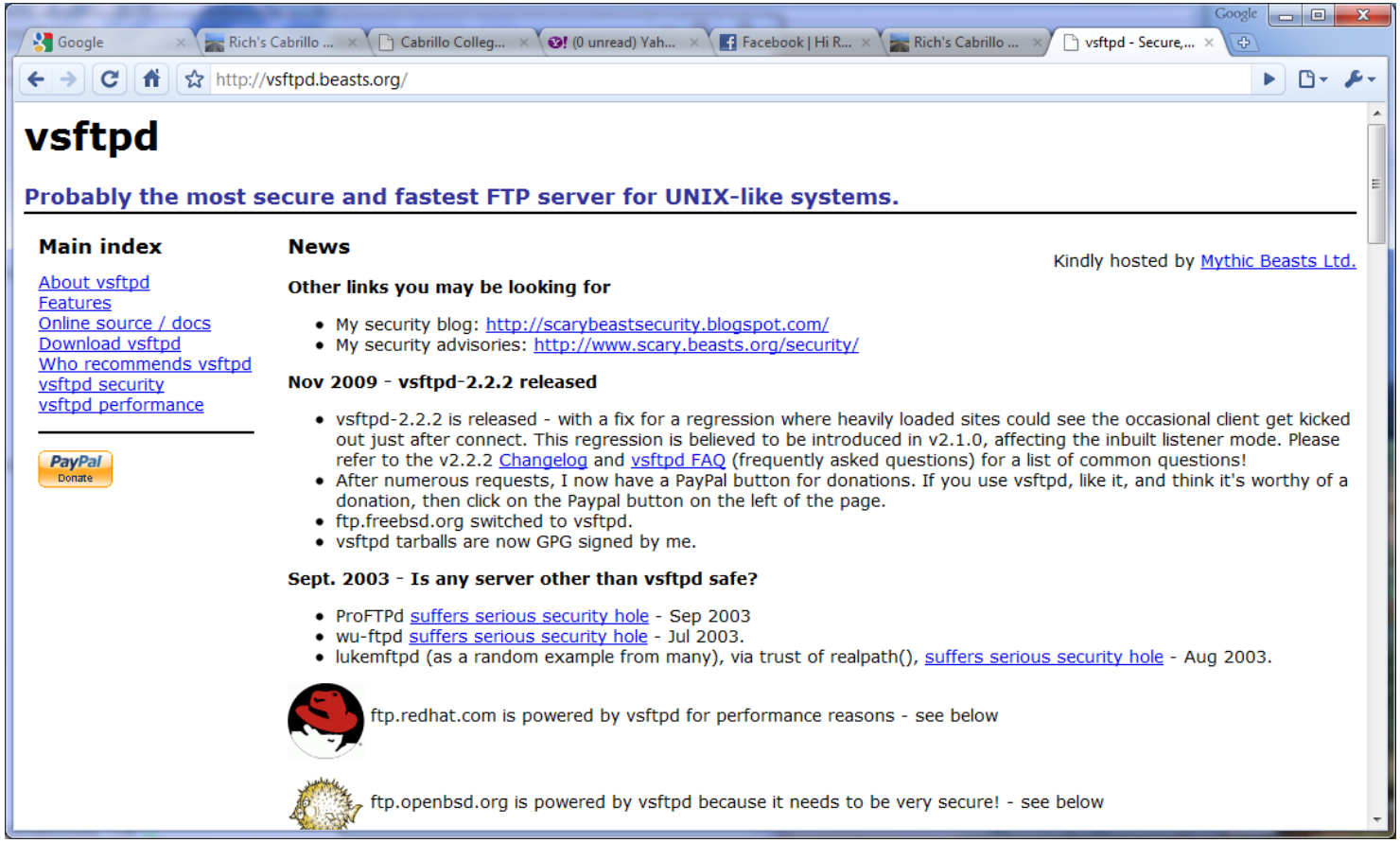
FTP uses ports 20 and 21

```
[root@elrond bin]# cat /etc/services
< snipped >
ftp-data        20/tcp
ftp-data        20/udp
# 21 is registered to ftp, but also used by fsp
ftp             21/tcp
ftp             21/udp          fsp fspd
< snipped >
[root@elrond bin]#
```



vsftpd

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



Installing and Configuring vsftpd (Red Hat Family)

Is it installed?

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

No response means it is not installed

*Use **dpkg –l | grep vsftpd** on the Debian family*

vsftpd

Installing vsftpd

Step 1 *Installing service*

yum install vsftpd

vsftpd

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

Installing and Configuring vsftpd

Step 2 *Customize the configuration file*

```
[root@celebrian ~]# cat /etc/vsftpd/vsftpd.conf
[root@celebrian ~]# 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@celebrian ~]#
```

Make your custom banner message here

Installing and Configuring vsftpd

Step 3 *Customize the firewall*

From the command line:

iptables -I INPUT 4 -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT

service iptables save

Installing and Configuring vsftpd

Step 3 *Customize the firewall (continued)*

***ip_conntrack_ftp** is a kernel module. It is used to track related FTP connections so they can get through the firewall.*

From the command line (temporary)

```
[root@celebrian ~]# modprobe ip_conntrack_ftp
[root@celebrian ~]# lsmod | grep ftp
ip_conntrack_ftp        11569  0
ip_conntrack            53281  3 ip_conntrack_ftp,ip_conntrack_netbios_ns,xt_state
[root@celebrian ~]#
```

To load at system boot (permanent), edit this file to include:

```
[root@celebrian ~]# cat /etc/sysconfig/iptables-config
# Load additional iptables modules (nat helpers)
#   Default: -none-
# Space separated list of nat helpers (e.g. 'ip_nat_ftp ip_nat_irc'), which
# are loaded after the firewall rules are applied. Options for the helpers are
# stored in /etc/modprobe.conf.
IPTABLES_MODULES="ip_conntrack_netbios_ns    ip_conntrack_ftp"
< snipped >
```

Firewall for FTP

Current firewall settings

CentOS Modified

```
[root@celebrian ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:21
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

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

FTP port is now open

Firewall for FTP

CentOS Modified

```
[root@celebrian ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.4.7 on Tue Nov 22 09:21:11 2011
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [96:7209]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 21 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Tue Nov 22 09:21:11 2011

[root@celebrian ~]# lsmod | grep ftp
nf_conntrack_ftp       10449  0
nf_conntrack           66010  4
nf_conntrack_ftp,nf_conntrack_ipv4,nf_conntrack_ipv6,xt_state
[root@celebrian ~]#
```

Permanent firewall settings

FTP port is now open

Module to track related FTP connections is loaded

SELinux for FTP (CentOS)

Step 4 *Configure SELinux*

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

Leave as enforcing

Installing and Configuring vsftpd (Red Hat Family)

Step 5 *Start or restart service*

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

Step 6 *Automatically start at system boot*

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

Installing and Configuring vsftpd

Step 7 *Verify service is running*

vsftpd processes

```
[root@celebrian ~]# service vsftpd status
vsftpd (pid 7979 6475) is running...
```

```
[root@celebrian ~]# ps -ef | grep vsftpd
root      6475     1  0 08:28 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
nobody    7975  6475  0 09:55 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
cis192    7979  7975  0 09:55 ?        00:00:00 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
root      7995  7866  0 09:56 pts/3    00:00:00 grep vsftpd
[root@celebrian ~]#
```

Individual vsftpd daemons are run for each session

Installing and Configuring vsftpd

netstat

```
[root@celebrian ~]# netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 127.0.0.1:2208              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@celebrian ~]#
```

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

Installing and Configuring vsftpd

netstat

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

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

Installing and Configuring vsftpd

Try it! *Create sample files on celebrian*

```
[root@celebrian ~]# cd /var/ftp/pub
[root@celebrian pub]# echo Contents > file1
[root@celebrian pub]# echo Contents > file2
[root@celebrian pub]# chmod 644 *
[root@celebrian pub]# ls -l
total 16
-rw-r--r-- 1 root root 9 Mar 17 09:09 file1
-rw-r--r-- 1 root root 9 Mar 17 09:09 file2
[root@celebrian pub]#
```

Installing and Configuring vsftpd

Try it! *On Elrond, download the files using **lftp** client from celebrian*

```
cis192@frodo:~$ lftp 172.30.4.240
lftp 172.30.4.240:~> ls
drwxr-xr-x    2 0        0            4096 Nov 22 17:10 pub
lftp 172.30.4.240:/> cd pub
lftp 172.30.4.240:/pub> ls
-rw-r--r--    1 0        0               9 Nov 22 17:10 file1
-rw-r--r--    1 0        0               9 Nov 22 17:10 file2
lftp 172.30.4.240:/pub> mget file*
18 bytes transferred
Total 2 files transferred
lftp 172.30.4.240:/pub> exit
cis192@frodo:~$
```

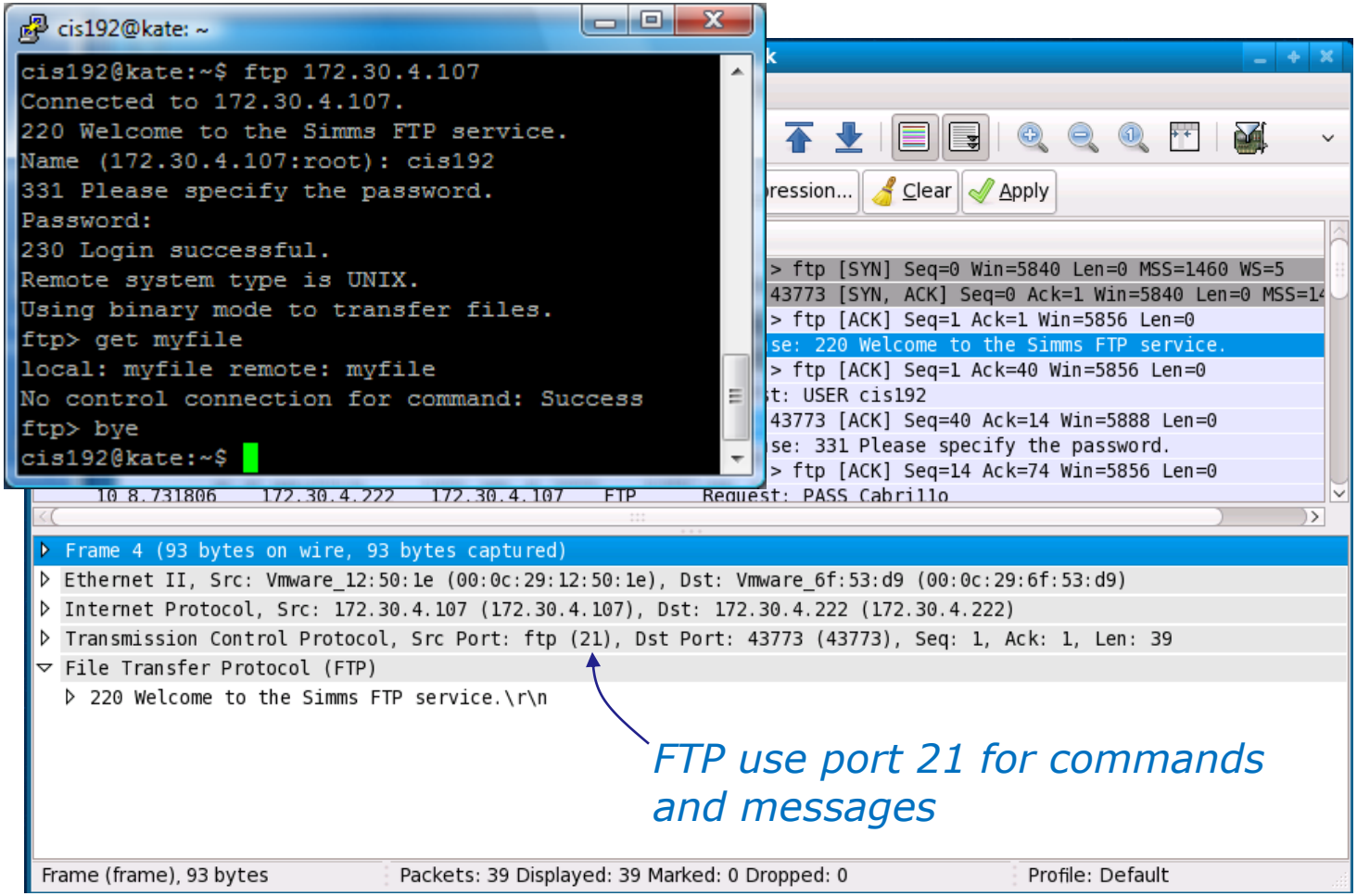
lftp is a ftp client that can run in the background, download multiple files at once and keep trying if the connection fails

Try it! Installing and Configuring vsftpd

```
cis192@frodo:~$ ftp 172.30.4.240
Connected to 172.30.4.240.
220 Welcome to Benji Simms FTP service.
Name (172.30.4.240:cis192): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        0            4096 Nov 22 17:10 pub
226 Directory send OK.
ftp> cd pub
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r--r--    1 0        0               9 Nov 22 17:10 file1
-rw-r--r--    1 0        0               9 Nov 22 17:10 file2
226 Directory send OK.
ftp> mget file*
mget file1? y
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for file1 (9 bytes).
226 Transfer complete.
9 bytes received in 0.00 secs (4.8 kB/s)
mget file2? y
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for file2 (9 bytes).
226 Transfer complete.
9 bytes received in 0.00 secs (19.9 kB/s)
ftp> exit
221 Goodbye.
cis192@frodo:~$
```

*On Elrond, download the files using regular **ftp** client from Celebrian*

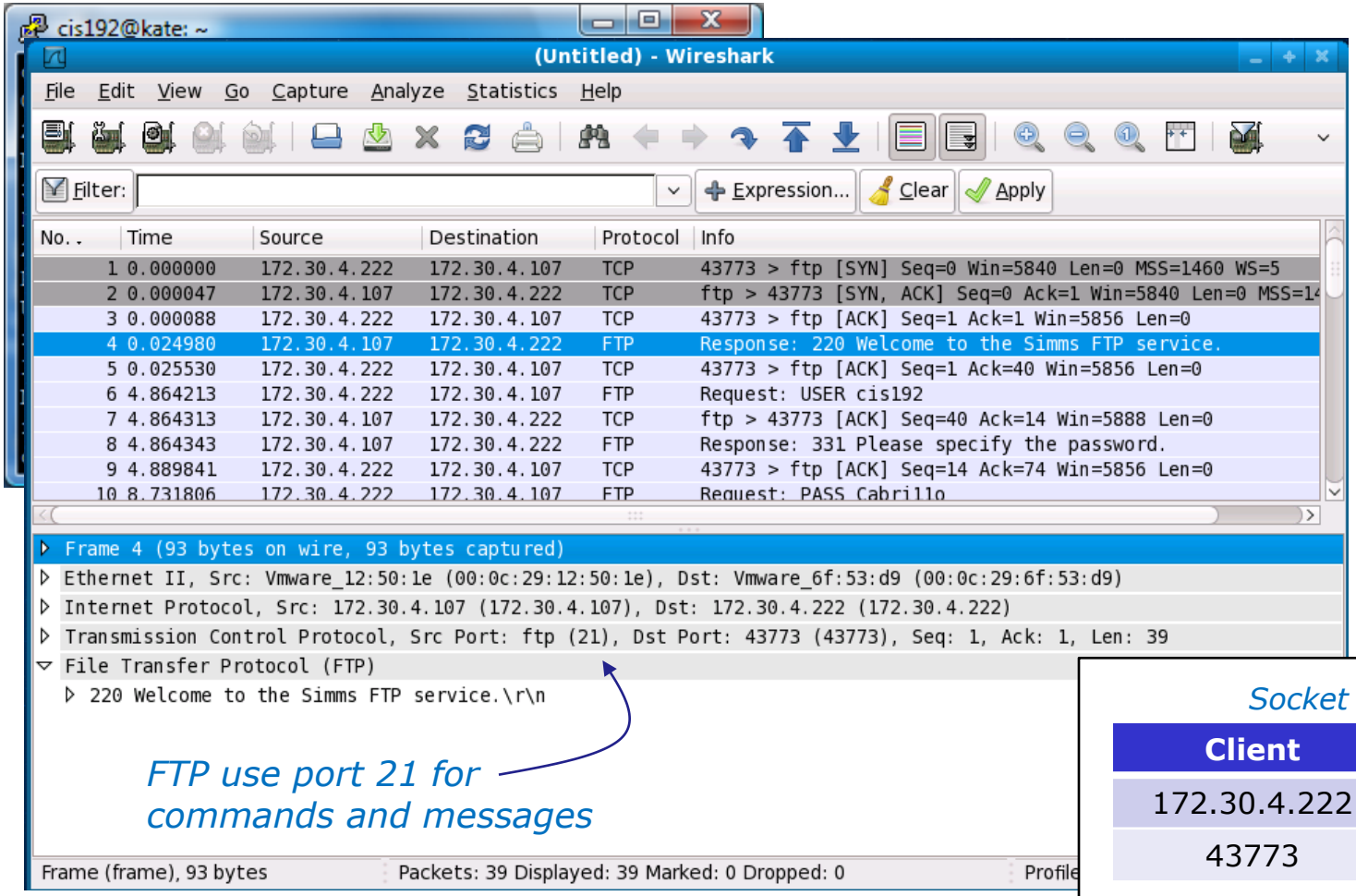
Installing and Configuring vsftpd



3-way handshake

Login is transmitted in clear text

Installing and Configuring vsftpd



FTP use port 21 for commands and messages

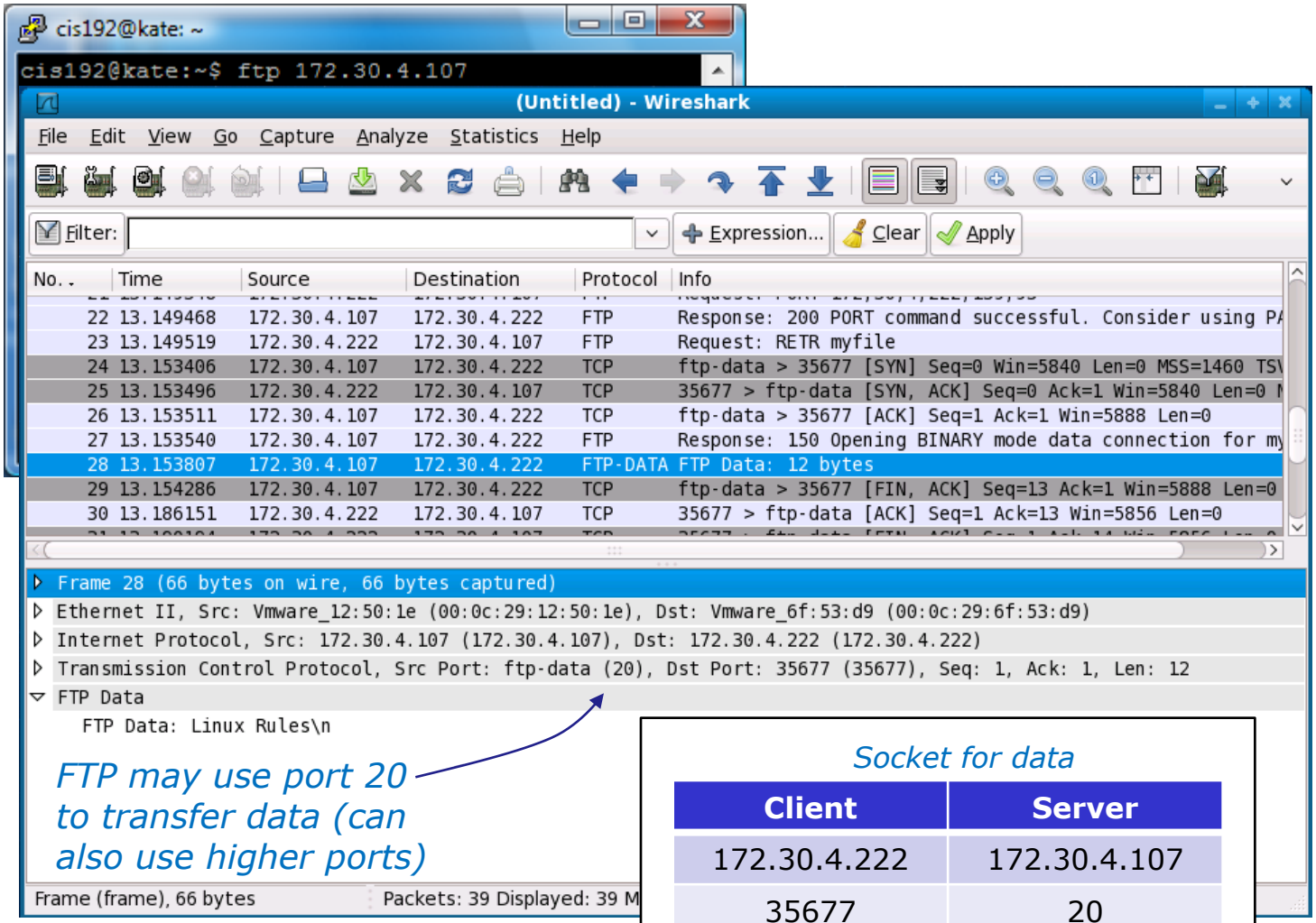
3-way handshake

Login is transmitted in clear text

Socket for commands

Client	Server
172.30.4.222	172.30.4.107
43773	21

Installing and Configuring vsftpd



FTP may use port 20 to transfer data (can also use higher ports)

Socket for data

Client	Server
172.30.4.222	172.30.4.107
35677	20

FTP data (Layer 5) is encapsulated in a TCP segment

The TCP segment (layer 4) is encapsulated in an IP packet

The IP packet (layer 3) is encapsulated in Ethernet frame

The Ethernet frame (layer 2) is placed in a low level frame that travels via electrical signals on a physical cable (Layer 1)

Installing and Configuring vsftpd

Step 8 Troubleshooting

```
[root@elrond ~]# lftp celebrian
lftp celebrian:~> ls
`ls' at 0 [Delaying before reconnect: 27]
```

On the FTP server:

- *Check FTP service is running,*
- *Check TCP port 21 is open*
- *Check ip_conntrack_ftp kernel module is loaded*

Installing and Configuring vsftpd

Step 8 Troubleshooting

```
[root@elrond ~]# ftp celebrian
ftp: connect: No route to host
ftp>
```

Open the firewall on the FTP sever to accept incoming FTP connections (TCP 21)

Use ***iptables -I RH-Firewall-1-INPUT 9 -m state --state NEW -m tcp -p tcp --dport 21 -j ACCEPT***

Installing and Configuring vsftpd

Step 8 Troubleshooting

```
[root@elrond ~]# ftp celebrian
ftp: connect: Connection refused
ftp>
```

Make sure service is up and running on FTP server.
*Use **service vsftpd start***

Installing and Configuring vsftpd

Step 8 Troubleshooting

```
[root@elrond ~]# ftp celebrian
Connected to celebrian.
220 Welcome to the SIMMS FTP service.
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
Name (celebrian:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (192,168,2,9,106,150)
ftp: connect: No route to host
ftp>
```

Make sure ip_conntrack_ftp kernel module has been loaded on FTP server. Use ***modprobe ip_conntrack_ftp***

Installing and Configuring vsftpd

Step 9 Monitor log files

```
[root@celebrian ~]# tail -f /var/log/xferlog
Wed Mar 17 15:50:41 2010 1 127.0.0.1 9 /pub/file1 b _ o a lftp@ ftp 0 * c
Wed Mar 17 15:50:41 2010 1 127.0.0.1 9 /pub/file2 b _ o a lftp@ ftp 0 * c
Wed Mar 17 16:03:00 2010 1 127.0.0.1 9 /pub/file1 b _ o a ? ftp 0 * c
Wed Mar 17 16:03:01 2010 1 127.0.0.1 9 /pub/file2 b _ o a ? ftp 0 * c
Wed Mar 17 16:35:06 2010 1 192.168.2.1 0 /pub/f* b _ o a lftp@ ftp 0 * i
Wed Mar 17 16:35:17 2010 1 192.168.2.1 0 /pub/file* b _ o a lftp@ ftp 0 * i
Wed Mar 17 16:35:21 2010 1 192.168.2.1 9 /pub/file1 b _ o a lftp@ ftp 0 * c
Wed Mar 17 16:35:21 2010 1 192.168.2.1 9 /pub/file2 b _ o a lftp@ ftp 0 * c
Wed Mar 17 16:39:27 2010 1 192.168.2.1 9 /pub/file1 b _ o a ? ftp 0 * c
Wed Mar 17 16:39:28 2010 1 192.168.2.1 9 /pub/file2 b _ o a ? ftp 0 * c
```

```
[root@celebrian ~]# cat /var/log/secure | grep -i  vsftpd
Mar 17 07:47:27 celebrian vsftpd: pam_unix(vsftpd:auth): authentication
failure; logname= uid=0 euid=0 tty=ftp ruser=cis192 rhost=elrond
user=cis192
Mar 17 08:02:56 celebrian vsftpd: pam_unix(vsftpd:auth): authentication
failure; logname= uid=0 euid=0 tty=ftp ruser=cis192 rhost=elrond
user=cis192
[root@celebrian ~]#
```

Installing and Configuring vsftpd

Does vsftpd use TCP Wrappers?

```
[root@celebrian ~]# type vsftpd
vsftpd is /usr/sbin/vsftpd
[root@celebrian ~]# 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@celebrian ~]#
```

yes it does

Installing and Configuring vsftpd

Step 10 *Configure additional security with TCP wrappers*

TCP Wrappers and vsftpd

vsftpd is compiled with TCP wrappers

- **/etc/hosts.allow** – for permitted hosts
- **/etc/hosts.deny** – to ban hosts

Installing and Configuring vsftpd

TCP Wrappers and vsftpd example

celebrian


```
[root@arwen ~]# cat /etc/hosts.allow
sshd: frodo 192.168. 10.0.0.0/255.0.0.0
in.telnetd:  192.168.2.10  127.0.0.1
vsftpd: frodo arwen celebrian
```

For vsftpd, only Frodo, celebrian and Sauron hosts are allowed

Nosmo at 172.30.1.1 is NOT included

```
[root@celebrian ~]# cat /etc/hosts.deny
ALL: ALL
```

Everyone else is denied (this includes Nosmo)

Installing and Configuring vsftpd

TCP Wrappers and vsftpd example

celebrian


```
[root@celebrian ~]# cat /etc/hosts.allow
sshd: frodo 192.168. 10.0.0.0/255.0.0.0
in.telnetd:  192.168.2.10  127.0.0.1
vsftpd: frodo celebrian sauron

[root@celebrian ~]# cat /etc/hosts.deny
ALL: ALL
```

Sauron

Access permitted

```
root@sauron:~# ftp celebrian
Connected to celebrian.
220 Welcome to the Cabrillo Super FTP service.
Name (celebrian:cis192): cis192
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> bye
221 Goodbye.
root@sauron:~#
```

Nosmo

Access denied

```
[root@nosmo root]# ftp 192.168.2.9
Connected to 192.168.2.9 (192.168.2.9).
421 Service not available.
ftp>
```

Class Activity

Work in teams to build a ftp server

When finished let me know your IP address so I can test downloading some files from it

netfilter module

Netfilter

Using iptables for firewalls and NAT

Examples

Firewalls and NAT

- Lets first look at some actual firewall and NAT configuration in all their complexity
- Then we will start breaking it down step-by-step

"previous" Red Hat default

Default "previous" Red Hat Firewall

Current settings

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

The three standard filter chains and one custom chain

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

The policy on the three filter chains is ACCEPT.

The policy is the final rule in the chain and is used when no other rules in the chain apply.

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

The INPUT and FORWARD filter chains have no rules of their own, they will use the rules in same custom chain named RH-Firewall-1-INPUT

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

Accept all traffic that arrives on the loopback interface.

Its not obvious from this output but the details can be seen in /etc/sysconfig/iptables

Default Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

All ICMP protocol traffic (of any type) is allowed.

Default Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

All ESP and AH protocol traffic is allowed.

ESP (Encapsulating Security Payload) and AH (Authentication Header) are used for IPsec.

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

All multicast DNS traffic to port 5353 is allowed.

This is used with zeroconf (Zero configuration networking) to locate DNS services on small LANs .

Default "previous" Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

All UDP and TCP protocol traffic to port 631 is allowed.

This allows CUPS to listen for IPP (Internet Printing Protocol) requests.

Default Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

Any traffic whose connection was locally originated or related to that connection is allowed

Default Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

Any new incoming connections to port 22 (ssh) are allowed

Default Red Hat Firewall

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0           icmp type 255
ACCEPT     esp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     ah   --  0.0.0.0/0            0.0.0.0/0
ACCEPT     udp  --  0.0.0.0/0            224.0.0.251         udp dpt:5353
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:631
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:631
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited
[root@elrond ~]#
```

If any of the previous rules did not apply, then send an error back using ICMP

Default "previous" Red Hat Firewall

```
[root@elrond ~]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.3.5 on Wed Mar 17 12:04:26 2010
*nat
:PREROUTING ACCEPT [1:94]
:POSTROUTING ACCEPT [6:994]
:OUTPUT ACCEPT [6:994]
COMMIT
# Completed on Wed Mar 17 12:04:26 2010
# Generated by iptables-save v1.3.5 on Wed Mar 17 12:04:26 2010
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [34:7149]
: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 -m icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p esp -j ACCEPT
-A RH-Firewall-1-INPUT -p ah -j ACCEPT
-A RH-Firewall-1-INPUT -d 224.0.0.251 -p udp -m udp --dport 5353 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
# Completed on Wed Mar 17 12:04:26 2010
[root@elrond ~]#
```

Permanent settings to be used at next system boot or when restarting the iptables service

Shows the actual iptables commands used to create the firewall

"new" Red Hat default

Default "new" Red Hat Firewall

Current settings

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

- *Much simpler than version on older version of Red Hat.*
- *The custom RH-Firewall-1-INPUT chain is no longer used.*
- *The policy is still set to ACCEPT on all three filter table chains.*

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Any traffic related to connections that originated on this system are accepted.

/etc/sysconfig/iptables
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Accept any pings

/etc/sysconfig/iptables
-A INPUT -p icmp -j ACCEPT

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Accept all loopback traffic.

/etc/sysconfig/iptables
-A INPUT -i lo -j ACCEPT

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Accept all new ssh connections

/etc/sysconfig/iptables
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT

Default "new" Red Hat Firewall

```
[root@elrond ~]# iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-
prohibited

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

Reject with a prohibited error anything else

/etc/sysconfig/iptables
-A FORWARD -j REJECT --reject-with icmp-host-prohibited

Nosmo RH9 VM

Nosmo RH9 Firewall

```
[root@nosmo root]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*filter
:INPUT ACCEPT [4229:434875]
:FORWARD ACCEPT [1481:444016]
:OUTPUT ACCEPT [3340:350240]
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*nat
:PREROUTING ACCEPT [8414:1265541]
:POSTROUTING ACCEPT [226:15381]
:OUTPUT ACCEPT [95:7826]
-A PREROUTING -d 207.62.187.53 -j DNAT --to-destination 192.168.0.1
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
[root@nosmo root]#
```

I used to use this VM at home to simulate the lab router

Nosmo RH9 Firewall

```
[root@nosmo root]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*filter
:INPUT ACCEPT [4229:434875]
:FORWARD ACCEPT [1481:444016]
:OUTPUT ACCEPT [3340:350240]
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*nat
:PREROUTING ACCEPT [8414:1265541]
:POSTROUTING ACCEPT [226:15381]
:OUTPUT ACCEPT [95:7826]
-A PREROUTING -d 207.62.187.53 -j DNAT --to-destination 192.168.0.1
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
[root@nosmo root]#
```

This is the DNS server IP address I use at home which goes to my Netgear router

Forward DNS traffic intended for Bubbles (Cabrillo DNS server) to my DNS server used at home

Nosmo RH9 Firewall

```
[root@nosmo root]# cat /etc/sysconfig/iptables
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*filter
:INPUT ACCEPT [4229:434875]
:FORWARD ACCEPT [1481:444016]
:OUTPUT ACCEPT [3340:350240]
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
# Generated by iptables-save v1.2.7a on Mon Jan 11 12:14:00 2010
*nat
:PREROUTING ACCEPT [8414:1265541]
:POSTROUTING ACCEPT [226:15381]
:OUTPUT ACCEPT [95:7826]
-A PREROUTING -d 207.62.187.53 -j DNAT --to-destination 192.168.0.1
-A POSTROUTING -o eth0 -j MASQUERADE
COMMIT
# Completed on Mon Jan 11 12:14:00 2010
[root@nosmo root]#
```

NAT all outgoing traffic to the public IP address on Nosmo. This gives CIS Lab hosts Internet access

Opus Firewall Brute force attacks

/var/log/wtmp and var/log/btmp

```
[root@opus ~]# lastb | grep "cool.nju.edu.cn" | head
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)
bind     ssh:notty    cool.nju.edu.cn  Sun Nov 30 06:35 - 06:35  (00:00)

[root@opus ~]# lastb | grep "cool.nju.edu.cn" | wc -l
3104
[root@opus ~]#
```

Shows break in attempt on 11/30/2008

/var/log/wtmp and var/log/btmp

```
[root@opus ~]# lastb | grep "Nov  2 17:45"
webadmin ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
webadmin ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
retsu    ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
retsu    ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
sbear    ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
sbear    ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
sky      ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
sky      ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
harvey   ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
harvey   ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
harvey   ssh:notty    211.96.97.179    Sun Nov  2 17:45 - 17:45  (00:00)
[root@opus ~]#

[root@opus ~]# lastb -i | grep "211.96.97.179" | wc -l
598
[root@opus ~]#
```

Shows break in attempt by 211.96.97.179 on 11/2/2008

/var/log/wtmp and var/log/btmp

```
[root@opus log]# lastb | sort | cut -f1 -d' ' | grep -v ^$ | uniq -c > bad
[root@opus log]# sort -g bad > bad.sort
[root@opus log]# [root@opus log]# cat  bad.sort | tail -50
    471 ftp
    472 public
    490 test
    490 tomcat
    498 user
    506 service
    508 mike
    508 username
    524 cyrus
    530 pgsql
    532 test1
    544 master
    554 linux
    554 toor
    576 paul
    584 support
    590 testuser
    604 irc
    610 test
    656 noc
    686 www
    690 postfix
    723 john
    734 testing
    738 adam
    746 alex
    754 info
    798 tester
    832 library
    935 guest
    990 admin
   1002 office
   1022 temp
   1070 ftpuser
   1138 webadmin
   1298 nagios
   1332 web
   1374 a
   1384 student
   1416 postgres
   1690 user
   1858 oracle
   1944 mysql
   2086 webmaste
   5324 test
  10803 root
  10824 admin
  18679 root
  24064 root
[root@opus log]#
```

Top 50 usernames used by the bad guys in 2008

/var/log/wtmp and var/log/btmp

22128 usernames used and failed

```
[root@opus log]# lastb | sort | cut -f1 -d' ' | grep -v ^$| uniq -c | wc -l
22128
[root@opus log]#
```

53117 failed root logins

```
[root@opus log]# lastb | grep root | wc -l
  54117
[root@opus log]#
```

Now you know why you need a strong password!

Impeding brute force attacks

Adds the current IP address to the recent list using the recent module

```
[rsimms@opus ~]$ cat /etc/sysconfig/iptables
< snipped >
# Impede brute force SSH dictionary attacks using the recent module (Rule added by RJS)
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --set –name SHBF
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j LOG --log-level info --log-prefix
"iptables brute force block: "
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j DROP
< snipped >
[rsimms@opus ~]$
```

http://kevin.vanzonneveld.net/techblog/article/block_brute_force_attacks_with_iptables/

Impeding brute force attacks

If four packets were sent from the same IP address in the last 60 seconds then log the packet.

```
[rsimms@opus ~]$ cat /etc/sysconfig/iptables
< snipped >
# Impede brute force SSH dictionary attacks using the recent module (Rule added by RJS)
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --set –name SHBF
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j LOG --log-level info --log-prefix
"iptables brute force block: "
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j DROP
< snipped >
[rsimms@opus ~]$
```

Impeding brute force attacks

If four packets were sent from the same IP address in the last 60 seconds then drop the packet.

```
[rsimms@opus ~]$ cat /etc/sysconfig/iptables
< snipped >
# Impede brute force SSH dictionary attacks using the recent module (Rule added by RJS)
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --set –name SHBF
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j LOG --log-level info --log-prefix
"iptables brute force block: "
-A RH-Firewall-1-INPUT -p tcp --dport 22 -m state --state NEW -m recent --update --
seconds 60 --hitcount 4 --rttl --name SSHBF -j DROP
< snipped >
[rsimms@opus ~]$
```

Impeding brute force attacks

```
[root@opus ~]# cat /var/log/messages | grep brute
< snipped >
Mar 14 11:32:56 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=202.113.16.118 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=49 ID=16335 DF PROTO=TCP SPT=34937 DPT=22 WINDOW=5840 RES=0x00 SYN
URGP=0
Mar 14 11:32:59 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=202.113.16.118 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=49 ID=16336 DF PROTO=TCP SPT=34937 DPT=22 WINDOW=5840 RES=0x00 SYN
URGP=0
Mar 14 11:33:05 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=202.113.16.118 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=49 ID=16337 DF PROTO=TCP SPT=34937 DPT=22 WINDOW=5840 RES=0x00 SYN
URGP=0
Mar 14 13:00:42 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=121.11.66.70 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=50 ID=18877 DF PROTO=TCP SPT=14752 DPT=22 WINDOW=5792 RES=0x00 SYN
URGP=0
Mar 14 13:00:45 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=121.11.66.70 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=50 ID=18879 DF PROTO=TCP SPT=14752 DPT=22 WINDOW=5792 RES=0x00 SYN
URGP=0
Mar 14 13:00:51 Opus kernel: iptables brute force block: IN=eth0 OUT=
MAC=00:50:56:90:7f:d8:00:22:55:97:10:0f:08:00 SRC=121.11.66.70 DST=207.62.186.9 LEN=60
TOS=0x00 PREC=0x00 TTL=50 ID=18881 DF PROTO=TCP SPT=14752 DPT=22 WINDOW=5792 RES=0x00 SYN
URGP=0
Mar 14 16:25:58 Opus kernel: iptables brute force block: IN=eth0 OUT=
< snipped >
```

Impeding brute force attacks

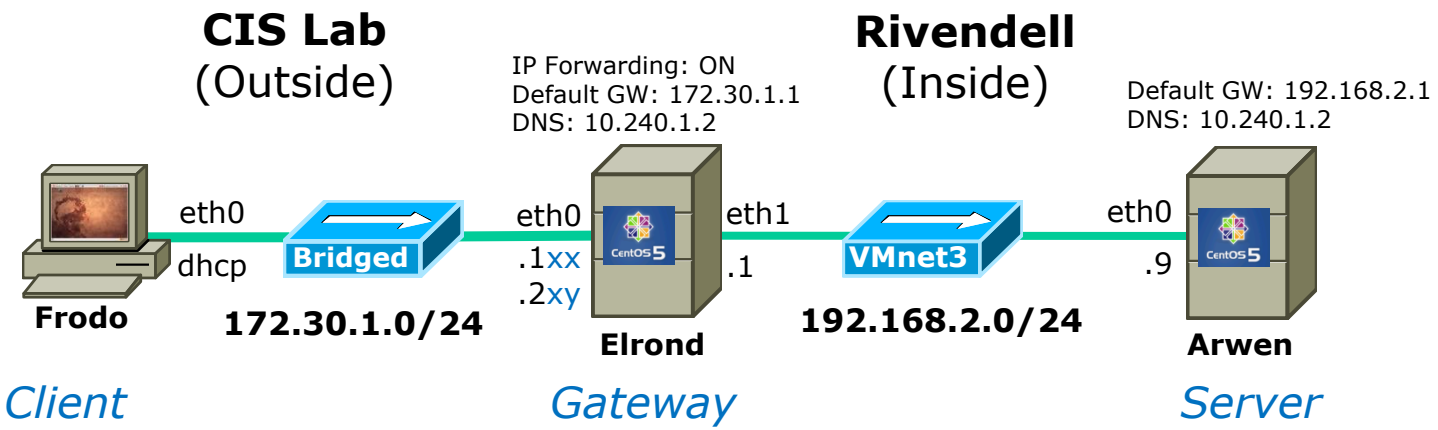
Recent dictionary attacks

From logwatch report:

```
Failed logins from:
    10.64.25.2: 1 time
    71.198.220.114 (c-71-198-220-114.hsd1.ca.comcast.net): 1 time
    74.220.66.39 (dsl-74-220-66-39.dhcp.cruzio.com): 1 time
    81.93.193.216 (credinfo.hu): 2 times
    95.18.14.156 (156.14.18.95.dynamic.jazztel.es): 1 time
    169.233.218.248 (dhcp-218-248.cruznet.ucsc.edu): 1 time
    180.153.127.111: 2 times

[root@opus ~]# lastb | grep 180.153.127.111
root     ssh:notty    180.153.127.111  Fri Nov 18 11:35 - 11:35  (00:00)
db2inst1 ssh:notty    180.153.127.111  Fri Nov 18 11:35 - 11:35  (00:00)
db2inst1 ssh:notty    180.153.127.111  Fri Nov 18 11:35 - 11:35  (00:00)
root     ssh:notty    180.153.127.111  Fri Nov 18 11:34 - 11:34  (00:00)
root     ssh:notty    180.153.127.111  Fri Oct  7 13:24 - 13:24  (00:00)
db2inst1 ssh:notty    180.153.127.111  Fri Oct  7 13:24 - 13:24  (00:00)
db2inst1 ssh:notty    180.153.127.111  Fri Oct  7 13:24 - 13:24  (00:00)
root     ssh:notty    180.153.127.111  Fri Oct  7 13:24 - 13:24  (00:00)
[root@opus ~]#
```

Lab 5 firewall

Firewall and NAT settings for Lab 5



Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  192.168.2.0/24       192.168.2.1         state NEW
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
LOG        all  --  0.0.0.0/0            0.0.0.0/0           LOG flags 0 level 6 prefix
`iptables INPUT: '

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  192.168.2.0/24       0.0.0.0/0           state NEW
ACCEPT     tcp  --  0.0.0.0/0            192.168.2.9         state NEW tcp dpt:23
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
LOG        all  --  0.0.0.0/0            0.0.0.0/0           LOG flags 0 level 6 prefix
`iptables FORWARD: '

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state
NEW,RELATED,ESTABLISHED
```

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond ~]# iptables -L -n -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
DNAT       all  --  0.0.0.0/0            172.30.1.200        to:192.168.2.9

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       all  --  192.168.2.9          0.0.0.0/0           to:172.30.1.200
SNAT       all  --  192.168.2.0/24       0.0.0.0/0           to:172.30.1.121

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

Note, using classroom addresses for this example

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Firewall and NAT settings for Lab 5

Elrond

Standard NAT chains

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Standard filter chains

Using lab addresses for this example

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Policy settings which are used if no rules on the chain apply

Firewall and NAT settings for Lab 5

Elrond

Forward any packets to 172.30.4.122 to 192.168.2.9

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Firewall and NAT settings for Lab 5

Elrond

Make outgoing packets from 192.168.2.9 appear as if they came from 172.30.4.122 (NAT)

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Firewall and NAT settings for Lab 5

Elrond

Make outgoing packets from private 192.168.2.0/24 network appear as if they came from 172.30.4.121 (NAT)

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Allow incoming ongoing traffic based on previous new connections that were allowed

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Log any input traffic that was not filtered out by rules above

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Forward all new outgoing connections from hosts on the private network

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Forward all traffic going to 192.168.2.9 port 23 (the Telnet server)

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Forward ongoing traffic based on previous new connections that were allowed

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Log any traffic that is about to be dropped (this is the last rule on the chain before policy get applied)

Firewall and NAT settings for Lab 5

Elrond

```
[root@elrond sysconfig]# cat iptables.lab5
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*nat
:PREROUTING ACCEPT [376:36875]
:POSTROUTING ACCEPT [74:4747]
:OUTPUT ACCEPT [60:3780]
-A PREROUTING -d 172.30.4.122 -i eth0 -j DNAT --to-destination 192.168.2.9
-A POSTROUTING -s 192.168.2.9 -o eth0 -j SNAT --to-source 172.30.4.122
-A POSTROUTING -s 192.168.2.0/255.255.255.0 -o eth0 -j SNAT --to-source 172.30.4.121
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
# Generated by iptables-save v1.3.5 on Sun Mar 14 16:08:44 2010
*filter
:INPUT DROP [313:33120]
:FORWARD DROP [9:756]
:OUTPUT DROP [0:0]
-A INPUT -s 192.168.2.0/255.255.255.0 -d 192.168.2.1 -i eth1 -m state --state NEW -j
ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j LOG --log-prefix "iptables INPUT: " --log-level 6
-A FORWARD -s 192.168.2.0/255.255.255.0 -m state --state NEW -j ACCEPT
-A FORWARD -d 192.168.2.9 -p tcp -m state --state NEW -m tcp --dport 23 -j ACCEPT
-A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -j LOG --log-prefix "iptables FORWARD: " --log-level 6
-A OUTPUT -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
COMMIT
# Completed on Sun Mar 14 16:08:44 2010
[root@elrond sysconfig]#
```

Allow all outgoing traffic

Firewall operations

Managing Red Hat Firewall

Backup the permanent settings

```
[root@elrond ~]# cp /etc/sysconfig/iptables /etc/sysconfig/iptables.bak
```

Save the current firewall and NAT settings for use at next reboot

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

Start using the rules saved in /etc/sysconfig/iptables

```
[root@elrond ~]# service iptables restart
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Setting chains to policy ACCEPT: nat filter      [  OK  ]
iptables: Unloading modules:                               [  OK  ]
iptables: Applying firewall rules:                         [  OK  ]
[root@elrond ~]#
```

Just like IP addresses and static routes we can set firewall and NAT rules temporarily (in memory) or permanently (in a file on the hard drive).

Firewall and SELinux

```
[root@celebrian ~]# iptables -L > baffling
[root@celebrian ~]# cat baffling
[root@celebrian ~]#
```

With SELinux in enforcing mode, iptables and iptables-save no longer will redirect stdout to a file!

There are several workarounds which can be found on the forum under the "Lab 3 iptables Issue" topic

Firewall and SELinux

```
[root@celebrian ~]# getenforce
Enforcing
[root@celebrian ~]# setenforce 0
[root@celebrian ~]# getenforce
Permissive
[root@celebrian ~]# iptables -L > baffling
[root@celebrian ~]# cat baffling
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state
RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp
dpt:ssh
< snipped >
[root@celebrian
```

With SELinux in permissive mode, iptables and iptables-save work as before

Netfilter

(iptables)

Netfilter

Netfilter

- Packet filtering (firewall)
- Port and Address translation (NAT*)
- Logging
- Other types of packet mangling
- Implemented by the iptables utility
- Replaces ipchains in older kernels (2.2 and earlier)

*Note, the term NAT can mean different things. Linux really does PAT which includes both address and port translation. This allows multiple private address to be concurrently translated to a single public IP address.

To do this we will use DNAT, SNAT actions or MASQUERADE actions.

Netfilter

Firewalls and Access Control Lists
A Firewall is a system that prevents unauthorized network communications to it, from it, and through it.

Netfilter

IPtables - Chains are grouped into tables:

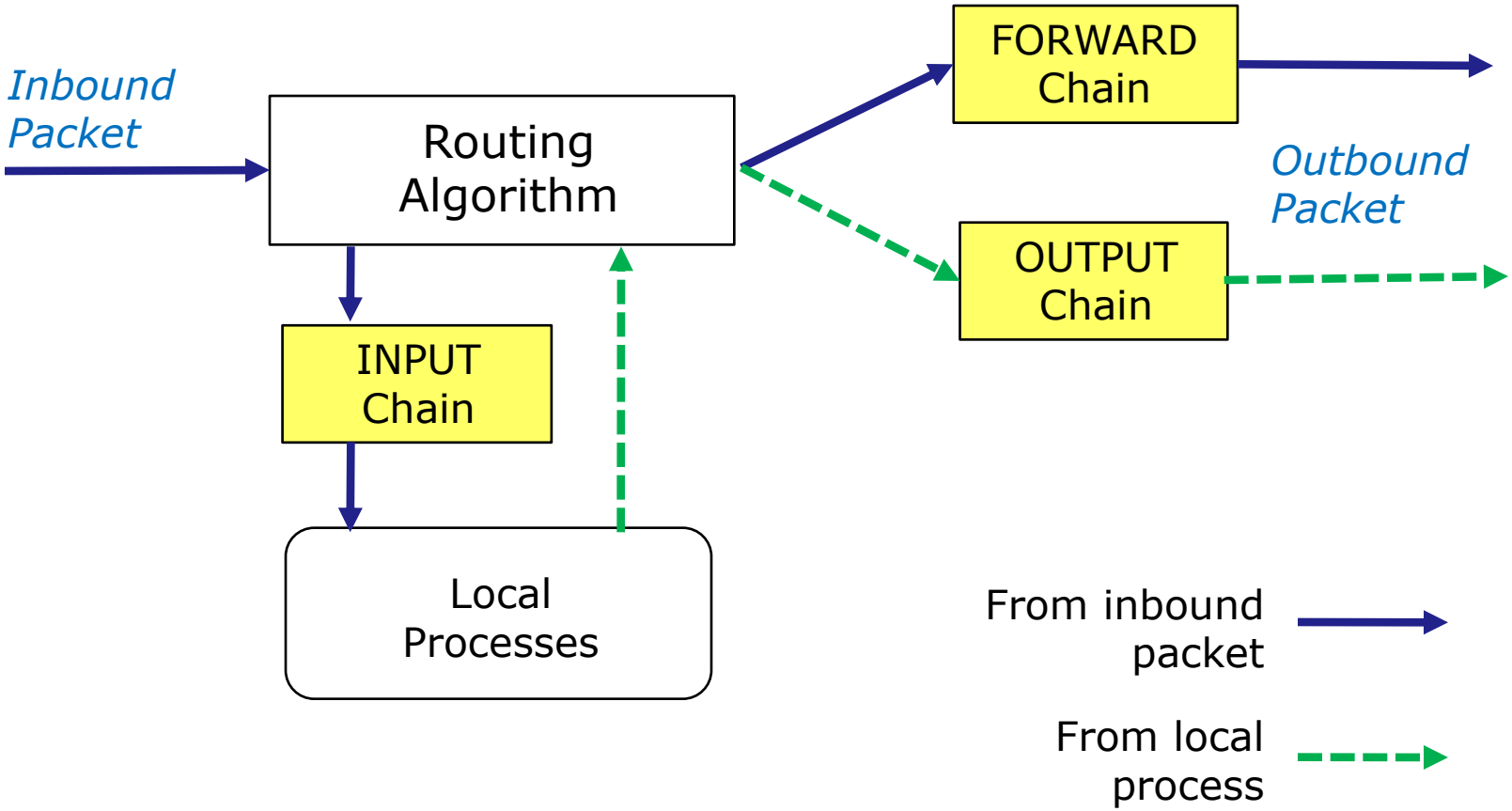
Filter chains:
- Input
- Output
- Forward

NAT chains:
- Prerouting
- Output
- Postrouting

Mangle chains:
- Prerouting
- Output

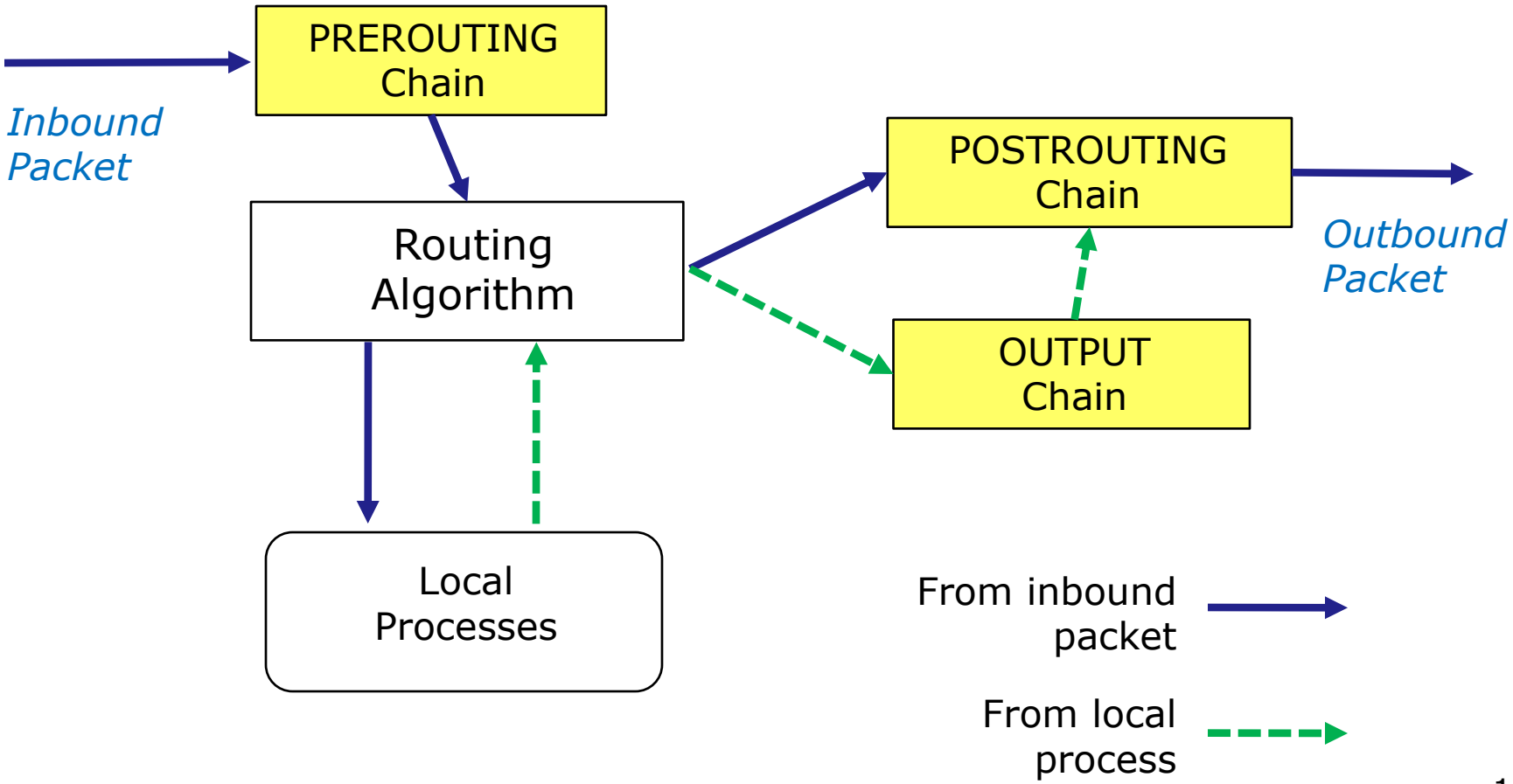
Netfilter

Filter table

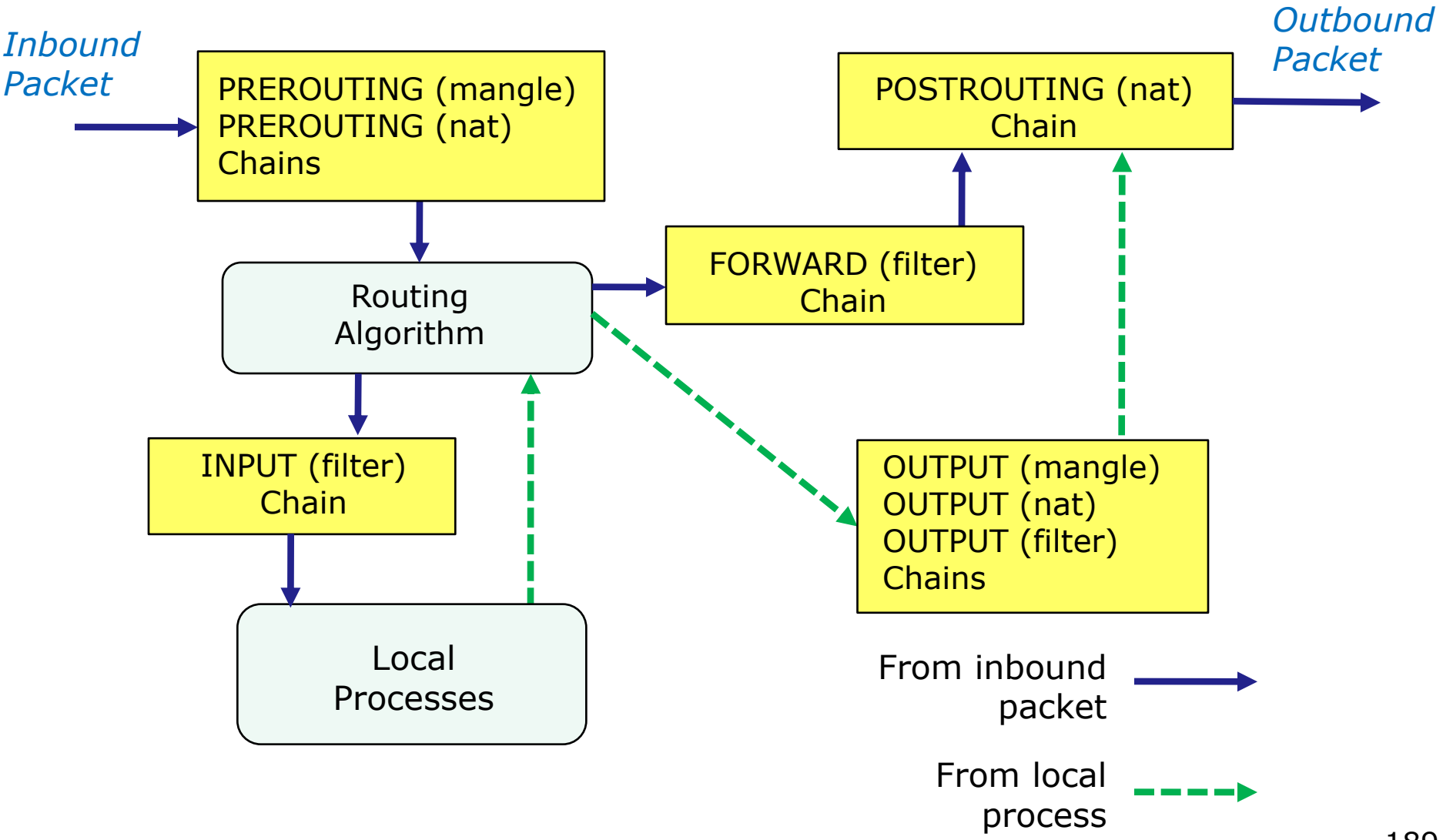


iptables

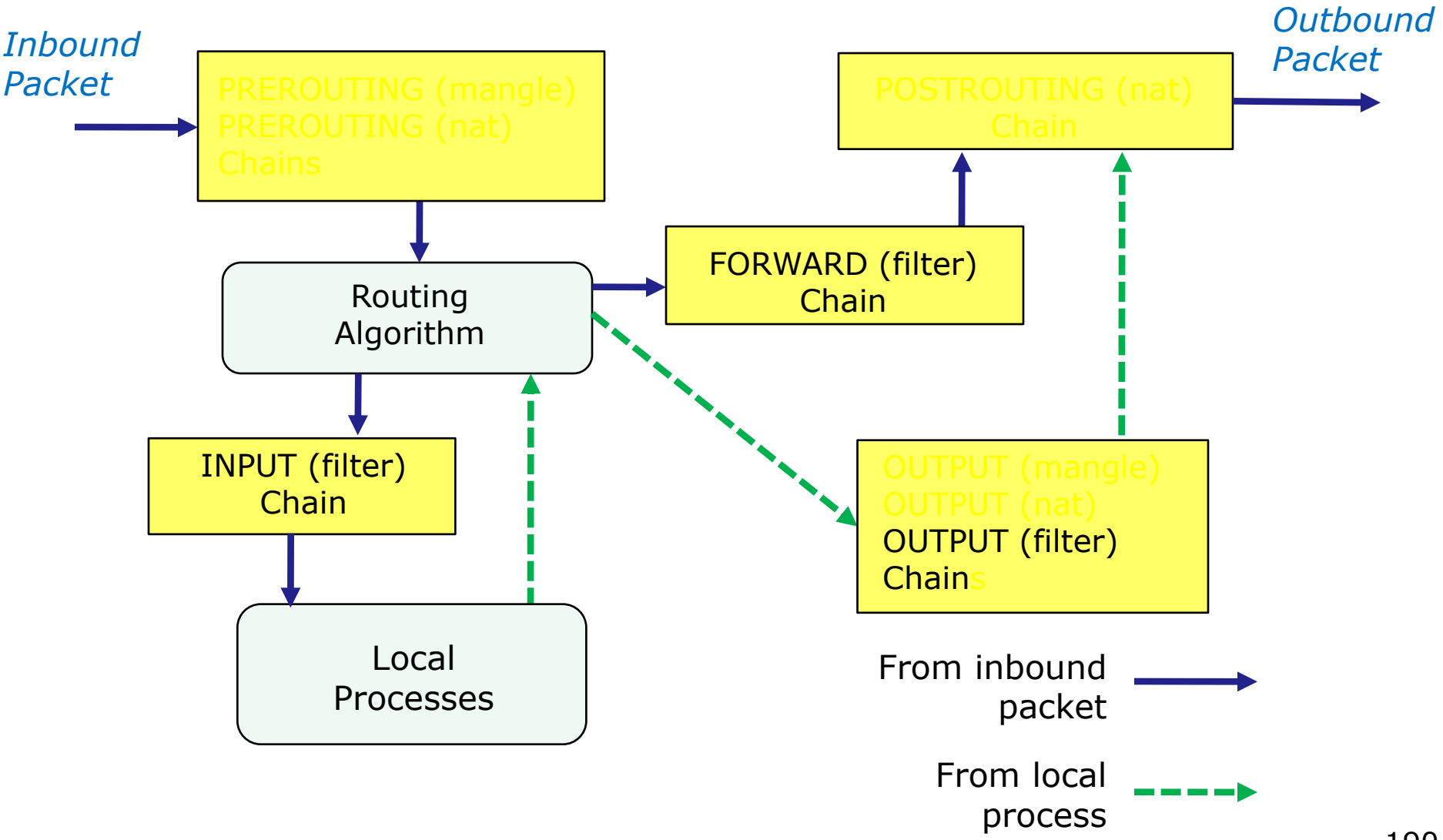
nat table



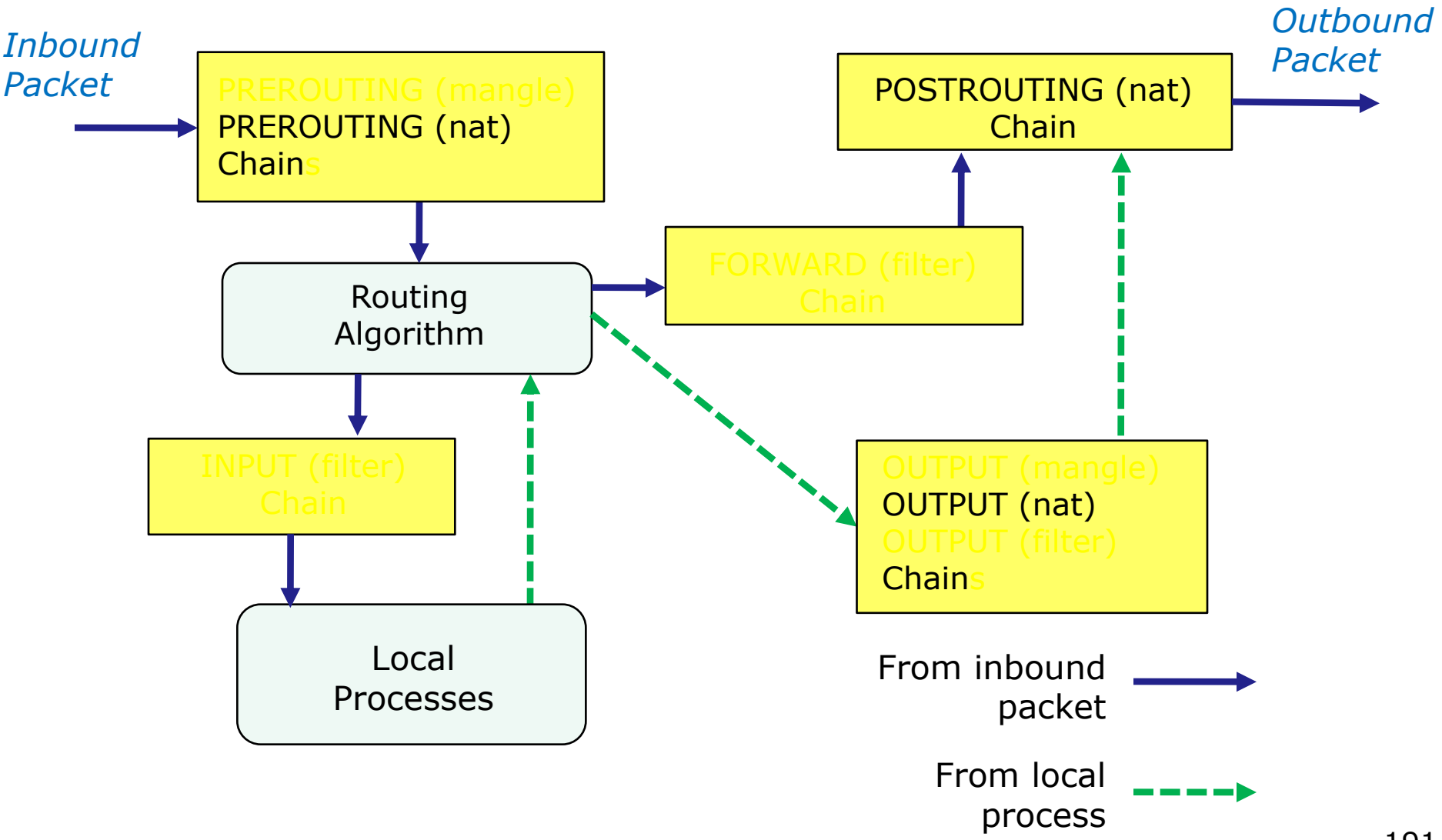
Netfilter – all tables and chains



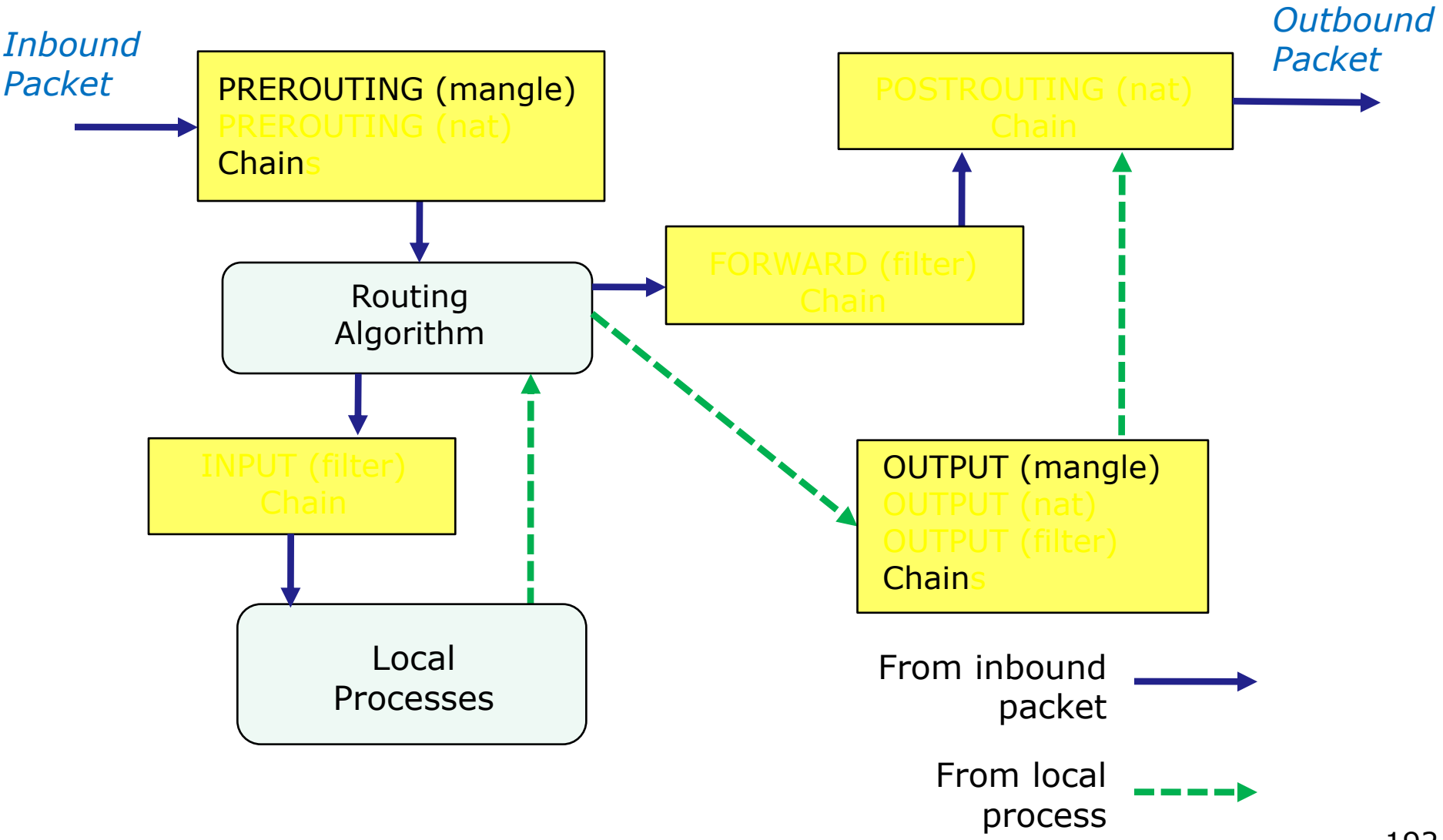
Netfilter – filter table chains



Netfilter – nat table chains



Netfilter – mangle table chains



Netfilter

iptables command syntax

iptables [-flags] [chain] [options [extensions]] [action]

Netfilter

Flags

-t table
-A append a rule
-D delete a rule
-F flush all rules for a specified chain
-I insert a rule at the specified position
-L list all rules
-P policy - the default chain rule
-R replace a rule

Netfilter

Options

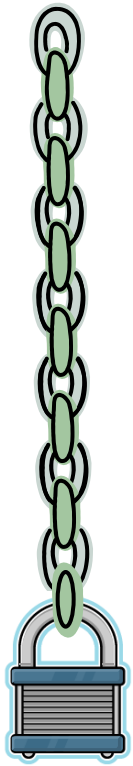
-d destination IP address (accepts CIDR and 0/0 as all)
-s source IP address (accepts CIDR and 0/0 as all)
-p protocol - any name listed in */etc/protocols*
-i the inbound interface
-o the outbound interface
-j the target action
-m extended matching module - has many extensions e.g.
state: --state NEW,ESTABLISHED,RELATED

Netfilter

Actions

ACCEPT
DROP
REJECT
LOG
DNAT
SNAT

Netfilter – chains



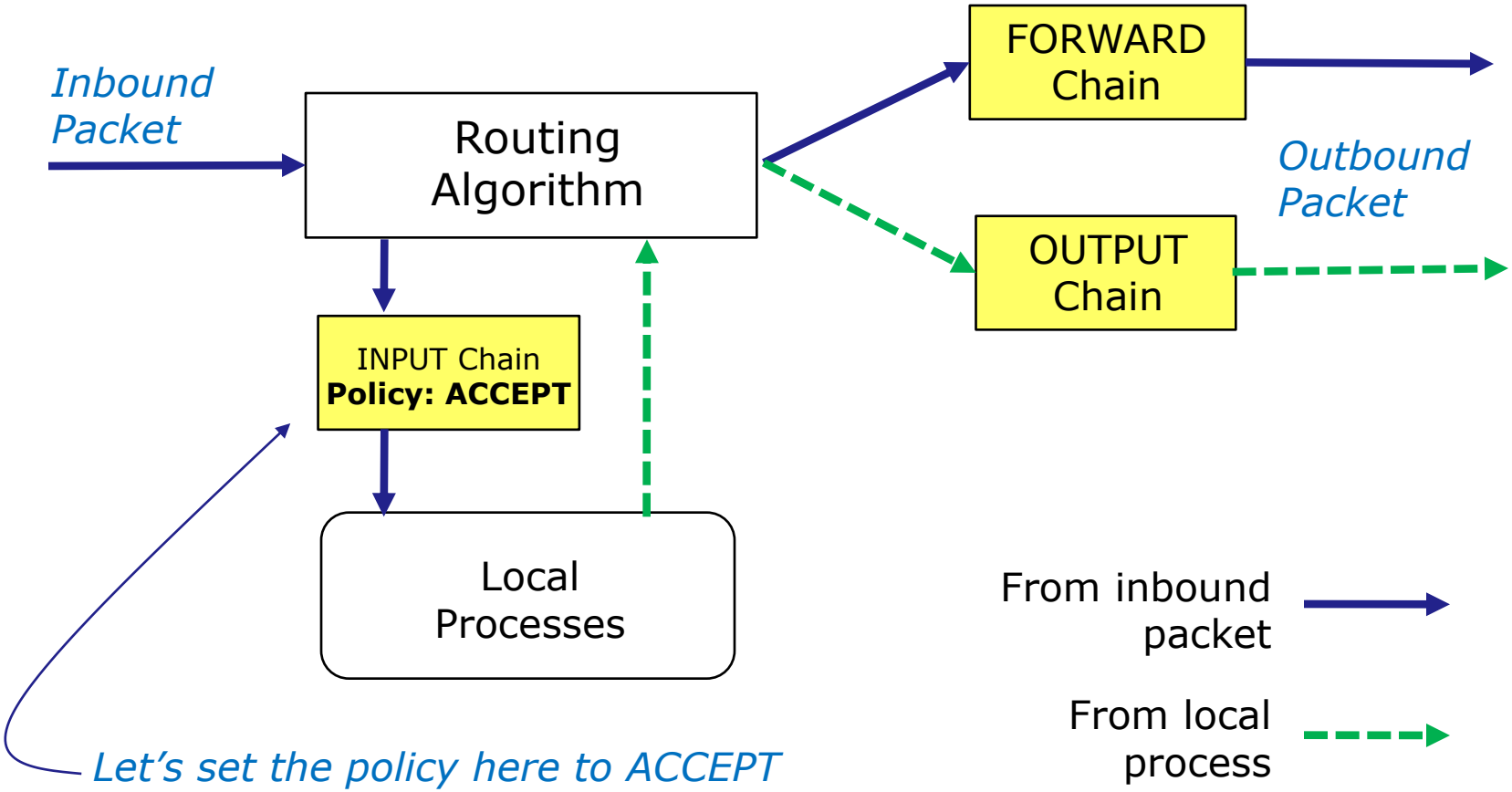
Rules

Policy – the action to take if you get through all the rules on the chain

Table: filter Chain: INPUT Policy: ACCEPT or DROP

Netfilter – examples

Filter table on Elrond



Netfilter – examples

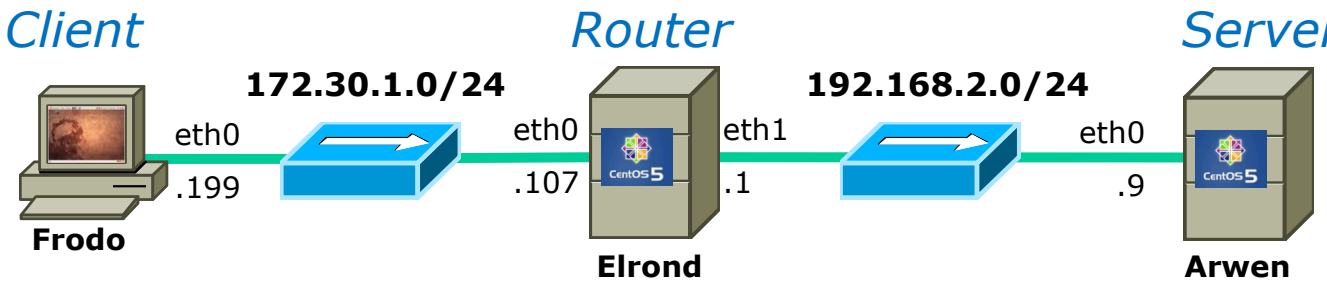
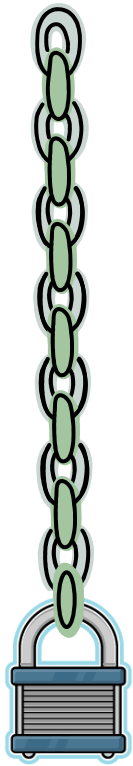


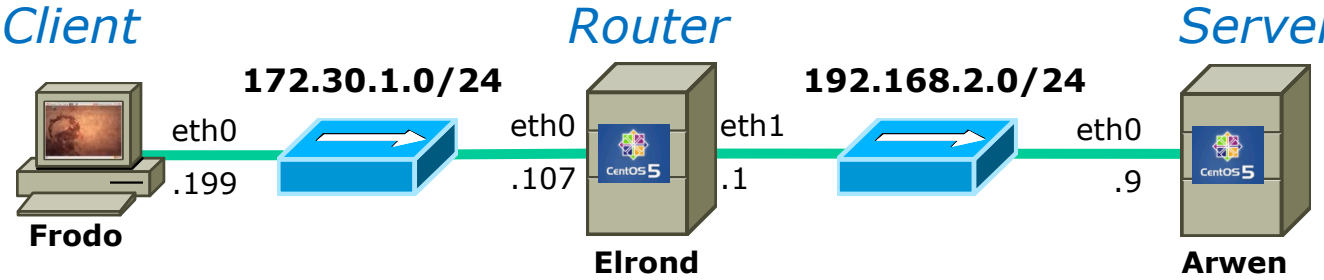
Table: filter
Chain: INPUT



No Rules

Chain Policy: ACCEPT

Netfilter – examples




```
[root@elrond ~]# iptables -F
[root@elrond ~]# iptables -X
[root@elrond ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

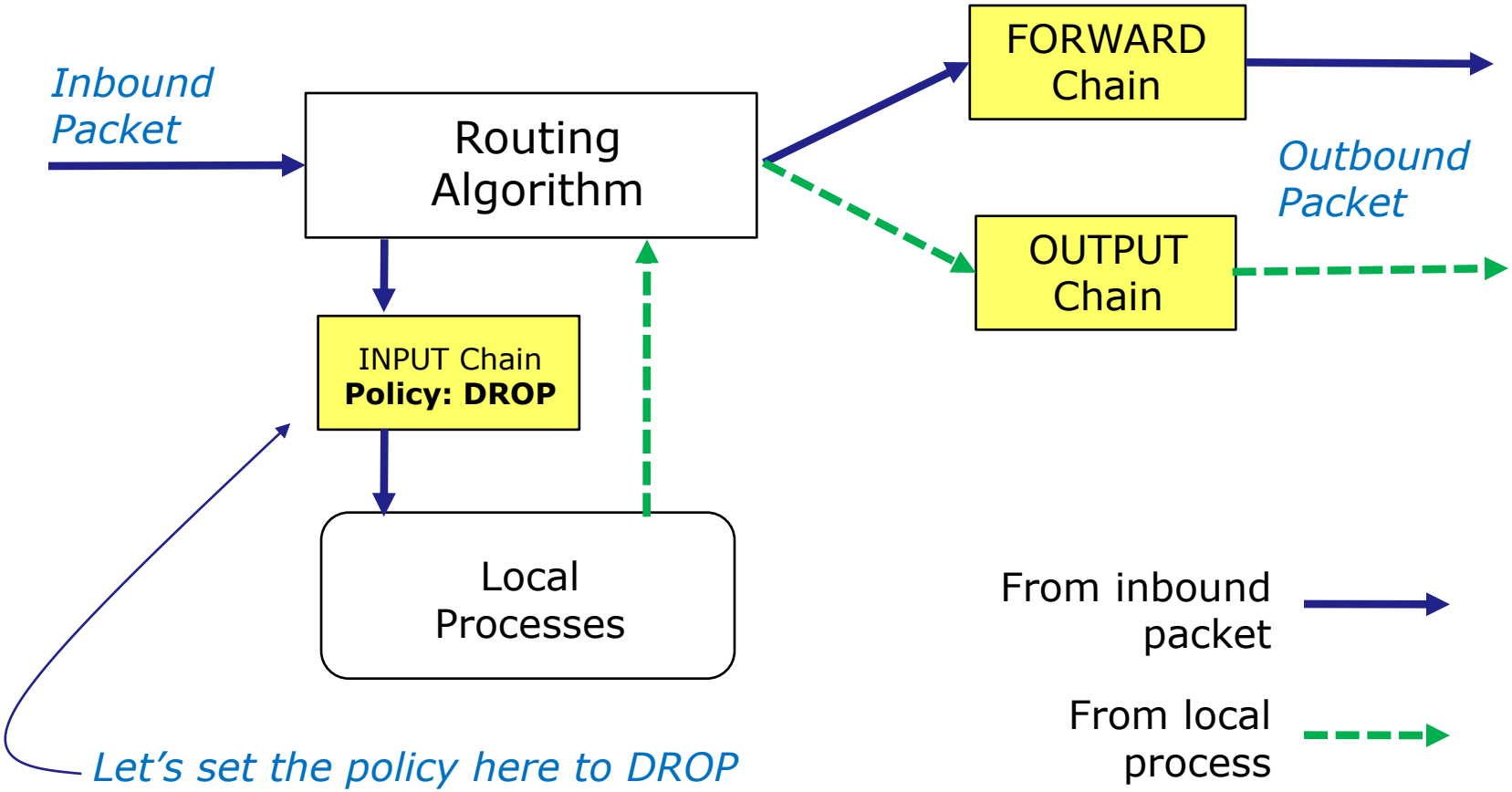
Flush filter chain rules and delete any custom chains.

INPUT chain policy is ACCEPT

```
root@frodo:~# ping -c 1 elrond
PING elrond (172.30.1.107) 56(84) bytes of data.
64 bytes from elrond (172.30.1.107): icmp_seq=1 ttl=64 time=0.803 ms
```

Netfilter – examples

Filter table on Elrond



Netfilter – examples

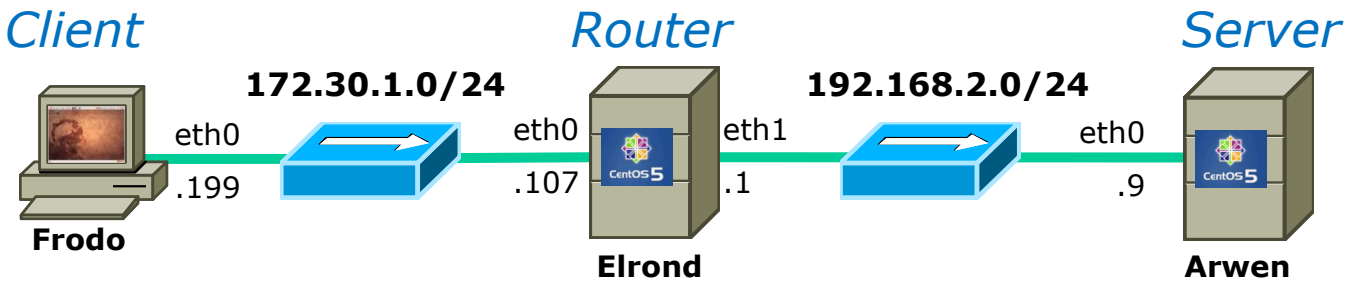
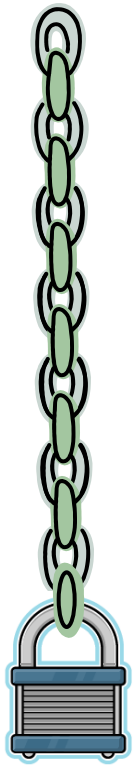


Table: filter
Chain: INPUT

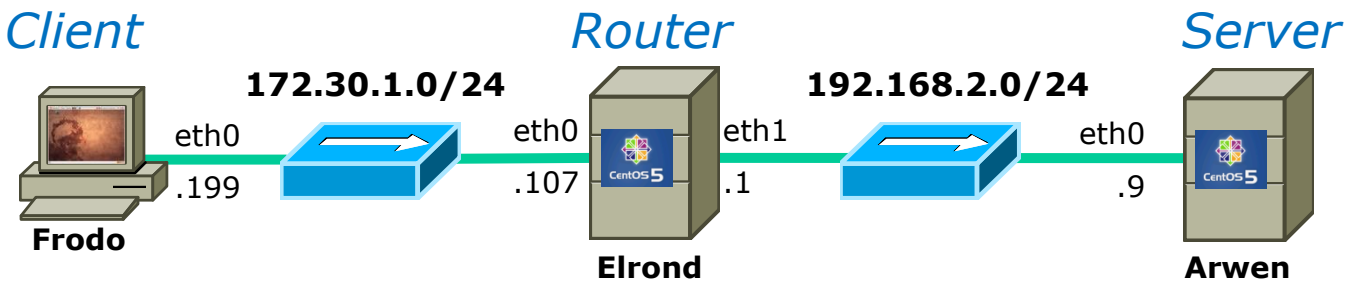


No Rules

Chain Policy: DROP

DROP everything else

Netfilter – examples

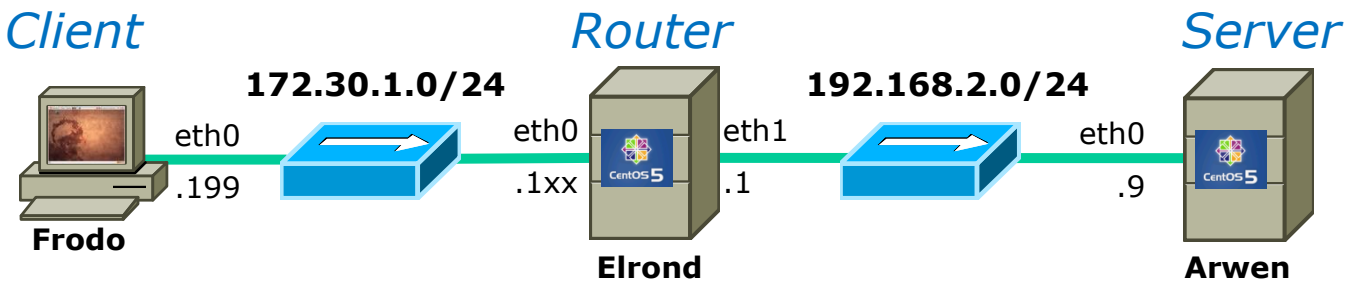



```
[root@elrond ~]# iptables -P INPUT DROP
[root@elrond ~]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

Frodo cannot ping Elrond now

```
root@frodo:~# ping -c 2 elrond
PING elrond (172.30.1.107) 56(84) bytes of data.

--- elrond ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 999ms
```

Netfilter – examples

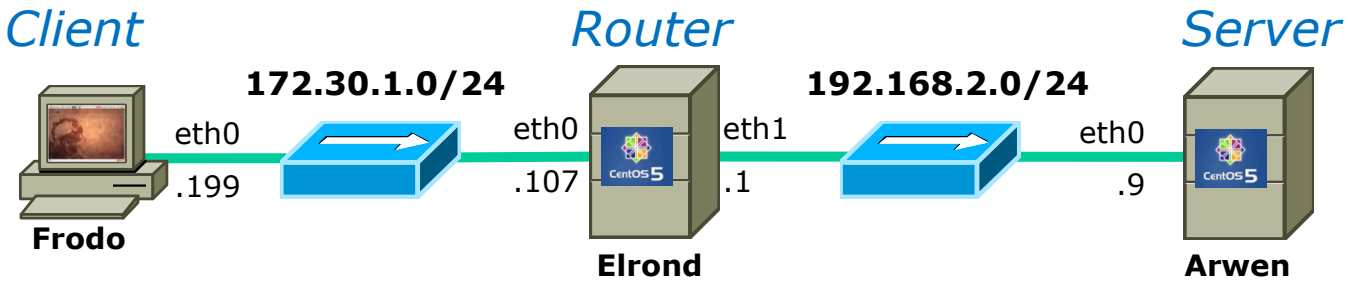


7	3.132262	172.30.4.199	172.30.4.107	ICMP	Echo (ping) request
10	4.132169	172.30.4.199	172.30.4.107	ICMP	Echo (ping) request
11	8.131620	Vmware_6f:53:d9	Vmware_4e:21:af	ARP	Who has 172.30.4.107? Tell 172.30.4.199
12	8.132788	Vmware_4e:21:af	Vmware_6f:53:d9	ARP	172.30.4.107 is at 00:0c:29:4e:21:af
13	10.119859	Vmware_4e:21:af	Vmware_30:16:94	ARP	Who has 172.30.4.1? Tell 172.30.4.107
14	10.119911	Vmware_30:16:94	Vmware_4e:21:af	ARP	172.30.4.1 is at 00:0c:29:30:16:94

Even though Frodo can no longer ping Elrond, Elrond will still respond to Frodo's ARP requests.

```
root@frodo:~# ping -c 2 elrond
PING elrond (172.30.1.107) 56(84) bytes of data.

--- elrond ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 999ms
```

Netfilter – iptables



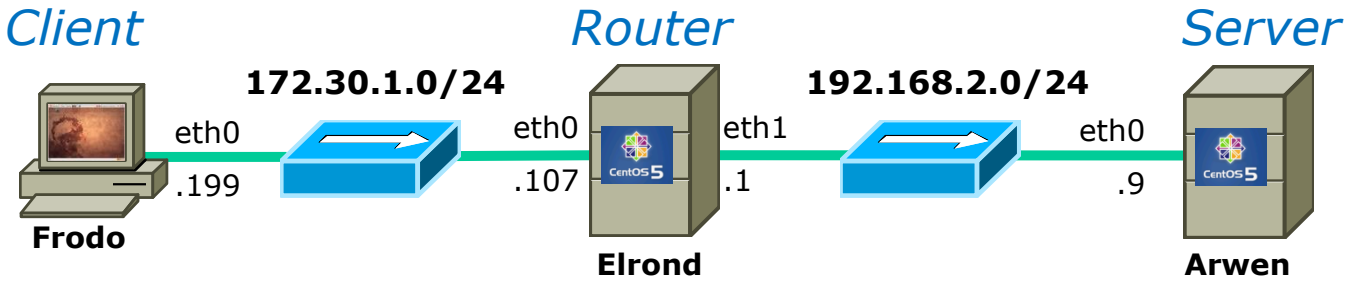

```
[root@elrond ~]# iptables -P INPUT DROP
[root@elrond ~]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

Elrond cannot ping Frodo either …

… because the returning echo responses get dropped by the INPUT chain policy

```
[root@elrond ~]# ping -c 2 frodo
PING frodo (172.30.1.199) 56(84) bytes of data.

--- frodo ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1002ms
```

Netfilter – examples



3 2.764346	172.30.4.107	172.30.4.199	ICMP	Echo (ping) request
4 2.764403	172.30.4.199	172.30.4.107	ICMP	Echo (ping) reply
5 4.142478	172.30.4.107	172.30.4.199	ICMP	Echo (ping) request
6 4.143043	172.30.4.199	172.30.4.107	ICMP	Echo (ping) reply
9 7.763088	Vmware_6f:53:d9	Vmware_4e:21:af	ARP	Who has 172.30.4.107? Tell 172.30.4.199
10 7.763496	Vmware_4e:21:af	Vmware_6f:53:d9	ARP	172.30.4.107 is at 00:0c:29:4e:21:af

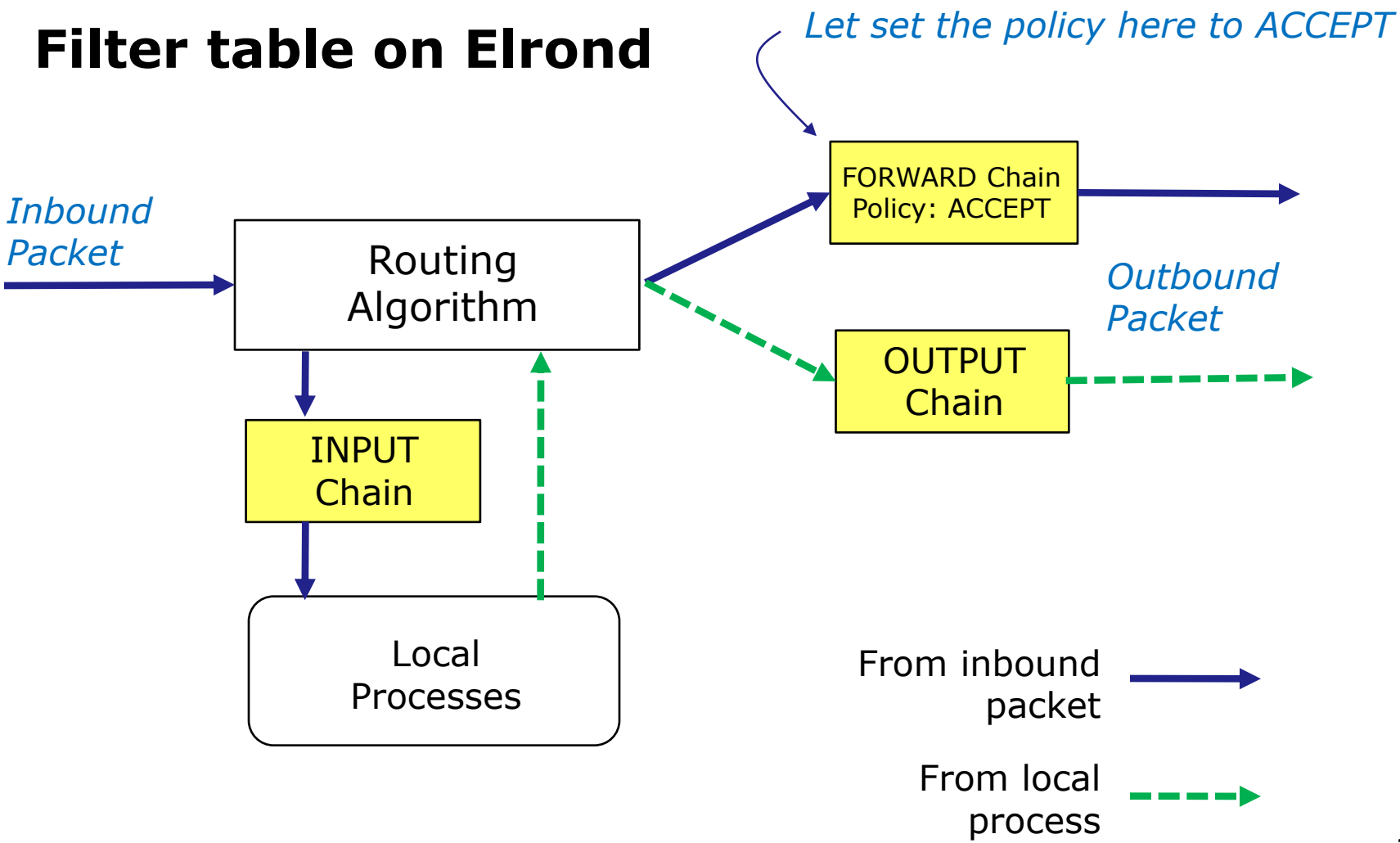
Note the ping requests get to Frodo and Frodo is responding, however the responses get dropped in Elrond's INPUT chain

```
[root@elrond ~]# ping -c 2 frodo
PING frodo (172.30.1.199) 56(84) bytes of data.

--- frodo ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1002ms
```

Table: filter Chain: FORWARD Policy: ACCEPT or DROP

Netfilter – examples

Filter table on Elrond



Netfilter – examples

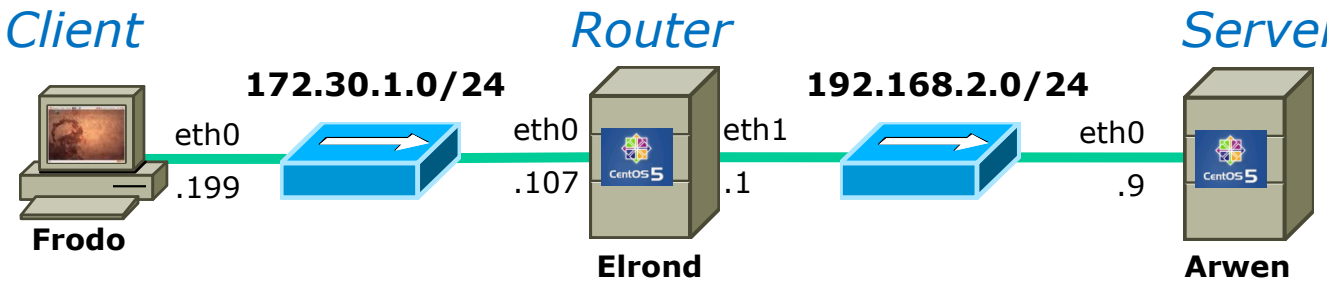
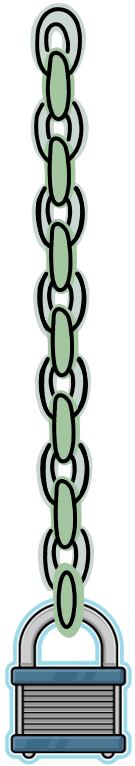


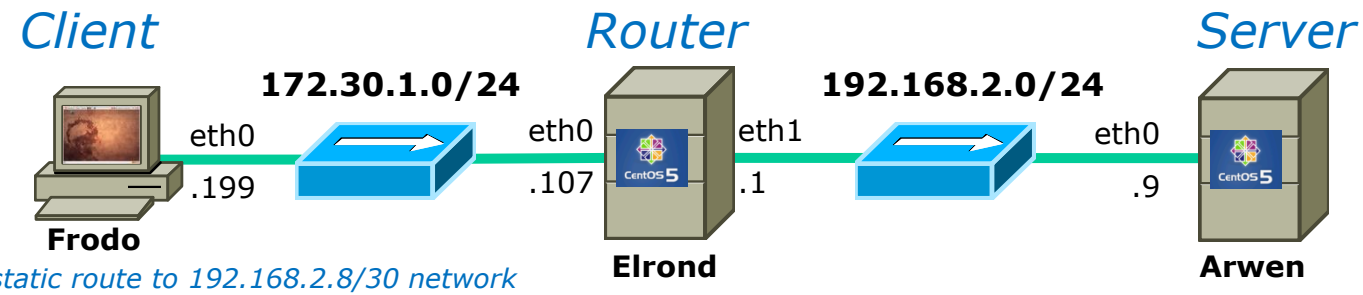
Table: filter
Chain: FORWARD



No Rules

Chain Policy: ACCEPT

Netfilter – examples



Frodo has static route to 192.168.2.8/30 network

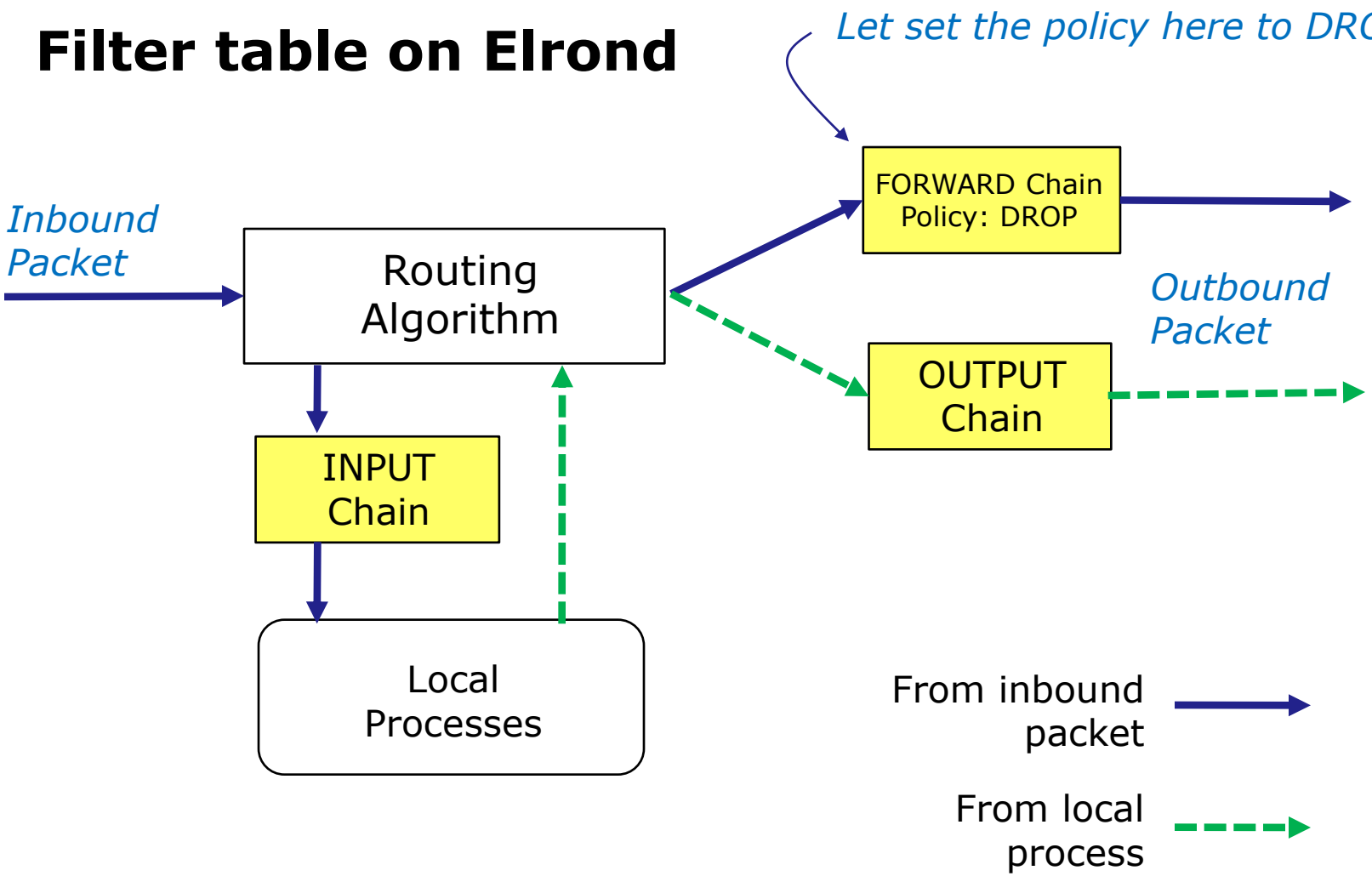
```
[root@elrond ~]# iptables -P FORWARD ACCEPT
[root@elrond ~]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

Frodo can ping via Elrond to Arwen because Elrond's FORWARD chain's policy is ACCEPT

```
root@frodo:~# ping -c 2 arwen
PING arwen (192.168.2.9) 56(84) bytes of data.
64 bytes from arwen (192.168.2.9): icmp_seq=1 ttl=63 time=5.38 ms
64 bytes from arwen (192.168.2.9): icmp_seq=2 ttl=63 time=1.13 ms

--- arwen ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
```

Netfilter – examples

Filter table on Elrond



Netfilter – examples

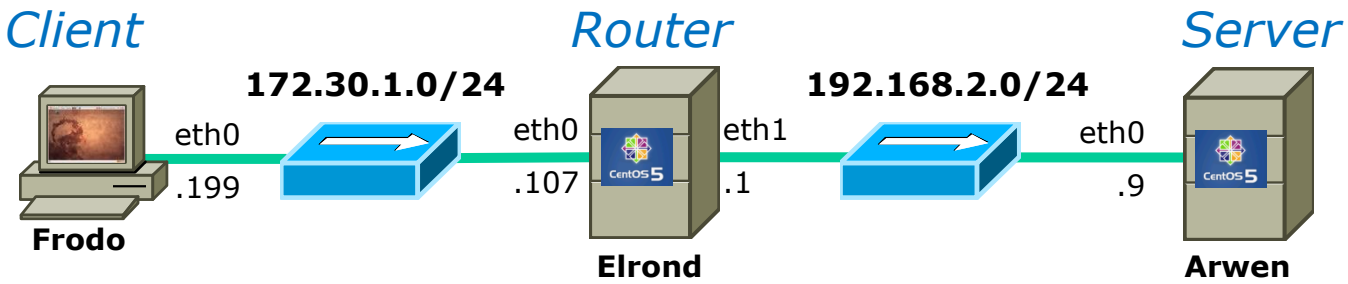
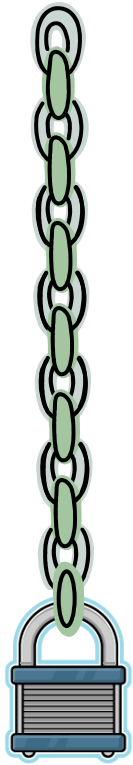


Table: filter
Chain: FORWARD

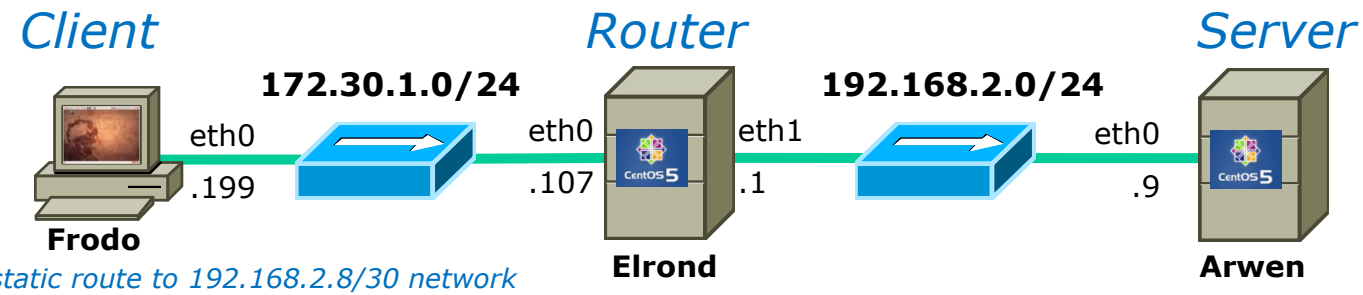


No Rules

Chain Policy: DROP

DROP everything else

Netfilter – examples



Frodo has static route to 192.168.2.8/30 network

```
[root@elrond ~]# iptables -P FORWARD DROP
[root@elrond ~]# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

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

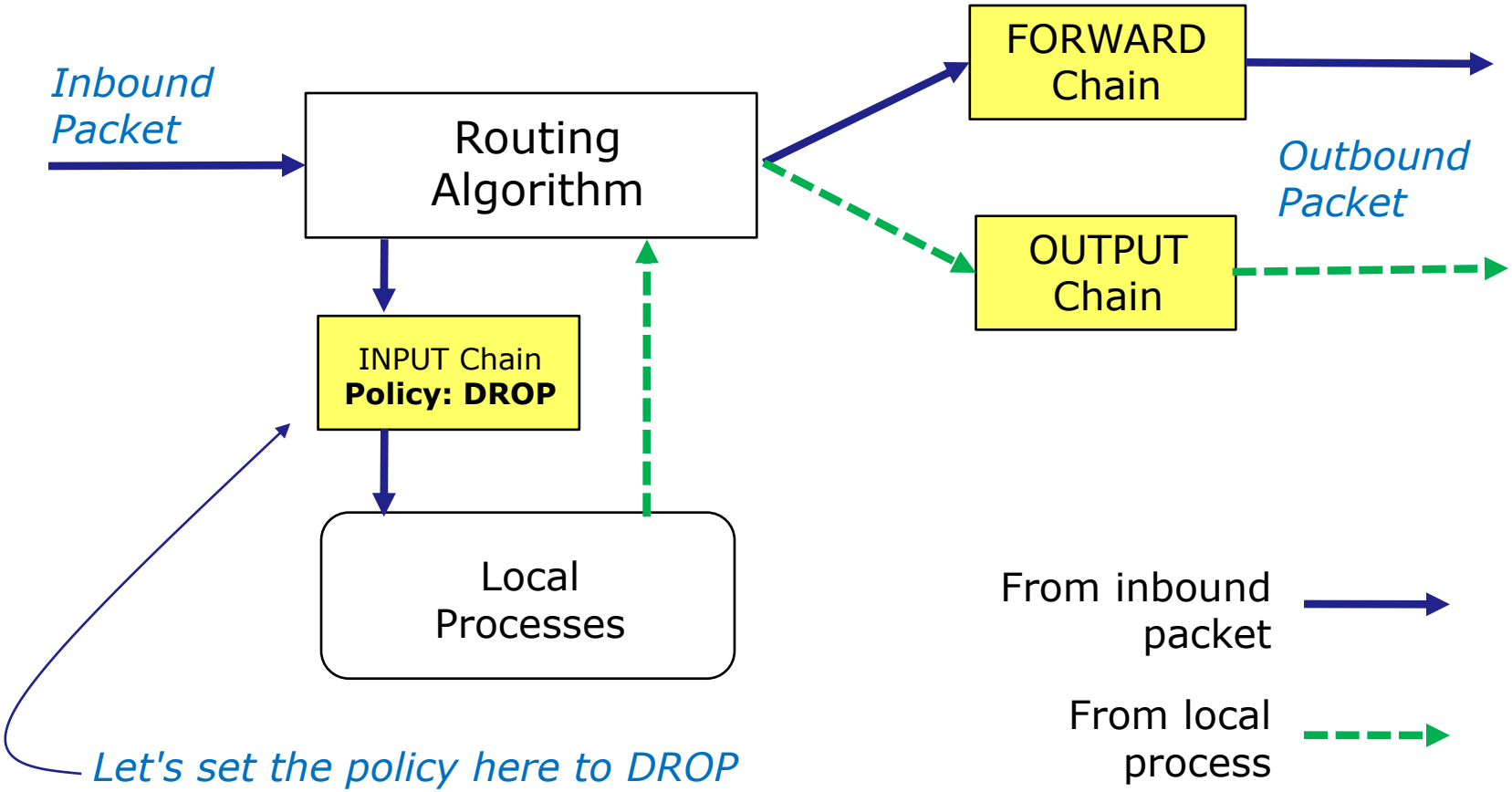
Frodo cannot ping Arwen via Elrond because Elrond's FORWARD chain policy is DROP

```
root@frodo:~# ping -c 2 arwen
PING arwen (192.168.2.9) 56(84) bytes of data.

--- arwen ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1004ms
```

Table: filter
Chain: INPUT
IP address rules

Netfilter – examples

Filter table on Elrond



Netfilter – examples

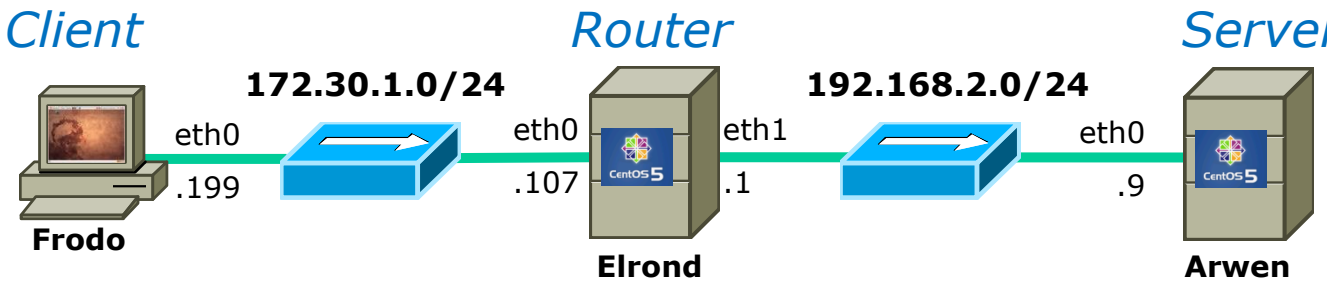
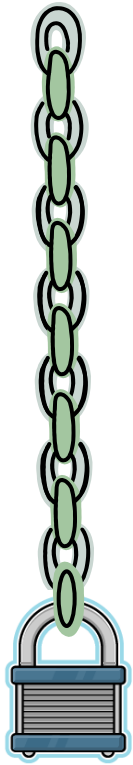


Table: filter
Chain: INPUT



Chain Rules:
-s 172.30.1.199/32 -j REJECT

Reject anything from Frodo

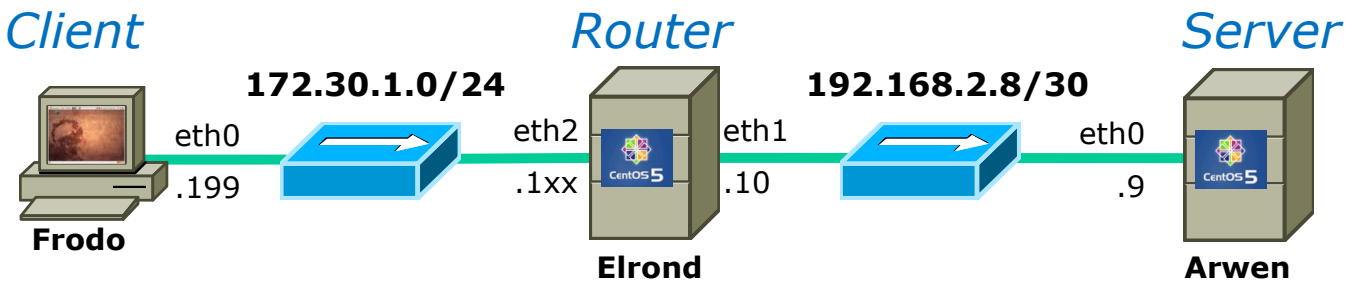
-s 192.168.0.0/16 –j ACCEPT

Accept all packets from 192.168.x.x

Chain Policy: DROP

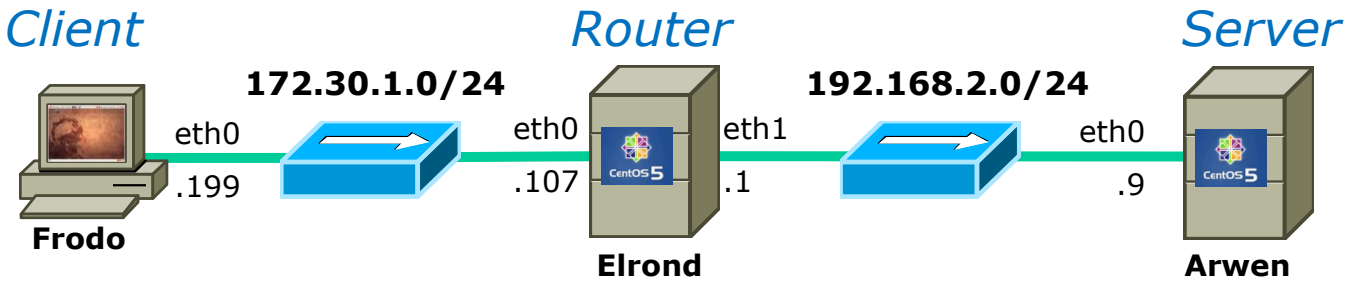
DROP everything else

Netfilter – examples




```
[root@elrond ~]# iptables -F
[root@elrond ~]# iptables -A INPUT -s 172.30.1.199/32 -j REJECT
[root@elrond ~]# iptables -A INPUT -s 192.168.0.0/16 -j ACCEPT
[root@elrond ~]# iptables -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
REJECT     all  --  172.30.1.199         0.0.0.0/0           reject-with
icmp-port-unreachable
ACCEPT     all  --  192.168.0.0/16       0.0.0.0/0

Chain FORWARD (policy DROP)
target     prot opt source               destination

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

Netfilter – examples

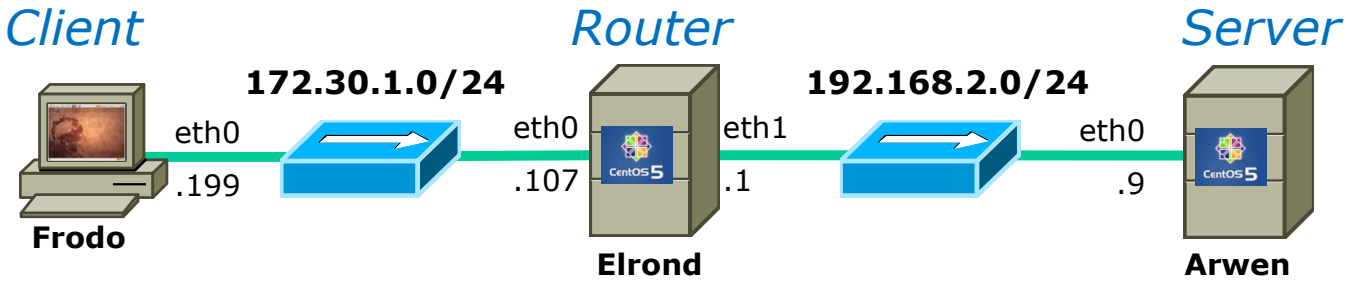



```
Chain INPUT (policy DROP)
target     prot opt source               destination
REJECT     all  --  172.30.1.199         0.0.0.0/0           reject-with
icmp-port-unreachable
ACCEPT     all  --  192.168.0.0/16       0.0.0.0/0
```

```
root@frodo:~# ping -c 2 elrond
PING elrond (172.30.1.107) 56(84) bytes of data.
From elrond (172.30.1.107) icmp_seq=1 Destination Port Unreachable
From elrond (172.30.1.107) icmp_seq=2 Destination Port Unreachable

--- elrond ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 1004ms

root@frodo:~#
```

Ping from Frodo to Elrond fails with port unreachable

Netfilter – examples

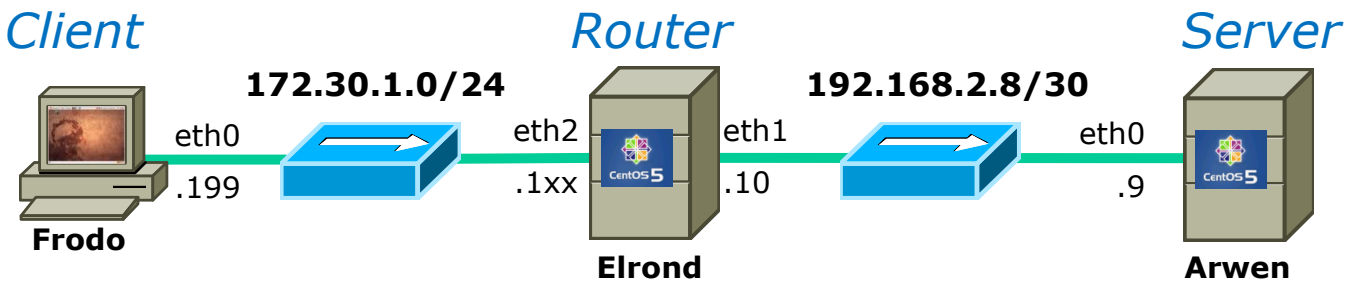



```
Chain INPUT (policy DROP)
target     prot opt source               destination
REJECT     all  --  172.30.1.199         0.0.0.0/0           reject-with
icmp-port-unreachable
ACCEPT     all  --  192.168.0.0/16       0.0.0.0/0
```

```
[root@arwen ~]# ping -c2 elrond
PING elrond (192.168.2.10) 56(84) bytes of data.
64 bytes from elrond (192.168.2.10): icmp_seq=1 ttl=64 time=5.86 ms
64 bytes from elrond (192.168.2.10): icmp_seq=2 ttl=64 time=1.74 ms

--- elrond ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 1.748/3.807/5.867/2.060 ms
[root@arwen ~]#
```

Ping from Arwen to Elrond succeeds

Netfilter – examples




```
Chain INPUT (policy DROP)
target     prot opt source               destination
REJECT     all  --  172.30.1.199         0.0.0.0/0           reject-with
icmp-port-unreachable
ACCEPT     all  --  192.168.0.0/16       0.0.0.0/0
```

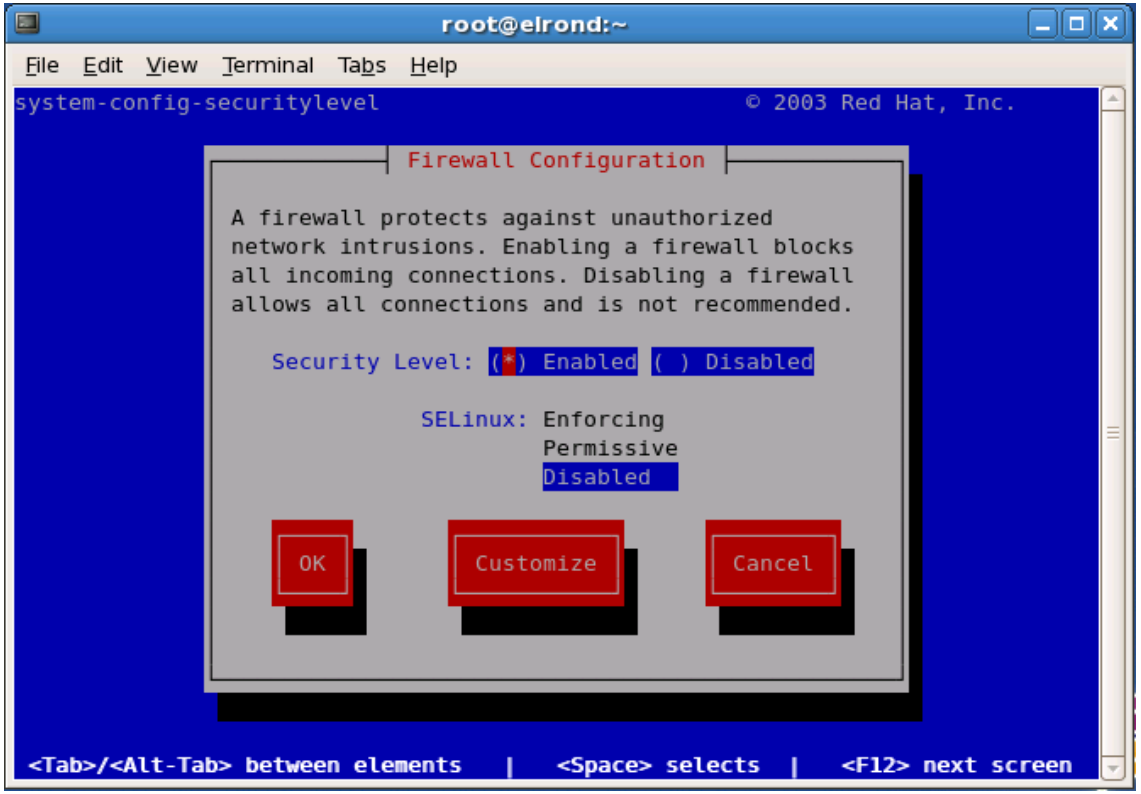
```
[root@nosmo root]# ping -c 2 elrond
PING elrond (172.30.1.107) 56(84) bytes of data.

--- elrond ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1012ms

[root@nosmo root]#
```

Ping from Nosmo (172.30.1.1) to Elrond fails, timing out without any error messages

Lab 5

lokkit

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



Settings kept in: /etc/sysconfig/system-config-securitylevel

Lokkit command for enabling and disabling firewall and SELinux settings.

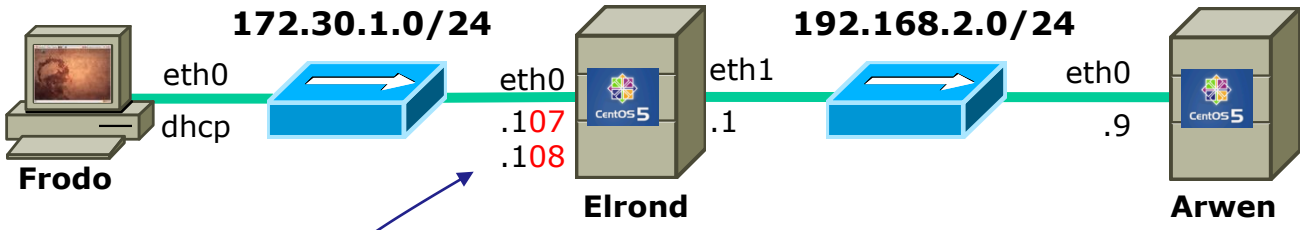
*Beware, this tool can overwrite any firewall settings you have made with **iptables** commands*

Frodo is an outside host that is allowed to use the Telnet Server on Arwen

Shire
(Outside)

Rivendell
(Inside)

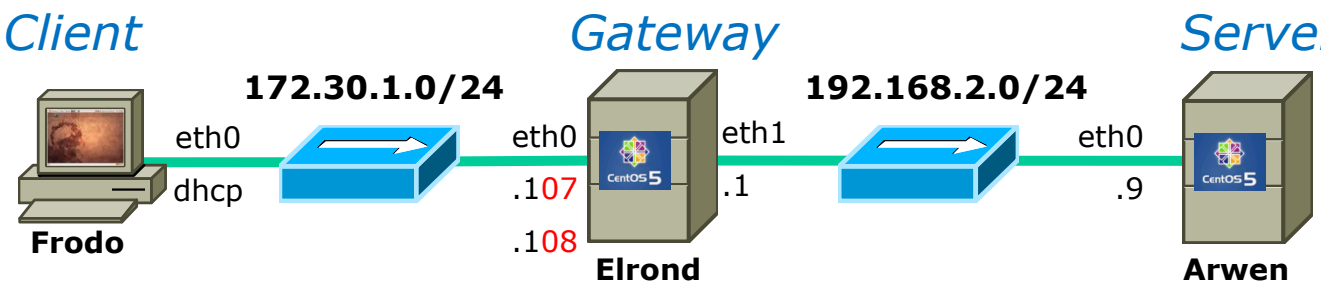
Telnet server is installed on Arwen



These address are for example only. Only use the static IP addresses designated for you station!

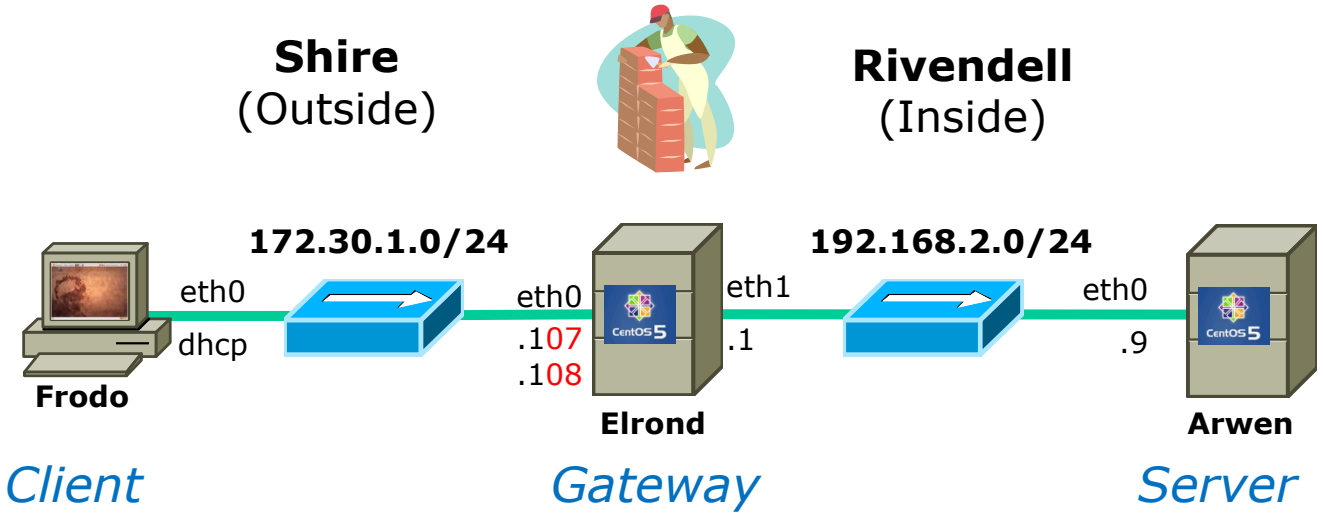
We create a firewall on Elrond, the gateway

NAT is also configured here so Rivendell hosts have Internet access



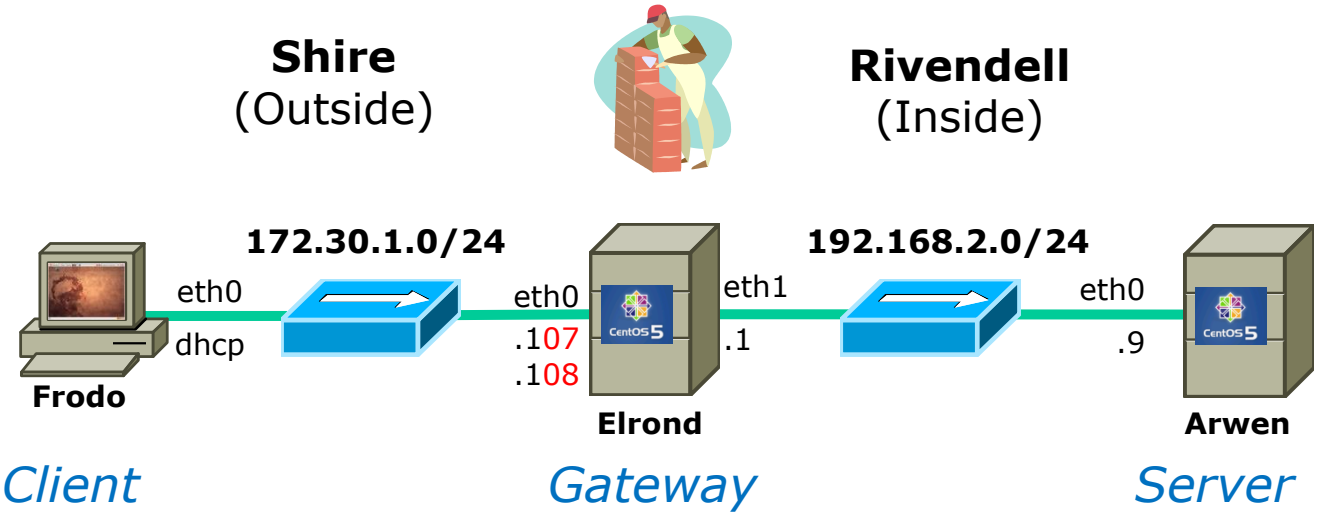

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP

iptables -A FORWARD -s 192.168.2.0/24 -d 0/0 -m state --state NEW -j ACCEPT
iptables -A FORWARD -s 0/0 -d 192.168.2.9 -m state --state NEW -p tcp --dport 23 -j ACCEPT
iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -i eth1 -s 192.168.2.0/24 -d 192.168.2.1 -m state --state NEW -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT


ifconfig eth0:1 172.30.1.108 netmask 255.255.255.0 broadcast 172.30.1.255

iptables -t nat -A PREROUTING -i eth0 -d 172.30.1.108 -j DNAT --to-destination 192.168.2.9
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.9 -j SNAT --to-source 172.30.1.108
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.0/24 -j SNAT --to-source 172.30.1.107

iptables -A INPUT -j LOG --log-level info --log-prefix "iptables INPUT: "
iptables -A FORWARD -j LOG --log-level info --log-prefix "iptables FORWARD: "
```

Note: Your Elrond static IP will be based on the station you use



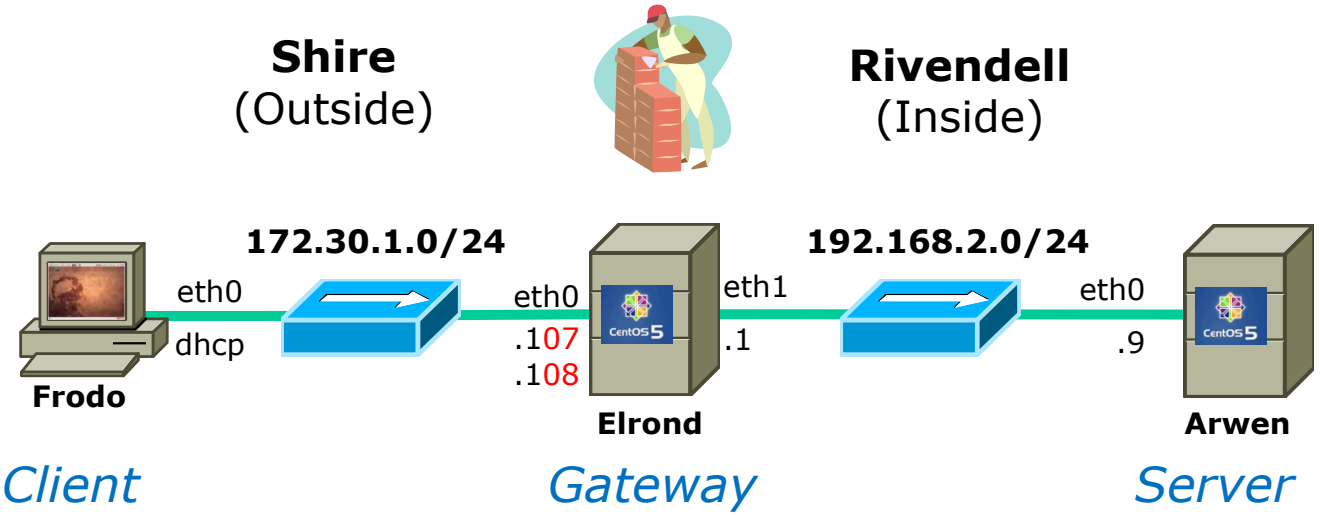

```
root@frodo:~# telnet 172.30.1.108
Trying 172.30.1.108...
Connected to 172.30.1.108.
Escape character is '^]'.
CentOS release 5.2 (Final)
Kernel 2.6.18-92.1.22.el5 on an i686
login: cis192
Password:
Last login: Mon Mar 16 23:13:09 from 172.30.1.195
[cis192@arwen ~]$ exit
```

From Frodo we have access to the Telnet server on Arwen using the public IP address on Elrond




```
[root@arwen ~]# ping google.com
PING google.com (74.125.67.100) 56(84) bytes of data.
64 bytes from gw-in-f100.google.com (74.125.67.100): icmp_seq=1 ttl=243 time=221 ms
64 bytes from gw-in-f100.google.com (74.125.67.100): icmp_seq=2 ttl=243 time=204 ms

--- google.com ping statistics ---
3 packets transmitted, 2 received, 33% packet loss, time 2003ms
rtt min/avg/max/mdev = 204.217/212.833/221.450/8.628 ms
[root@arwen ~]#
```

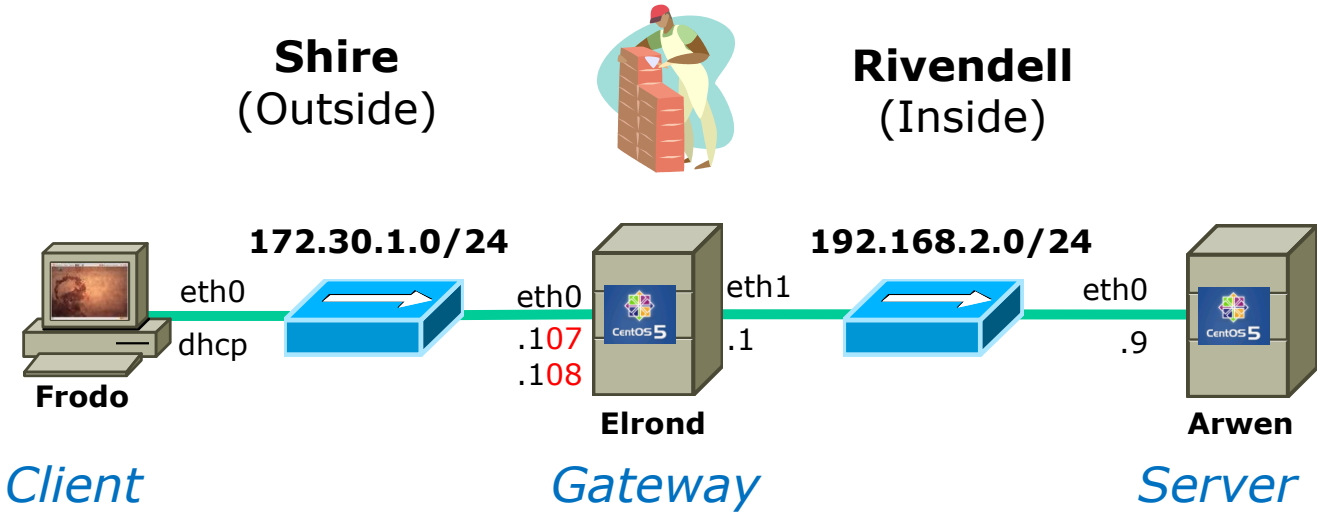
Arwen, which is on a private network, has Internet access via NAT on Elrond. No static routes needed!




```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

Set the policies on each filter chain to DROP. If no rules in the chains match then the packets will be dropped.

DROP is "silent" - no error messages in a response are sent back



iptables -A FORWARD -s 192.168.2.0/24 -d 0/0 -m state --state NEW -j ACCEPT

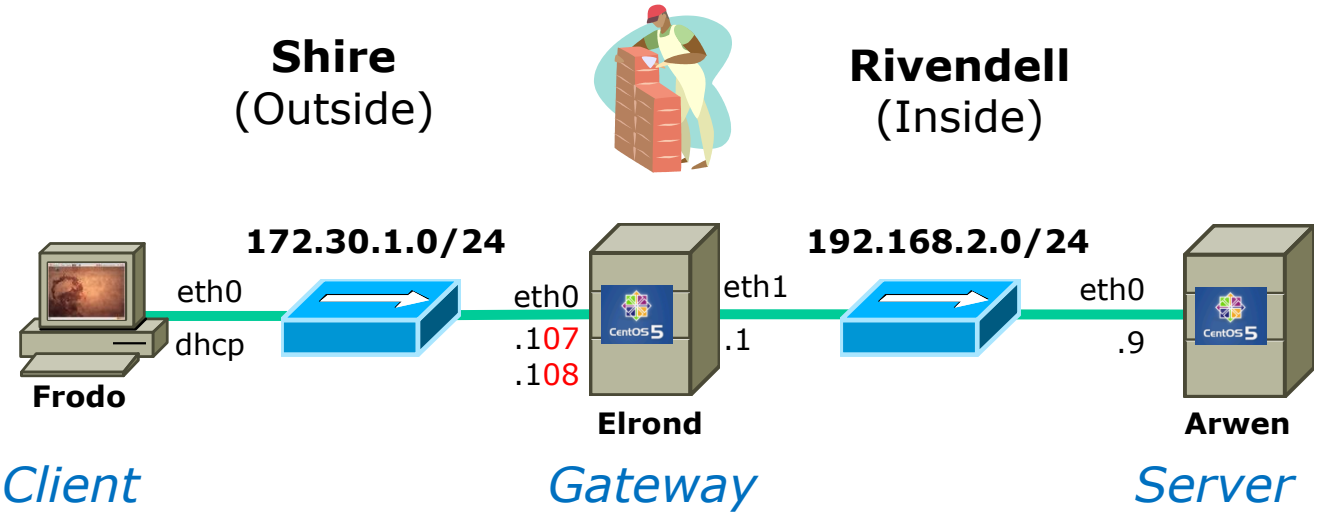
All new packets from Rivendell hosts (Arwen) will be forwarded

iptables -A FORWARD -s 0/0 -d 192.168.2.9 -m state --state NEW -p tcp --dport 23 -j ACCEPT

All Telnet packets going to the Telnet Server will be forwarded

iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

All related and established connection packets will be forwarded

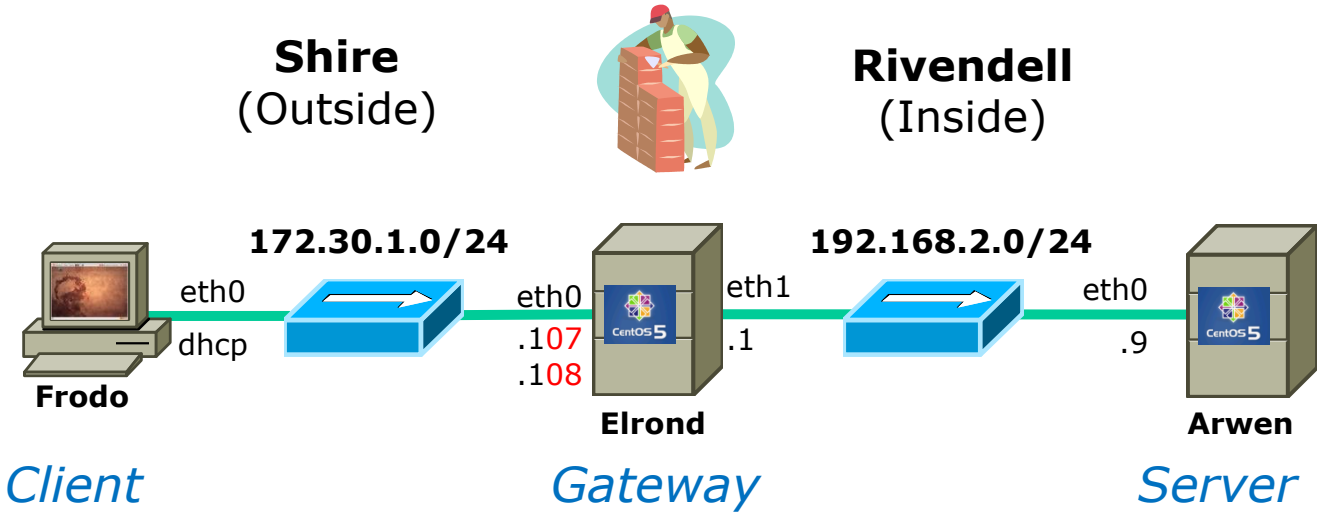


iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

All packets from Elrond are allowed out

iptables -A INPUT -i eth1 -s 192.168.2.0/24 -d 192.168.2.1 -m state --state NEW -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

Allow any new incoming connections as well as ongoing traffic from Rivendell hosts

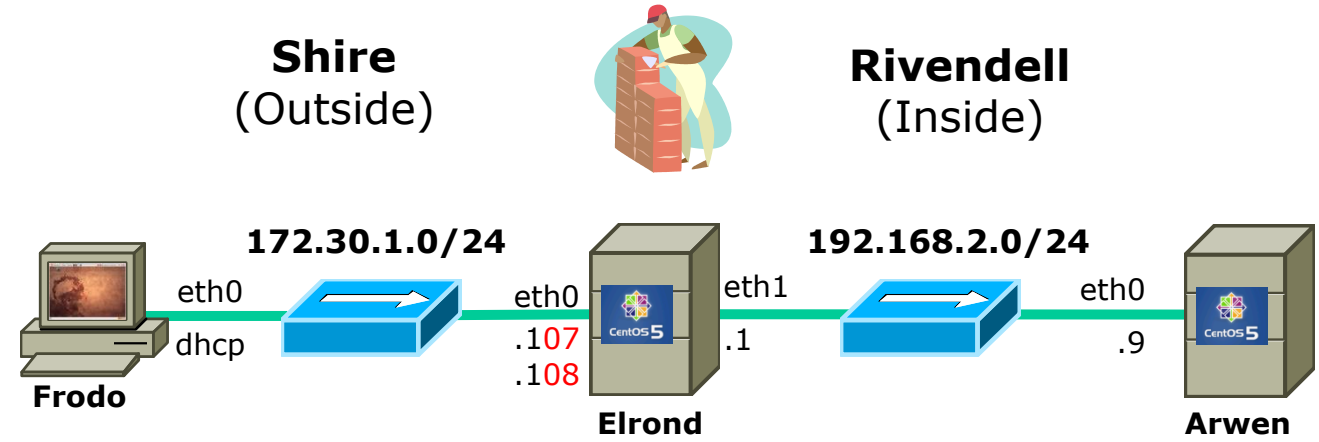


iptables -t nat -A PREROUTING -i eth0 -d 172.30.1.108 -j DNAT --to-destination 192.168.2.9

Translate any incoming packets to the public IP address to Arwen

iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.9 -j SNAT --to-source 172.30.1.108
iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.0/24 -j SNAT --to-source 172.30.1.107

NAT outgoing Arwen packets to use the second public IP address (used for the Telnet server), for the other Rivendell hosts we will NAT to Elrond's first public IP address.




```
iptables -A INPUT -j LOG --log-level info --log-prefix "iptables INPUT: "
iptables -A FORWARD -j LOG --log-level info --log-prefix "iptables FORWARD: "
```

Log firewall events for the INPUT and FORWARD chains

Wrap

New commands, daemons and files:

- service
- chconfig
- killall
- netstat
- iptables
- netstat
- service
- yum

Daemons and related configuraton 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
iptables	/etc/sysconfig/iptables

New commands, daemons and files:

- iptables
- netstat
- service
- yum

Daemons and related configuration files

tcpd	/etc/hosts.allow,hosts.deny

Next Class

Assignment: Check Calendar Page
http://simms-teach.com/cis192calendar.php

Lab 5 due

Quiz questions for next class:

- How do you show the current filter table chains?
- How do you show the current nat table chains?
- How do set the FORWARD chain policy to ACCEPT?

Backup

super daemons

Application Layer

inet Daemon

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

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