NC STATE UNIVERSITY Computer Science

CSC 474/574
Information Systems Security

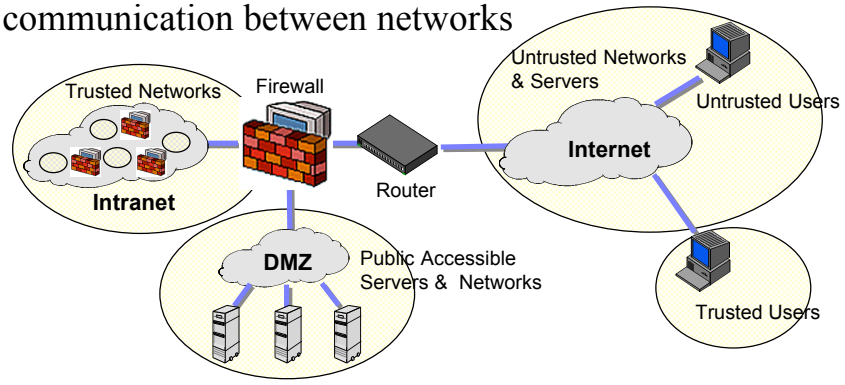
Topic 7.4 Firewalls

Outline

- What are firewalls?
- Types
  - Filtering
    - Packet filtering
    - Session filtering
  - Proxy
    - Circuit Level
    - Application Level
- Brief introduction to Linux firewall

What is a firewall?

- Device that provides secure connectivity between networks (internal/external; varying levels of trust)
- Used to implement and enforce a security policy for communication between networks



Firewalls

- From Webster's Dictionary: *a wall constructed to prevent the spread of fire*
- Internet firewalls are more the moat around a castle than a building firewall
- Controlled access point

Firewalls can:

- Restrict incoming and outgoing traffic by IP address, ports, or users
- Block invalid packets

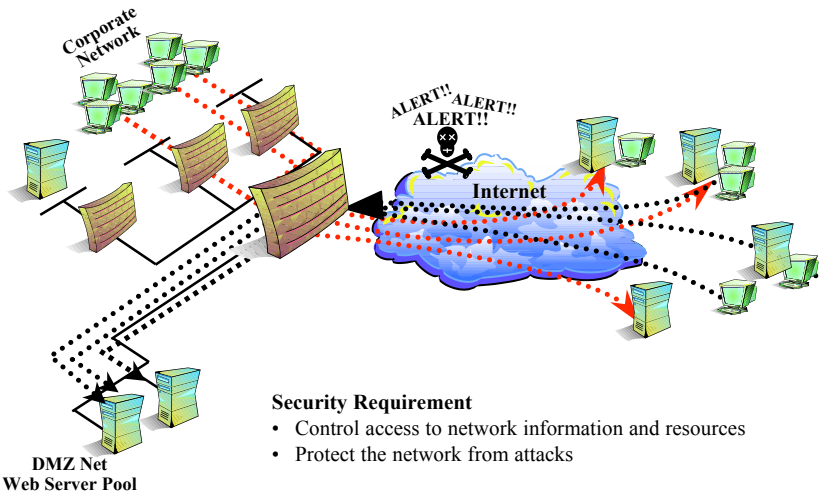
Convenient

- Give insight into traffic mix via logging
- Network Address Translation
- Encryption

Firewalls Cannot Protect…

- traffic that does not cross it
  - routing around
  - Internal traffic
- when misconfigured

Access Control



Security Requirement
- Control access to network information and resources
- Protect the network from attacks

Filtering

- Typically route packets
- Packets checked then passed
- Inbound & outbound affect when policy is checked
- Client ↔Server

Filtering

- Packet filtering
  - Access Control Lists
- Session filtering
  - Dynamic Packet Filtering
  - Stateful Inspection
  - Smart packet filtering
  - Context Based Access Control

Filtering

- Fragmentation/reassembly
- Sequence number checking
- ICMP

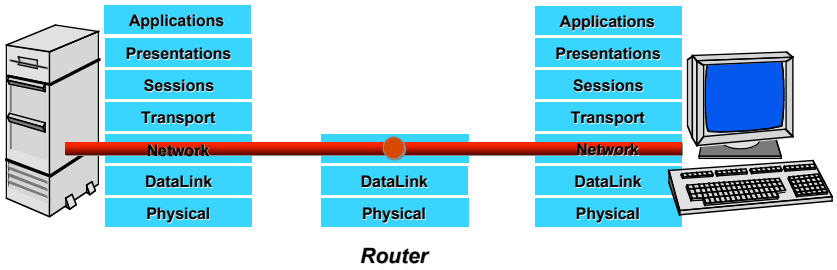
Packet Filtering

- Decisions made on a per-packet basis
- No state information saved

Typical Configuration

- Ports > 1024 left open
- If dynamic protocols are in use, *entire ranges of ports must be allowed* for the protocol to work.

Packet Filter



Session Filtering

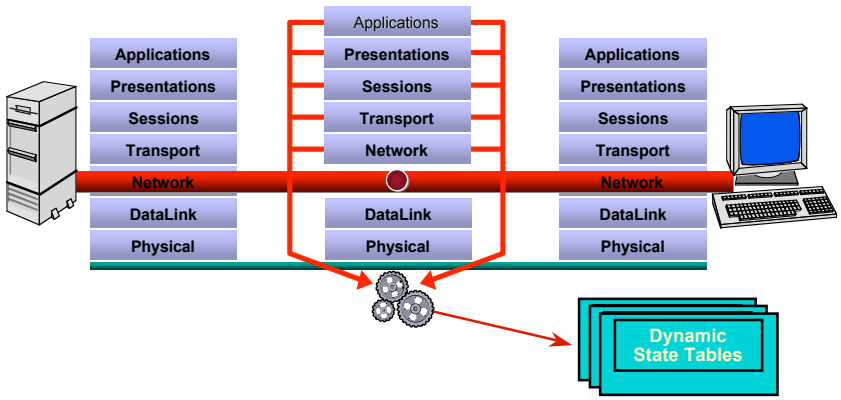
- Packet decision made in the context of a connection
- If packet is a new connection, check against security policy
- If packet is part of an existing connection, match it up in the state table & update table

Typical Configuration

- All denied unless specifically allowed
- Dynamic protocols (FTP, H323, RealAudio, etc.) allowed only if supported

Session Filtering

- **Screens** ALL attempts, **Protects** All applications
- **Extracts** & **maintains** 'state' information
- **Makes** an intelligent **security** / **traffic** decision



Proxy Firewalls

- Relay for connections
- Client ↔ Proxy ↔ Server
- Two flavors
  - Application level
  - Circuit level

Application Gateways

- Understands specific applications
  - Limited proxies available
  - Proxy 'impersonates' both sides of connection
- Resource intensive
  - process per connection
- HTTP proxies may cache web pages

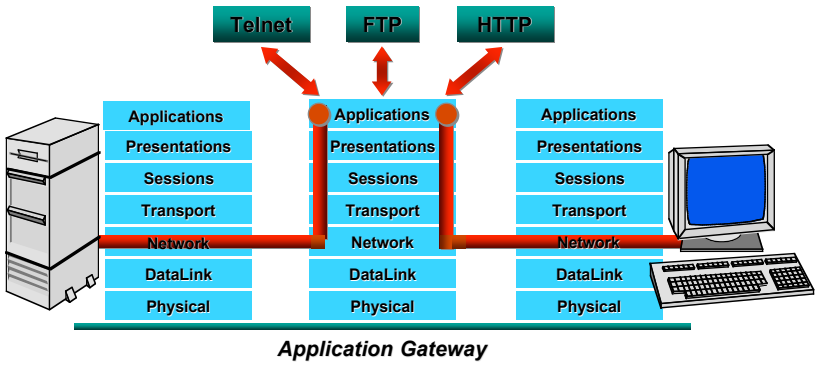
Application Gateways

- More appropriate to TCP
- ICMP difficult
- *Block all unless specifically allowed*
- Must write a new proxy application to support new protocols
  - Not trivial!

Application Gateways

- Clients configured for proxy communication
- Transparent Proxies

Application Layer GW/proxy



Circuit-Level Gateways

- Support more services than Application-level Gateway
  - less control over data
- Hard to handle protocols like FTP
- Clients must be aware they are using a circut-level proxy
- Protect against fragmentation problem

SOCKS

- Circuit level Gateway
- Support TCP
- SOCKS v5 supports UDP, earlier versions did not
- See http://www.socks.nec.com

Comparison

	Security	Performance	Service Support
Packet Filter	3	1	No dynamic w/o holes
Session Filter	2	2	Dependent on vendor for dynamic support
Circuit GW	2	3	
App. GW	1	4	Typically < 20

Lower is better for security & performance

Comparison (Cont'd)

	Modify Client Applications?
Packet Filter	No
Session Filter	No
Circuit GW	Typical, SOCKS-ify client applications
App. GW	Unless transparent, client application must be proxy-aware & configured

Comparison (Cont'd)

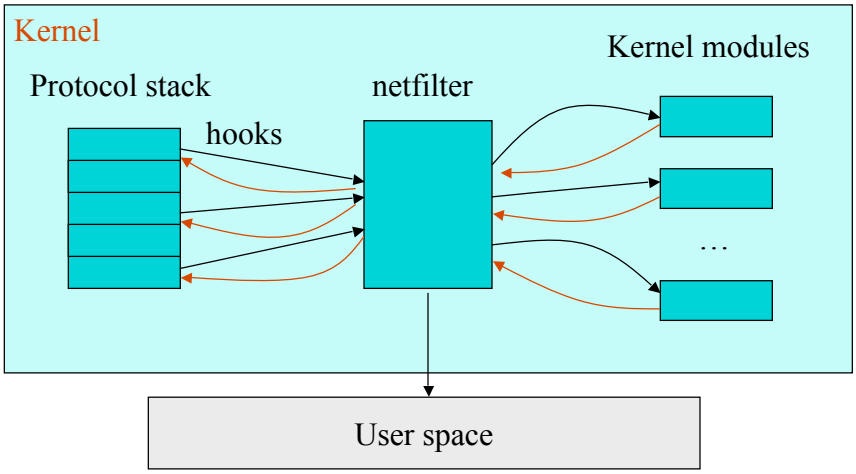
	ICMP	Fragmentation
Packet Filter	Yes	No
Session Filter	Yes	Maybe
Circuit GW	(SOCKS v5)	Yes
App. GW	No	Yes

Linux Firewall: iptables

- History
  - ipfw
  - ipfwadm
  - ipchains
  - iptables
    - Based on the netfilter framework

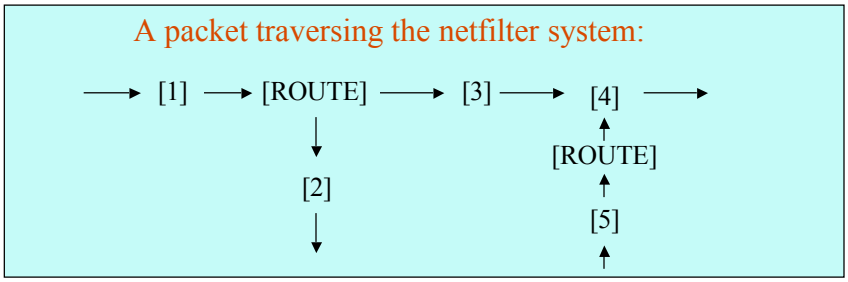
The Netfilter Framework

- A framework for packet mangling

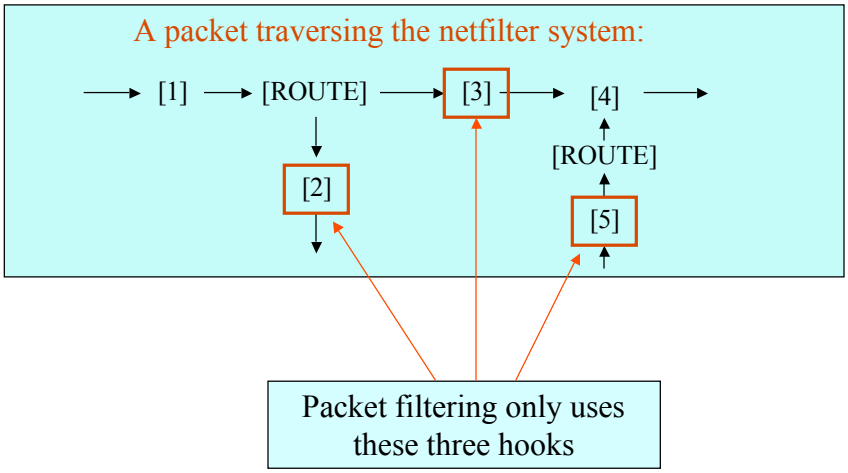


The Netfilter Framework (Cont'd)

- Current protocols
  - IPv4, IPv6, and DECnet.
- Five hooks for IPv4
  - [1]: Pre-routing hook; [2]: Local-in hook;
  - [3]: Forward hook; [4]: Local-out hook;
  - [5]: Post-routing hook



Packet Filtering



IP Tables

- A packet selection system
  - Direct descendent of ipchains
- Used for
  - Packet filtering
  - Network Address Translation (NAT)
    - Masquerading, port forwarding, transparent proxying
  - Packet mangling
    - Actual changing of packet information

User Space Tool: iptables

- iptables
  - Command to configure and communicate with the kernel modules
- iptables for packet filtering
  - Three chains
    - INPUT
    - OUTPUT
    - FORWARD

Iptables for Packet Filtering

- You need three things to configure a firewall rule
  - Which chain?
  - What packet pattern?
  - What action to apply?
- Example
  - Drop all packets from 200.200.200.1
  - iptables -A INPUT -s 200.200.200.1 -j DROP
  - Use "man iptables" on Linux to get more information.