Firewall
Roadmap

- Basic firewall concept
- Filtering firewall
- Proxy firewall
- Network Address Translation

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 FIREWALL

Filtering Firewall

- Packets checked then passed
- Inbound & outbound affect when policy is checked
Filtering

• Packet filtering
  – Access Control Lists
• Session filtering
  – Dynamic Packet Filtering
  – Stateful Inspection
  – Context Based Access Control

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.
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, RealAudio, etc.) allowed only if supported

Session Filtering

- **Screens** ALL attempts, **Protects** All applications
- **Extracts & maintains** ‘state’ information
- **Makes** an intelligent security / traffic decision
Example: FTP Protocol

FTP Server

- Client opens command channel to server; tells server second port number.
- Server acknowledges.
- Server opens data channel to client's second port.
- Client acknowledges.

FTP Client

- "PORT 5151"
- "OK"
- DATA CHANNEL
- TCP ACK

Example FTP – Packet Filter

Format:

access-list <rule number> <permit|deny> <protocol> <SOURCE host with IP address| any|IP address and mask> [<gt|eq port number>] <DEST host with IP address| any|IP address and mask> [<gt|eq port number>]

The following allows a user to FTP (not passive FTP) from any IP address to the FTP server (172.168.10.12):

```
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 21
access-list 100 permit tcp any gt 1023 host 172.168.10.12 eq 20

! Allows packets from any client to the FTP control and data ports
access-list 101 permit tcp host 172.168.10.12 eq 21 any gt 1023
access-list 101 permit tcp host 172.168.10.12 eq 20 any gt 1023

! Allows the FTP server to send packets back to any IP address with TCP ports > 1023
```

interface Ethernet 0
access-list 100 in  ! Apply the first rule to inbound traffic
access-list 101 out  ! Apply the second rule to outbound traffic

FTP – Passive Mode

1. Client opens command channel to server; requests passive mode.
2. Server allocates port for data channel; tells client port number.
3. Client opens data channel to server’s second port.

Example FTP : Session Filter
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

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 circuit-level proxy
- Protect against fragmentation problem
Example: SOCKS

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

## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Security</th>
<th>Performance</th>
<th>Service Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Filter</td>
<td>3</td>
<td>1</td>
<td>No dynamic w/o holes</td>
</tr>
<tr>
<td>Session Filter</td>
<td>2</td>
<td>2</td>
<td>Dependent on vendor for dynamic support</td>
</tr>
<tr>
<td>Circuit GW</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>App. GW</td>
<td>1</td>
<td>4</td>
<td>Typically &lt; 20</td>
</tr>
</tbody>
</table>

*Lower is better for security & performance*
### Comparison

<table>
<thead>
<tr>
<th></th>
<th>Modify Client Applications?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Filter</td>
<td>No</td>
</tr>
<tr>
<td>Session Filter</td>
<td>No</td>
</tr>
<tr>
<td>Circuit GW</td>
<td>Typical, SOCKS-ify client applications</td>
</tr>
<tr>
<td>App. GW</td>
<td>Unless transparent, client application must be proxy-aware &amp; configured</td>
</tr>
</tbody>
</table>

### Comparison

<table>
<thead>
<tr>
<th></th>
<th>ICMP</th>
<th>Fragmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet Filter</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Session Filter</td>
<td>Yes</td>
<td>Maybe</td>
</tr>
<tr>
<td>Circuit GW</td>
<td>(SOCKS v5)</td>
<td>Yes</td>
</tr>
<tr>
<td>App. GW</td>
<td>No</td>
<td>yes</td>
</tr>
</tbody>
</table>
Proxying UDP/ICMP

- Why isn’t UDP or ICMP proxied as much as TCP?
- TCP’s connection-oriented nature easier to proxy
- UDP & ICMP harder (but not impossible) since each packet is a separate transaction
- Session filters determine which packets appear to be replies

Circuit Level GW

- Operate at user level in OS
- Have circuit program ‘route’ packets between interfaces instead of OS routing code
NETWORK ADDRESS TRANSLATION

NAT: Network Address Translation

- Useful if organization does not have enough real IP addresses
- Extra security measure if internal hosts do not have valid IP addresses (harder to trick firewall)
- Only really need real IP addresses for services outside networks will originate connections to
NAT

- Many-to-1 (n-to-m) mapping
- 1-to-1 (n-to-n) mapping
- Proxies provide many-to-1
- NAT not required on filtering firewalls

Encryption (VPNs)

- Allows trusted users to access sensitive information while traversing untrusted networks
- Useful for remote users/sites
- IPSec