

## Distributed Systems

### 22. Firewalls & VPNs

Paul Krzyzanowski  
pxk@cs.rutgers.edu

## inetd

Most UNIX systems ran a large number of TCP services as daemons

- e.g., *rlogin*, *rsh*, *telnet*, *ftp*, *finger*, *talk*, ...

Later, one process, *inetd*, was created to listen to a set of ports and then spawn the service on demand

- pass sockets as standard in/standard out file descriptors
- servers don't run unless they are in use

## TCP wrappers (*tcpd*)

- Plug-in replacement to *inetd*
- Restrict access to TCP services
  - Allow only specified machines to execute authorized services
  - Monitor and log requests
- Specify rules in two files:
  - *hosts.allow* and *hosts.deny*
  - access:
    - grant access if service:client in /etc/hosts.allow
    - deny access if service:client in /etc/hosts.deny
    - otherwise allow access
- support for booby traps (*honeypots*)

## Defending the network

## Firewalls

Isolate trusted domain of machines from the rest of the untrusted world

- move all machines into a private network
- disconnect all other systems
- untrusted users not allowed

not acceptable – we want to be connected

Solution:

protect the junction between a trusted internal network of computers from an external network with a firewall

## Firewall

- First line of defense
  - Protect the WAN-LAN boundary
- Also: protect communications within internal LANs
  - Ensure that employees to not access LANs/services that they are not allowed to

## Firewalls

Two major approaches to building firewalls:

packet filtering

proxies

## Packet filtering

- Selective routing of packets
  - Between internal and external hosts
- By routers, kernel modules, or firewall software
- Allow or block certain types of packets

### Screening router

- determine route *and* decide whether the packet should be routed

## Packet filtering: screening router

IP packet data

Filter by

- IP source address, IP destination address
- TCP/UDP source port, TCP/UDP destination port
- Protocol (TCP, UDP, ICMP, ...)
- ICMP message type
- interface packet arrives on
- destination interface

Allow or block packets based on any/all fields

- Block any connections from certain systems
- Disallow access to "dangerous services"

## Packet filtering

### Stateless inspection

- filter maintains no state
- each packet examined on its own

## Packet filtering

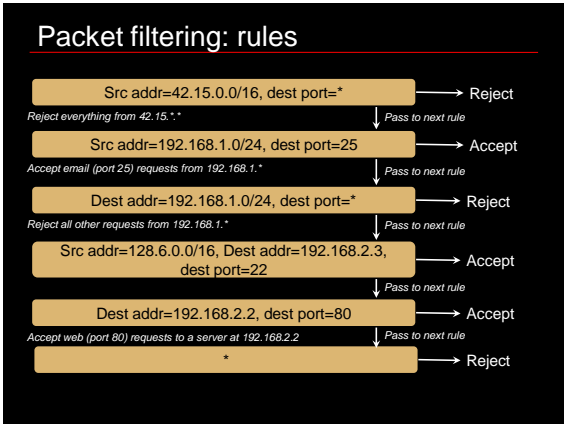
### Stateful inspection

- keep track of TCP connections (SYN, SYN/ACK packets – has a TCP connection been established?)
  - e.g. no rogue packets when connection has not been established
- "related" ports: allow data ports to be opened for FTP sessions
- Port triggering (outbound port triggers other port access to be redirected to the originating system)
  - Generally used with NAT (Network Address Translation)
- Limit rates of SYN packets
  - avoid SYN flood attacks
- Other application-specific filtering
  - Drop connections based on pattern matching
  - Rewrite port numbers in data stream

## Packet filtering

### Screening router

- allows/denies access to a service
- cannot protect operations within a service



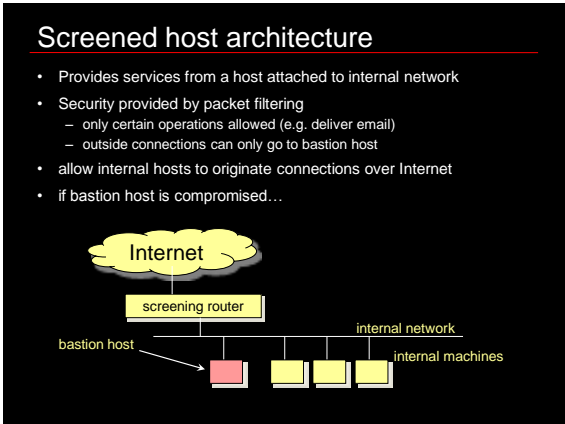
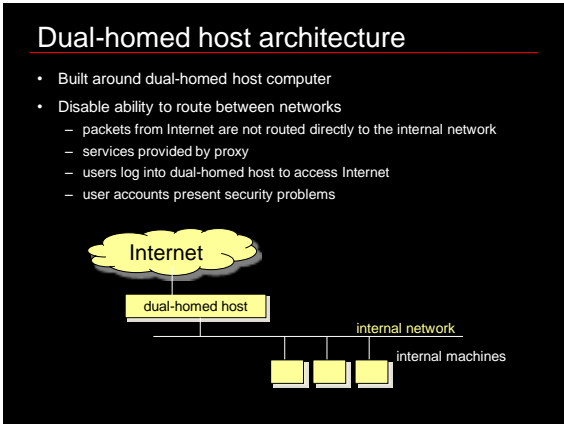
- ### Proxy services
- Application or server programs that run on firewall host
    - dual-homed host
    - bastion host
  - Take requests for services and forward them to actual services
  - provide replacement connections and act as gateway services
  - Application-level gateway
- Stateful inspection and protocol validation

### Proxy services

Proxies are effective in environments where direct communication is restricted between internal and external hosts

- dual-homed machines and packet filtering

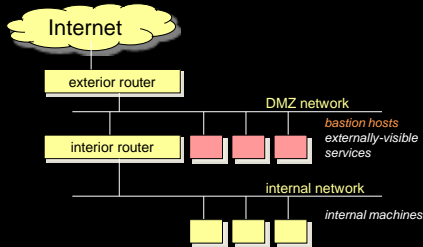
- ### Proxy example
- Checkpoint Software Technologies' Firewall
- mail proxy:
- mail address translation: rewrite From:
  - redirect To:
  - drop mail from given address
  - strip certain mime attachments
  - strip Received info on outbound mail
  - drop mail above given size
  - perform anti-virus checks on attachments
- does not allow outsiders direct connection to a local mailer



## Screened subnet architecture

Add extra level of isolation for internal network

- Place any externally visible machines on a separate *perimeter network (DMZ)*



## Screened subnet architecture

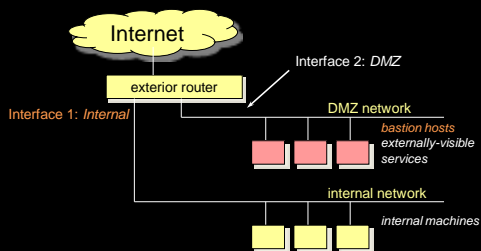
### Exterior router (access router)

- protects DMZ and internal network from Internet
- generally... allow anything outbound ... that you need
- block incoming packets from Internet that have forged source addresses
- allow incoming traffic only for bastion hosts/services.

### Interior router (choke router)

- protects internal network from Internet and DMZ
- does most of packet filtering for firewall
- allows selected outbound services from internal network
- limit services between bastion host and internal network

## Single router DMZ



## Deep Packet Inspection (DPI)

- "Standard" firewalls
  - Operate at layers 2 & 3 (IP, TCP/IP, UDP/IP, ICMP)
  - Examine packet headers
- Application-level (layer 7) inspection
  - Apply policies based on application data
  - Protocol validation
  - Port triggering based on application data (e.g., FTP, SIP)
  - Packet scheduling

## Intrusion Protection System (IPS)

- Filter packets in real time
- Scan for known malicious patterns in code/data
  - Malware signatures
  - Protocol errors
  - Malicious headers
- Scan for anomalous patterns
  - Port scanning
  - Monitor rate of connection requests (DoS attacks)

## Example of DPI & Application-level IPS

- HTTP
  - Validate that HTTP is valid
  - Block headers above a certain size
  - Block headers with binary data (possible attack)
  - Limit server request rate to avoid DoS attacks
  - Filter URLs



### Private networks

**Problem**

- You have several geographically separated local area networks that you would like to have connected securely

**Solution**

- Set up a private network line between the locations
- Routers on either side will be enabled to route packets over this private line

### Private networks

LAN A (New York)      LAN B (London)

- Problem: \$\$\$\$\$\$\$\$\$\$\$!

### Virtual private networks (VPNs)

Alternative to private networks

- Use the public network (internet)

Service appears to users as if they were connected directly over a private network

- Public infrastructure is used in the connection

### Building a VPN: tunneling

**Tunneling**

- Links two network devices such that the devices appear to exist on a common, private backbone
- Achieve it with encapsulation of network packets

### Tunneling

external address: 129.42.16.99      external address: 17.254.0.91

src:	dest:	data	
192.168.1.10	192.168.2.32		

### Tunneling

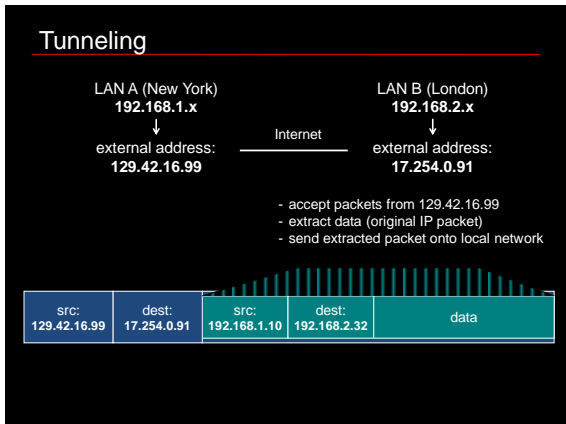
LAN A (New York)      LAN B (London)

192.168.1.x      192.168.2.x

external address: 129.42.16.99      Internet      external address: 17.254.0.91

- route packets for 192.168.2.x to VPN router
- VPN router:
  - envelope packet with address of remote router
  - sends it to IP interface

src:	dest:	src:	dest:	data
129.42.16.99	17.254.0.91	192.168.1.10	192.168.2.32	



### Building a VPN: tunneling

#### Operation

- LAN-A and LAN-B each expose a single outside address and port.
- A machine in the DMZ (typically the router) listens on this address and port
- On LAN-A, any packets addressed to LAN-B are routed to this system.
- VPN software takes the entire packet that is destined for LAN-B and, treating it as data, sends it over an to the VPN listener on LAN-B (router)
- On LAN-B, the VPN software extracts the data (the entire packet) and sends it out on its local area network

### Building a VPN: security

No need to make all machines in the local area networks accessible to the public network ... just the VPN router

**BUT...** an intruder can:

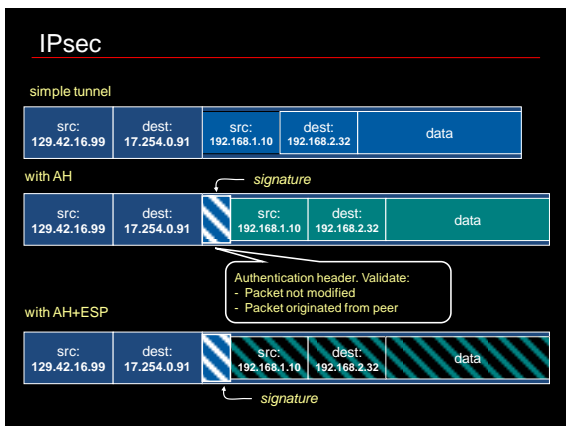
- examine the encapsulated packets
- forge new encapsulated packet

Solution:

- **encrypt** the encapsulated packets
  - Symmetric algorithm for encryption using session key
- need mechanism for key exchange


### IPsec: RFC 1825, 1827

- IP-layer security mechanism
- Covers authentication and encryption
- Application gets benefits of network encryption without modification
- Additional header added to packet:
  - **IP Authentication header**
    - Identifies proper source and destination – basis of point-to-point authentication
    - *Signature for IP header* [HMAC: SHA-1 hash(message, key)]
- **Encapsulating Security Protocol (ESP)**
  - Tunnel mode: *encrypt entire IP packet* (data and IP/TCP/UDP headers) [typically 3DES or AES encryption]
  - or Transport mode: encrypt only IP/TCP/UDP headers (faster)
- Key management: manual, Diffie-Hellman, or RSA



### OpenVPN

- Network-network or network-to-host VPN
- Key exchange
  - SSL/TLS, username-password
- Encryption
  - Any cryptographic algorithm supported in OpenSSL package
- Data transmission
  - ESP tunnels on a single TCP or UDP port



The End