Firewalls and NAT

By conventional definition, a firewall is a partition made of fireproof material designed to prevent the spread of fire from one part of a building to another.

Firewalls

firewall
isolates organization’s internal net from larger Internet, allowing some packets to pass, blocking others.

222.22/16

privately administered
Firewalls: Goals

- All traffic from outside to inside and vice-versa passes through the firewall
- Only authorized traffic, as defined by local security policy, will be allowed to pass
- Firewall itself is immune to penetration

Firewalls: Taxonomy

1. Traditional packet filters
   - Filters often combined with router, creating a firewall
2. Stateful filters
3. Application gateways

Major firewall vendors:
- Checkpoint
- Cisco PIX
Firewall

- Firewall == system that filters TCP/IP UDP/IP packets according to rules
- Either software on user machine or network router
Traditional packet filters

Analyzes each datagram going through it; makes drop decision based on:

- Source IP address
- Destination IP address
- Source port
- Destination port
- TCP flag bits
  - SYN bit set: datagram for connection initiation
  - ACK bit set: part of established connection
- TCP or UDP or ICMP
  - Firewalls often configured to block all UDP
- Direction
  - Is datagram leaving or entering internal network?
- Router interface
  - Decisions can be different for different interfaces

Filtering rules - examples

<table>
<thead>
<tr>
<th>Policy</th>
<th>Firewall Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>No outside Web access.</td>
<td>Drop all outgoing packets to any IP address, port 80</td>
</tr>
<tr>
<td>Outside connections to public Web server only.</td>
<td>Drop all incoming TCP SYN packets to any IP except 130.207.244.203, port 80</td>
</tr>
<tr>
<td>Prevent Web-radios from eating up the available bandwidth.</td>
<td>Drop all incoming UDP packets - except DNS and router broadcasts.</td>
</tr>
<tr>
<td>Prevent your network from being used for a Smuft DoS attack.</td>
<td>Drop all ICMP packets going to a “broadcast” address (e.g. 130.207.255.255).</td>
</tr>
<tr>
<td>Prevent your network from being tracerouted</td>
<td>Drop all outgoing ICMP unreachables</td>
</tr>
</tbody>
</table>
Access control lists

Apply rules from top to bottom:

<table>
<thead>
<tr>
<th>action</th>
<th>source address</th>
<th>dest address</th>
<th>proto</th>
<th>source port</th>
<th>dest port</th>
<th>flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>222.22/16</td>
<td>outside of 222.22/16</td>
<td>TCP</td>
<td>&gt; 1023</td>
<td>80</td>
<td>any</td>
</tr>
<tr>
<td>allow</td>
<td>outside of 222.22/16</td>
<td>222.22/16</td>
<td>TCP</td>
<td>80</td>
<td>&gt; 1023</td>
<td>ACK</td>
</tr>
<tr>
<td>allow</td>
<td>222.22/16</td>
<td>outside of 222.22/16</td>
<td>UDP</td>
<td>&gt; 1023</td>
<td>53</td>
<td>---</td>
</tr>
<tr>
<td>allow</td>
<td>outside of 222.22/16</td>
<td>222.22/16</td>
<td>UDP</td>
<td>53</td>
<td>&gt; 1023</td>
<td>----</td>
</tr>
<tr>
<td>deny</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
</tr>
</tbody>
</table>

Access control lists (2.)

- Each router/firewall interface can have its own ACL
- Most firewall vendors provide both command-line and graphical configuration interface
Traditional packet filters

- Advantages
  - One screening router can protect entire network
  - Can be efficient if filtering rules are kept simple
  - Widely available. Almost any router, even Linux boxes

- Disadvantages
  - Can be penetrated
  - Cannot enforce some policies. For example, permit certain users.
  - Rules can get complicated and difficult to test

Network or host firewall

Network firewall:

Host firewall:
Example: Iptables – chain types

Internet

Linux host w/ iptables

INPUT chain (to iptables host)

protected network

Internet

Linux host w/ iptables

OUTPUT chain (from iptables host)

protected network

Internet

Linux host w/ iptables

FORWARD chain

protected network

Iptables: Example command

```bash
iptables -A INPUT -i eth0 -s 232.16.4.0/24 -j ACCEPT
```

- Sets a rule
  - Accepts packets that enter from interface eth0 with source address in 232.16.4/24
- Kernel applies rules in order
  - First matching rule determines action for packet
- Append: -A
  - Adds rule to bottom of existing rules
Stateful filters

- Stateless filters: Any packet with ACK=1 and source port 80 gets through
  - Attack with malformed packets: send ACK=1 segments
- Stateful filter: Adds more intelligence to decision-making process
  - Stateful = remember past packets
  - Needs very dynamic state table

Stateful filters: Example

- Log each TCP conn initiated through firewall: SYN segment
- Timeout entries without activity after, e.g., 60 seconds

<table>
<thead>
<tr>
<th>source address</th>
<th>dest address</th>
<th>source port</th>
<th>dest port</th>
</tr>
</thead>
<tbody>
<tr>
<td>222.22.1.7</td>
<td>37.96.87.123</td>
<td>12699</td>
<td>80</td>
</tr>
<tr>
<td>222.22.93.2</td>
<td>199.1.205.23</td>
<td>37654</td>
<td>80</td>
</tr>
<tr>
<td>222.22.65.143</td>
<td>203.77.240.43</td>
<td>48712</td>
<td>80</td>
</tr>
</tbody>
</table>

Rule table indicates check of stateful table:
- See if there is a connection entry in stateful table
- Stateful filters can remember outgoing UDP segments
Stateful example: Example (2.)

1) Pkt arrives from outside: src=37.96.87.123, src port=80, dst=222.22.1.7, dst port=12699, SYN=0, ACK=1
2) Check filter table → check stateful table

<table>
<thead>
<tr>
<th>action</th>
<th>source address</th>
<th>dest address</th>
<th>proto</th>
<th>source port</th>
<th>dest port</th>
<th>flag bit</th>
<th>check conn</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>222.22/16</td>
<td>outside of 222.22/16</td>
<td>TCP</td>
<td>&gt; 1023</td>
<td>80</td>
<td>any</td>
<td></td>
</tr>
<tr>
<td>allow</td>
<td>outside of 222.22/16</td>
<td>222.22/16</td>
<td>TCP</td>
<td>80</td>
<td>&gt; 1023</td>
<td>ACK</td>
<td>X</td>
</tr>
<tr>
<td>allow</td>
<td>222.22/16</td>
<td>outside of 222.22/16</td>
<td>UDP</td>
<td>&gt; 1023</td>
<td>53</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>allow</td>
<td>outside of 222.22/16</td>
<td>222.22/16</td>
<td>UDP</td>
<td>53</td>
<td>&gt; 1023</td>
<td>---</td>
<td>X</td>
</tr>
<tr>
<td>deny</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
<td>all</td>
</tr>
</tbody>
</table>

3) Connection is in connection table → let packet through

Application gateways (aka proxy servers)

- App gateway between user (inside) and server (outside)
- User and server talk through proxy
- Allows fine grained/ sophisticated control
- Hinders protocol attacks
- E.g.: ftp server may not allow files >= size X
Mail servers and proxy Web servers

- Local mail server is application gateway
  - Virus detection and removal
- So is Web proxy cache
  - E.g.: virus detection and removal

Proxy gateways

- Advantages
  - Can log all connections, activity in connections
  - Can provide caching
  - Can do intelligent filtering based on “content”
  - Simplifies service control
    - Can perform user level authentication
    - Simplifies firewall rules
- Disadvantages
  - Not all services have proxied versions
  - Need different proxy server for each service
  - Requires modification of client
  - Performance
  - Hinders end-to-end encryption
**Demilitarized Zone (DMZ)**

- **Used for:** Gateways and public services
- **Advantage:** Hacked server – limited damage

**IP traceback**

**Problem:** How do we determine where malicious packet came from?

- **Why?** Attackers can spoof source IP address
- **Benefits:**
  - Determine attacker
  - Determine zombie machine participating in DDoS attack
- **Alternative:** Use ingress filtering
Methods for finding source

- Manual using current IP routing
  - Link testing: how?
    - Start from victim and test upstream links
    - Recursively repeat until source is located
    - Assume attack remains active until trace complete
  - Link testing: problem
    - Handle ISPs
    - Located zombie ...
- Logging
- Automatic using marking algorithms

Logging

- Key routers log packets (useful for forensics)
- Use data mining to find path
- Pros
  - Post mortem – works after attack stops
- Cons
  - High resource demand: 
    - need to store and process tons of data
**Marking algorithms**

- Mark packets with router addresses
  - Deterministically or probabilistically
- Trace attack using marked packets
- Strengths
  - Independent of ISP management
  - Little network overhead, traffic
  - Trace distributed attacks, attacks post-mortem

**Marking: Assumptions**

- Most routers remain uncompromised
- Attacker sends many packets
- Route from attacker to victim remains relatively stable
Marking: Summary

- Can determine attack path with a relatively small number of attack packets
- Need to include addresses, counter in IP datagram (e.g., via fragment fields)
- E.g.: “Practical Network Support for IP Traceback” by Savage et al.
- Status:
  - Lots of RFCs
  - But not yet deployed...

Network address translation (NAT)

- Also known as
  - Network masquerading
  - IP masquerading
- Re-writes source and/or destination address as they pass through NAT gateway

- Why
  - IPv4 address shortage
  - Standard feature
  - Some believe it enhances privacy, security, ...
**Simple NAT**

(Public IP addresses)

Main Internet

NAT

(Private IP addresses)

**Multiple NAT**

156.148.70.32

(Public IP addresses)

Main Internet

ISP NAT

ISP network

192.168.2.12

192.168.2.99

Home network

10.0.0.12

(Private IP addresses)
NAT traversal: Relay

NAT traversal: Connection reversal
TURN protocol

- Protocol for UDP/TCP relaying behind NAT
- Data is bounced to a public TURN server
- No hole punching
- TURN works even behind symmetric NAT

Hole punching

- Technique to allow traffic from/to a host behind a firewall/NAT without collaboration of the NAT itself
- UDP: simple 😊
- TCP:
  - Berkeley sockets allows TCP socket to initiate an outgoing or listen for an incoming connections but not both
  - Solution: bind multiple sockets to same local endpoint
STUN (RFC 3489)

- Defines operations and message formats to understand type of NAT
- Discovers presence and type of NAT and firewalls between them and Internet
- Allows applications to determine their public NAT IP address

STUNT

- Simple Traversal of UDP Through NATs and TCP too (STUNT)
- Extends STUN to include TCP functionality
NAT traversal: Cooperating NAT

- SOCKS
  - Client server protocol
  - Enables client (behind firewall) to use server (in public Internet)
  - Relays traffic
  - Widely adopted
    - E.g.: Mozilla can use SOCKS

NAT traversal: UPnP

- Defines:
  Internet Gateway Device (IGD) protocol
- Enables:
  - Learning of ones public (external) IP address
  - Enumeration of existing port mappings
  - Adding and removing port mappings
  - Assigning lease times to mappings
  - Applications to automatically configure NAT routing