Types of Firewalls

- Packet Filters
- Dynamic Packet Filters
- Application Gateways
- Circuit Relays
- Personal and/or Distributed Firewalls

Many firewalls are combinations of these types.

Conceptual Pieces

- An “inside” — everyone on the inside is presumed to be a good guy
- An “outside” — bad guys live there
- A “DMZ” (Demilitarized Zone) — put necessary but potentially dangerous servers there
Packet Filters

- Usually Router-based (and hence cheap).
- Individual packets are accepted or rejected; no context or connection information is used.
- Advanced filter rules are hard to set up; the primitives are often inadequate, and different rules can interact.
- Packet filters a poor fit for ftp and X11.
- Hard to manage access to dynamic services.

Stateless Packet Filtering

- We want to permit out-bound connections
- We have to permit reply packets
- For TCP, this can be done without state
- The very first packet of a TCP connection has just the SYN bit set
- All others have the ACK bit set
- Solution: allow in all packets with ACK turned on

Firewall Rules Setup

- Action:
  - Permit (Pass) Allow the packet to proceed
  - Deny (Block) Discard the packet
- Direction:
  - Source (where the packet comes from)
    <IP Address, Port> or network
  - Destination (where the packet goes)
    <IP Address, Port> or network
- Protocol:
  - TCP
  - UDP
- Packet Flags:
  - ACK
  - SYN
  - RST
  - etc.

Sample Rule Set

We want to block a spammer, but allow anyone else to send email to our mail server.

```
block:  Source IP Address  =  SPAMMER
        Source Port  =  any

allow:  Source IP Address  =  any
        and
        Source Port  =  any
        and
        Destination IP Address  =  OUR-MAIL
        and
        Destination Port  =  25
```
Incorrect Rule Set

We want to allow all TCP connection to mail servers.

allow: Source IP Address = any and Source Port = 25 and Destination IP Address = any and Destination Port = any

We don’t control port number selection on the remote host. Any remote process on port 25 can call in.

Your Own Filter

Your company has decided that web browsing is not permitted for the employees. It is your task to create a filter that denies web browsing for all the machines inside the company. Assume that all the company IP addresses are known.

Outgoing packets to port 80, Web servers.

The Right Choice

allow: Source IP Address = any and Source Port = 25 and Destination IP Address = any and Destination Port = any

Flag (ACK) = Set

Permit outgoing calls.

Filtering In-bound Packets

If you filter out-bound packets to the DMZ link, you can’t tell where they came from.
UDP Filtering

- UDP has no notion of a connection. It is therefore impossible to distinguish a reply to a query—which should be permitted—from an intrusive packet.
- Address-spoofing is easy — no connections
- At best, one can try to block known-dangerous ports. But that’s a risky game.
- The safe solution is to permit UDP packets through to known-safe servers only.

UDP Example: DNS

- Accepts queries on port 53
- Block if handling internal queries only; allow if permitting external queries
- What about recursive queries?
- Bind local response socket to some other port; allow in-bound UDP packets to it
- Or put the DNS machine in the DMZ, and run no other UDP services
- (Deeper issues with DNS semantics; stay tuned)

ICMP Problems

- Often see ICMP packets in response to TCP or UDP packets
- Important example: “Path MTU” response
- Must be allowed in or connectivity can break
- Simple packet filters can’t match things up
**The Problem with RPC**

- RPC services bind to random port numbers
- There's no way to know in advance which to block and which to permit
- Similar considerations apply to RPC clients
- Systems using RPC cannot be protected by simple packet filters

**FTP, SIP, et al.**

- FTP clients (and some other services) use secondary channels
- Again, these live on random port numbers
- Simple packet filters cannot handle this
- Trying to create rules simple, packet-based rules will NOT work

**Incorrect Approach**

Block a range of UDP ports.

```
astavrou@ise:~> rpcinfo -p ise.gmu.edu
```

```
program vers proto port service
--- ---- .. .. .. ----
100000 4 tcp 111 rpcbind
100000 2 udp 111 rpcbind
390113 1 tcp 7937
100005 1 udp 32800 mountd
100005 3 tcp 32776 mountd
100003 3 udp 2049 nfs
100227 2 udp 2049 nfs_acl
100003 2 tcp 2049 nfs
100227 2 tcp 2049 nfs_acl
100011 1 udp 36613 rquotad
100008 1 udp 36614 walld
100001 2 udp 36615 rstatd
```

**Saving FTP**

- By default, FTP clients send a PORT command to specify the address for an in-bound connection
- If the PASV command is used instead, the data channel uses a separate out-bound connection
- If local policy permits arbitrary out-bound connections, this works well
The Role of Packet Filters

- Packet filters are not very useful as general-purpose firewalls
- However, they are very efficient and can be applied even in high capacity links (why?)
- Several special situations where they’re perfect
- Can be used to drop connections we don’t want to reach the more expensive application-level firewall

Application: Point Firewalls

Application: Address Filtering

- At the border router, block internal IP addresses from coming in from the outside
- Similarly, prevent address spoofing (fake IP addresses) from going out

Sample Configuration
Sample Rules

<table>
<thead>
<tr>
<th>Interface</th>
<th>Action</th>
<th>Addr</th>
<th>Port</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outside</td>
<td>Block</td>
<td>src=10.0.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Block</td>
<td>src=192.168.42.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Allow</td>
<td>dst=Mail</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Block</td>
<td>dst=DNS</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Allow</td>
<td>dst=DNS</td>
<td>UDP</td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Allow</td>
<td>Any</td>
<td>ACK</td>
<td></td>
</tr>
<tr>
<td>Outside</td>
<td>Block</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMZ</td>
<td>Block</td>
<td>src≠192.168.42.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMZ</td>
<td>Allow</td>
<td>dst=10.0.0.0/16</td>
<td>ACK</td>
<td></td>
</tr>
<tr>
<td>DMZ</td>
<td>Allow</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td>Block</td>
<td>src≠10.0.0.0/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td>Allow</td>
<td>dst=Mail</td>
<td>993</td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td>Allow</td>
<td>dst=DNS</td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td>Block</td>
<td>dst=192.168.42.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inside</td>
<td>Allow</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Stateful Packet Filters

- Most common type of packet filter
- Solves many — but not all — of the problems with simple packet filters
- Requires per-connection state in the firewall

Keeping State

- When a packet is sent out, record that in memory
- Associate in-bound packet with state created by out-bound packet
Problems Solved

- Can handle UDP query/response
- Can associate ICMP packets with connection
- Solves some of the in-bound/out-bound filtering issues — but state tables still need to be associated with in-bound packets
- Still need to block against address-spoofing

Remaining Problems

- Still have problems with secondary ports
- Still have problems with RPC
- Still have problems with complex semantics (i.e., DNS)
- The amount of state we can keep is limited

Network Address Translators (NATs)

- Translates source address (and sometimes port numbers)
- Primary purpose: coping with limited number of global IP addresses
- Sometimes marketed as a very strong firewall — is it?
- It’s not really stronger than a stateful packet filter

Basic NAT operation
Basic NAT operation

Firewalls
Stateful Packet Filtering
Stateful Packet Filters
Keeping State
Problems Solved
Remaining Problems
Network Address Translators (NATs)
Basic NAT operation
Comparison

UDP Filtering
Stateful Packet Filters
Keeping State
Problems Solved
Remaining Problems
Network Address Translators (NATs)
Basic NAT operation
Comparison

Network Address Translators (NATs)
Basic NAT operation
Comparison

Basic NAT operation

Firewalls
Stateful Packet Filtering
Stateful Packet Filters
Keeping State
Problems Solved
Remaining Problems
Network Address Translators (NATs)
Basic NAT operation
Comparison

UDP Filtering
Stateful Packet Filters
Keeping State
Problems Solved
Remaining Problems
Network Address Translators (NATs)
Basic NAT operation
Comparison

Network Address Translators (NATs)
Basic NAT operation
Comparison
### Comparison

<table>
<thead>
<tr>
<th>Stateful Packet Filter</th>
<th>NAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Out-bound</strong></td>
<td><strong>Out-bound</strong></td>
</tr>
<tr>
<td>Create state table entry.</td>
<td>Create state table entry.</td>
</tr>
<tr>
<td><strong>In-bound</strong></td>
<td><strong>In-bound</strong></td>
</tr>
<tr>
<td>Look up state table entry; drop if not present.</td>
<td>Look up state table entry; drop if not present.</td>
</tr>
</tbody>
</table>

The lookup phase and the decision to pass or drop the packet are identical; all that changes is whether or not addresses are translated.