Firewalling and Network Security I -Linux

Jeff Muday
Academic Computing Specialist
Wake Forest University

WAKE FOREST
UNIVERSITY

Objectives: **Firewalling and Network Security**

After completing this module you should be able to understand and utilize:

- Firewalling and Network Security principles
- Controlling access to daemons
- Basic firewalling with **ipchains** and **iptables**
- Network/routing debugging procedures
- Interface configuration under Linux
- The secure shell (sshd , ssh , and scp)

NB: this is only an introduction. An in-depth treatment would take days.

Concepts

Three important concepts:

- Controlling network traffic into / through your system (packet filtering)
- Controlling access to services / daemons
- Avoid insecure services like telnet, ftp; and replace with ssh, scp etc.

Types of Firewalls

- External
  - Dedicated Hardware
  - Hybrid Systems (NAT router)
- Application Level

  (application makes decision of whom to service- PAM, hosts.allow, hosts.deny)
- Packet Detection and Filtering
  - WinXP – firewall service or 3rd party program
  - Linux (kernel) – ipchains, iptables
  - BSD (kernel) – ipfw

What is Packet Filtering?

- Checks packet headers before acting on them
- Can ignore, reject or accept packets
- Makes decision based on source, destination, or
- packet type, or a combination parameters
- Set up filtering using **ipchains** under kernel 2.2
  - Older kernels used **ipfwadm**
  - The new 2.4 kernel now uses **iptables**

Controlling Access to Daemons

- Access control for run-on-demand daemons done with **inetd**

  `/etc/hosts.allow`

  `/etc/hosts.deny`

  `/etc/inetd.conf`

- Flaw in **inetd** would still let things through
  - Best to drop *rogue* packets as soon as possible
  - Should combine use of **inetd** with packet filtering
  - Consider using the **xinetd** replacement

TCP Wrappers (/usr/sbin/tcpd)

- Raw **inetd** applies no access controls
- OK if you trust your network
- ‘TCP Wrappers’ invented to fix this
- Standard with most installations
- the wrapper sits between **inetd** and the server daemon
- **inetd** not itself insecure
  - Insecurity springs from how you use it
  - Wrappers now integral with **xinetd**

TCP Wrapper Validation

- Uses /etc/hosts.deny and /etc/hosts.allow
- Well-documented, see man hosts.allow

  Example of /etc/hosts.deny

  ALL:ALL

  – Denies all services to everyone
- Selectively enable trusted hosts in /etc/hosts.allow

```
in.telnetd : .mydomain.org 192.168.1.
ipop3d : 192.168.1.
in.ftpd : 192.168.1.
```

- Can base permissions on full/partial domains or addresses

Introduction to Packet Filtering

- Allows you to protect your machine as well as machines *behind* them
- Checks packet headers before acting on them
  - Can ignore, reject or accept packets
  - Makes decision based on source, destination, or packet type or a combination of parameters
- Filtering is set up by using **ipchains** under kernel 2.2
  - Older kernels used **ipfwadm**
  - 2.4 kernel now uses **iptables**

Basic Packet Filtering

- Two main considerations
  - Port Filtering
  - Host Filtering
- Block services you don't need
- Limit services you *do* need to specific machines/networks

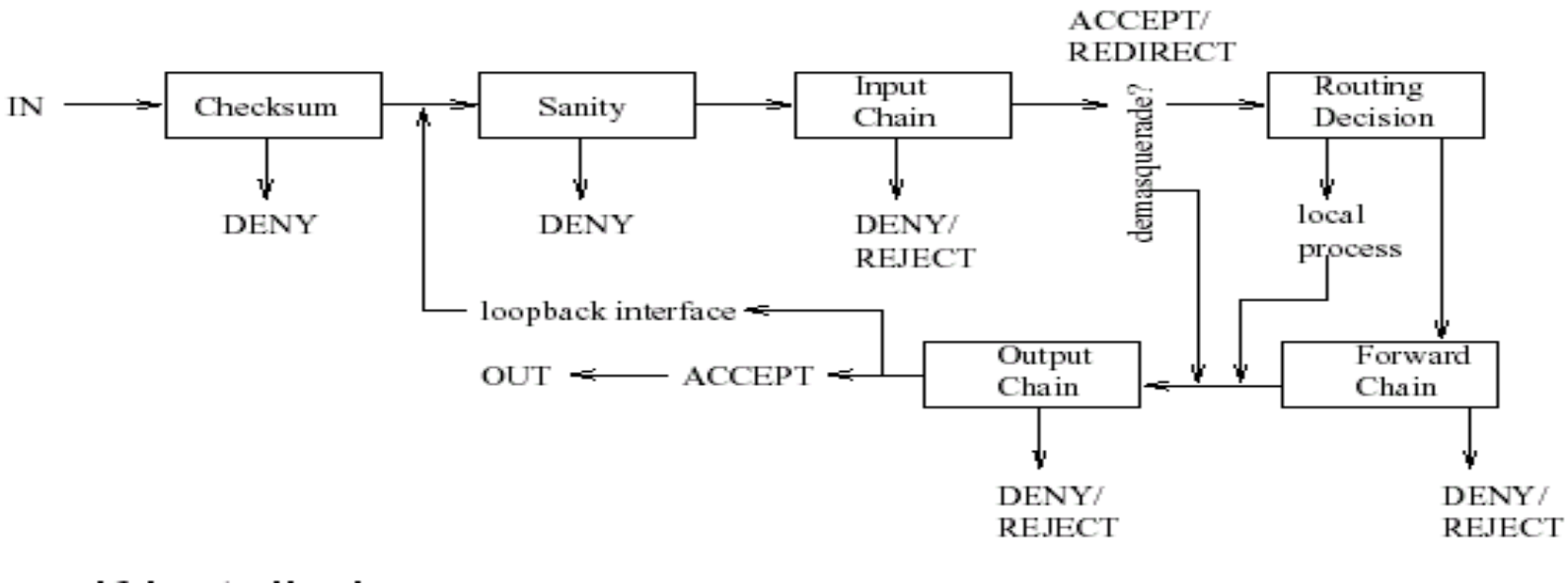
ipchains

- Packet filtering set up using **`ipchains`**
- All the filtering is done at the *kernel level*
  - Not by **`ipchains`**
  - **`ipchains`** only sets up/modifies kernel rules
- All packets entering and leaving are examined and accepted, denied, rejected, etc. according to user specified rules.
    - This includes loopback (127.0.0.1) traffic!

`ipchains` Details

- Every packet goes through one or more '*chains*'
  - A 'chain' is a set of rules
  - Rules can accept, reject, or deny a packet
  - Can also send it to another chain
- Three default chains, *input*, *output*, *forward*
  - If a packet passes through a default chain without matching:
    - Fate is determined by the chain's selected *default policy*
    - Default policies can be *ACCEPT*, *DENY*, or *REJECT*
  - If it reaches the end of a user defined chain, it carries on where it left off

`ipchains` "schematic"



- If installed, see man pages and linuxdocs.org
  `/usr/doc/ipchains-1.3.9/HOWTO.html`
  for much more detail
- *forward* is for packets routed to other hosts
  Not covered here (used in router-like ops)

More Information

- Man pages
- Linux Documentation Project (TLDP.ORG)
- SANS institute www.sans.org