Firewalls

What's a Firewall

- Barrier between *us* and *them*.
- Limits communication to the outside world.

⇒ The outside world can be another part of the same organization.

- Only a very few machines exposed to attack.

Limitations

A firewall is "a sort of crunchy shell around a soft, chewy center".

—Bill Cheswick, 1990

Why Use Firewalls?

- Most hosts have security holes.
  Proof: Most software is buggy. Therefore, most security software has security bugs.
- Firewalls run much less code, and hence have few bugs (and holes).
- Firewalls can be professionally (and hence better) administered.
- Firewalls run less software, with more logging and monitoring.
- They enforce the partition of a network into separate security domains.
- *Without such a partition, a network acts as a giant virtual machine, with an unknown set of privileged and ordinary users.*

Tradtitonal Firewalls by Analogy

- Passports are (generally) checked at the border.
- My office doesn't have a door direct to the outside.
- My bedroom doesn't have a real lock.
- But a bank still has a vault. . .

Should We Fix the Network Protocols Instead?

- Network security is not the problem.
- Firewalls are *not* a solution to network problems. They are a network response to a host security problem.
- More precisely, they are a response to the dismal state of software engineering; taken as a whole, the profession does not know how to produce software that is secure, correct, and easy to administer.
- Consequently, better network protocols will not obviate the need for firewalls. The best cryptography in the world will not guard against buggy code.

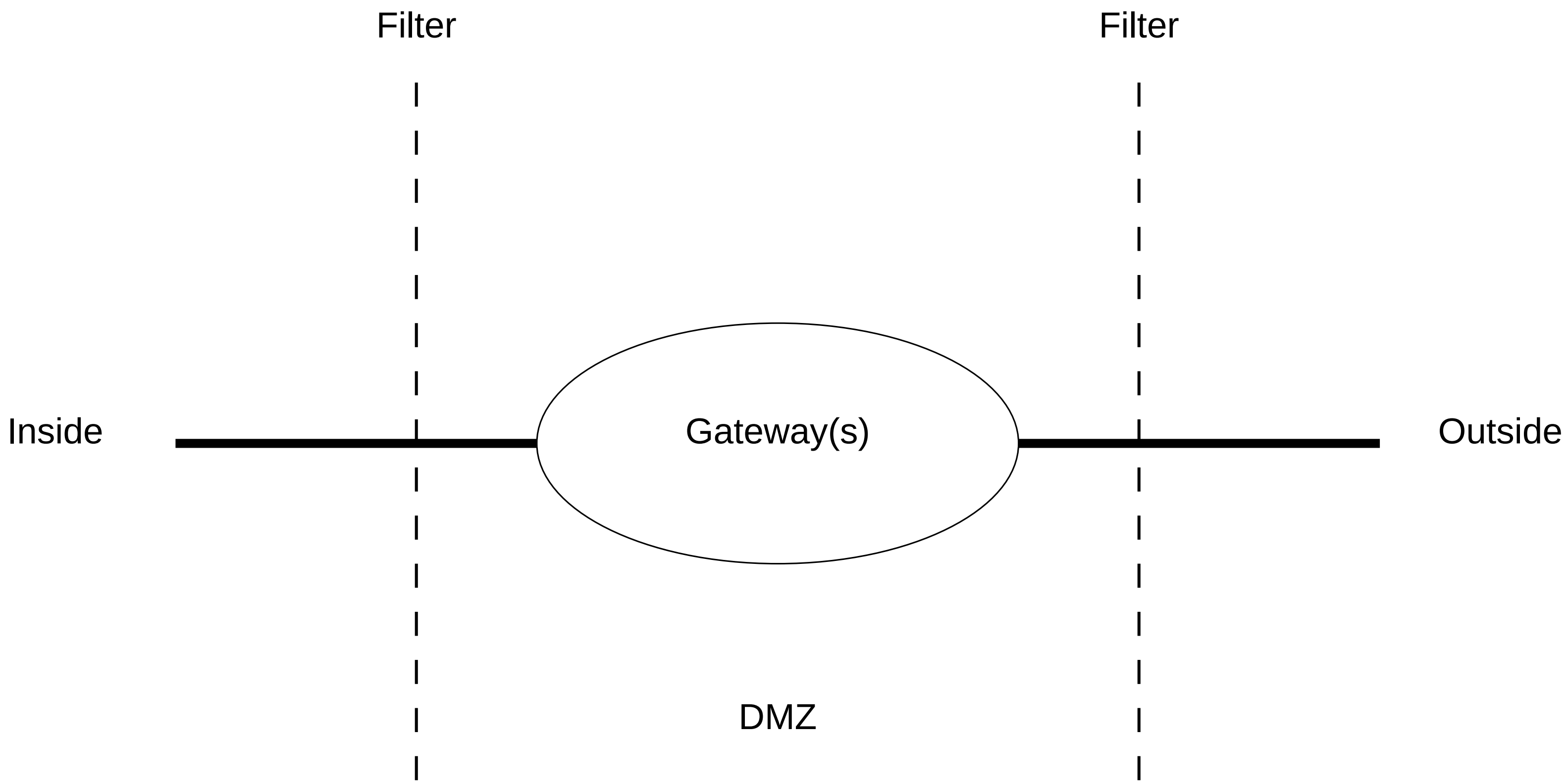
Firewall Advantages

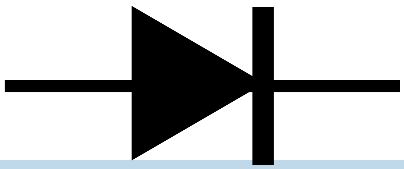
If you don't need it, get rid of it.

- No ordinary users, and hence no passowrds for them
- Run as few servers as possible
- Install conservative software, don't get the latest fancy servers, etc.)
- Log everything, and monitor the log files.
- Keep copious backups, including a "Day 0" backup.

Ordinary machines cannot be run that way.

Schematic of a Firewall





Conceptual Pieces

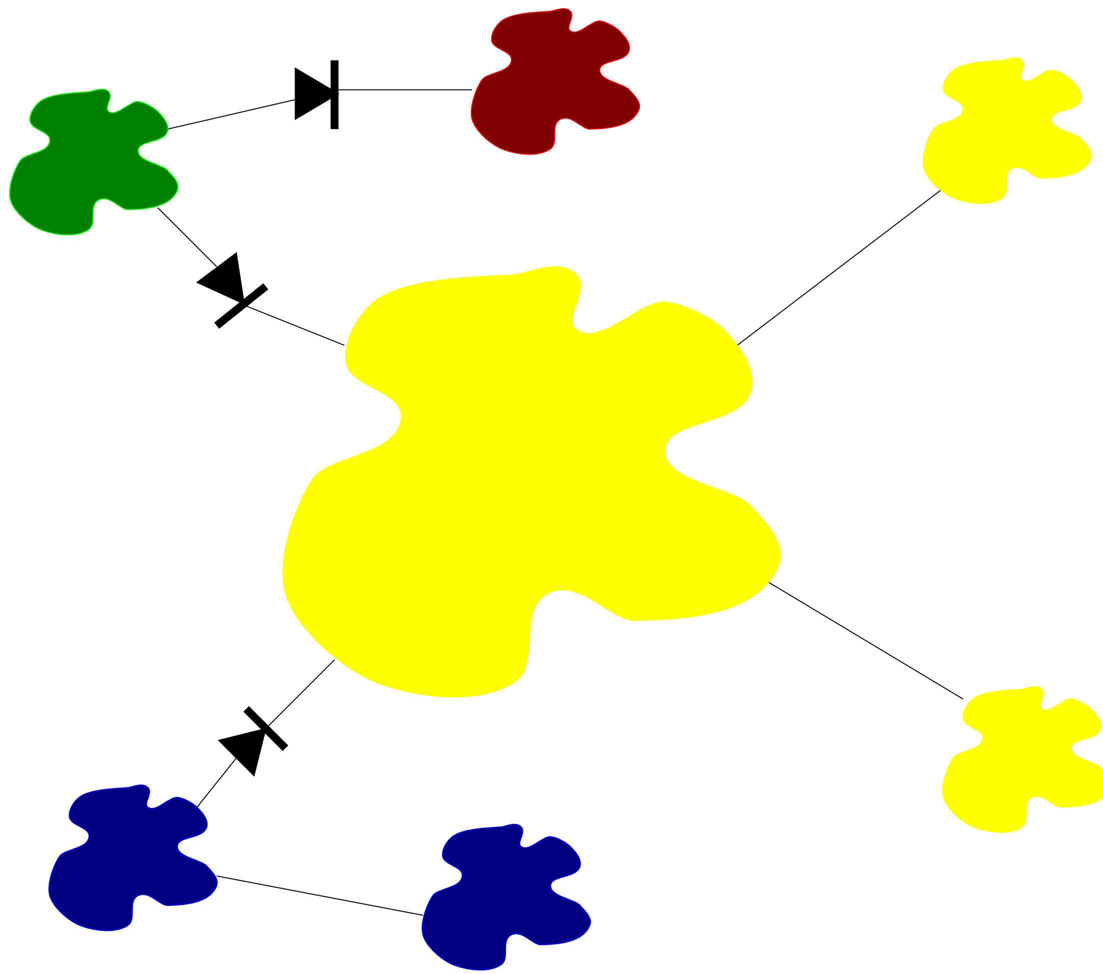
- An "inside" — everyone on the inside is presumed to be a good guy
- An "outside" — bad guys live there
- A "DMZ" (Demilitarized Zone) — put necessary but potentially dangerous servers there

The DMZ

- Good spot for things like mail and web servers
- Outsiders can send email, retrieve web pages
- Insiders can retrieve email, update web pages
- Must monitor such machines very carefully!

Positioning Firewalls

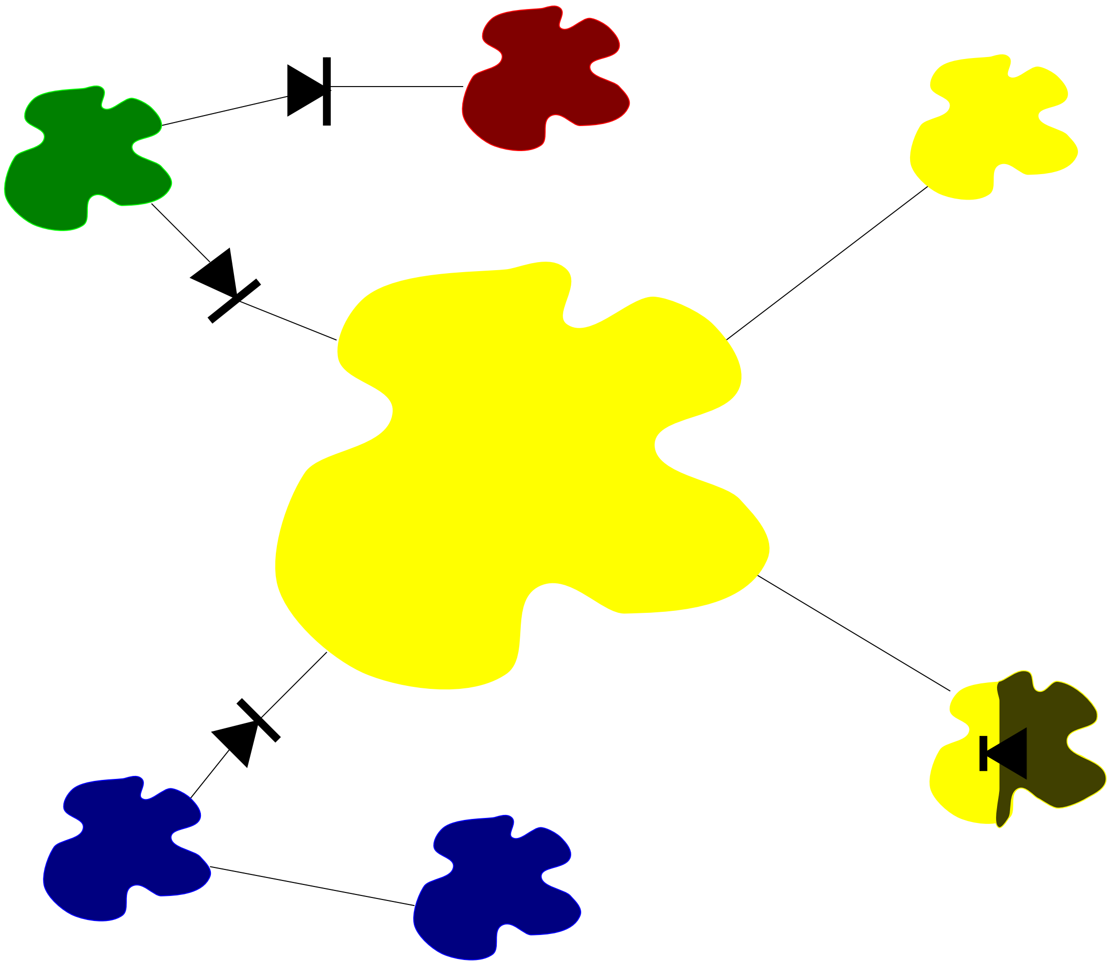
Firewalls protect *administrative* divisions.



Why Administrative Domains?

- Firewalls enforce policy
- Policy follows administrative boundaries, not physical ones
- Example: separate protection domains for Legal, HR, Research, etc.

Splitting a Location



Firewall Philosophies

1. Block all dangerous destinations.
2. Block everything; unblock things known to be both safe and necessary.

Option 1 gets you into an arms race with the attackers; you have to *know* everything that is dangerous, in all parts of your network. Option 2 is much safer.

Blocking Outbound Traffic?

- Many sites permit arbitrary outbound traffic, but. . .
- Internal bad guys?
- Extrusion detection?
- Regulatory requirements?
- Other corporate policy?

Types of Firewalls

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

Many firewalls are combinations of these types.

Packet Filters

Packet Filters

- Router-based (and hence cheap).
- Individual packets are accepted or rejected; no context is used.
- 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 RPC-based services.

Running Without State

- We want to permit outbound 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

Sample Rule Set

We want to block a spamme, but allow anyone else to send email to our gateway.

$$
\begin{aligned}
\textbf{block:}\quad & \mathit{theirhost} = \text{SPAMMER} \\
\textbf{allow:}\quad & \mathit{theirhost} = \textit{any}\ \textbf{and} \\
& \mathit{theirport} = \textit{any}\ \textbf{and} \\
& \mathit{ourhost} = \text{OUR-GW}\ \textbf{and} \\
& \mathit{ourport} = 25.
\end{aligned}
$$

Incorrect Rule Set

We want to allow all conversations with remote mail gateways.

$$\begin{aligned} \textbf{allow:}\quad theirhost &= \textit{any}\ \textbf{and} \\ theirport &= 25\ \textbf{and} \\ ourhost &= \textit{any}\ \textbf{and} \\ ourport &= \textit{any}. \end{aligned}$$

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

The Right Choice

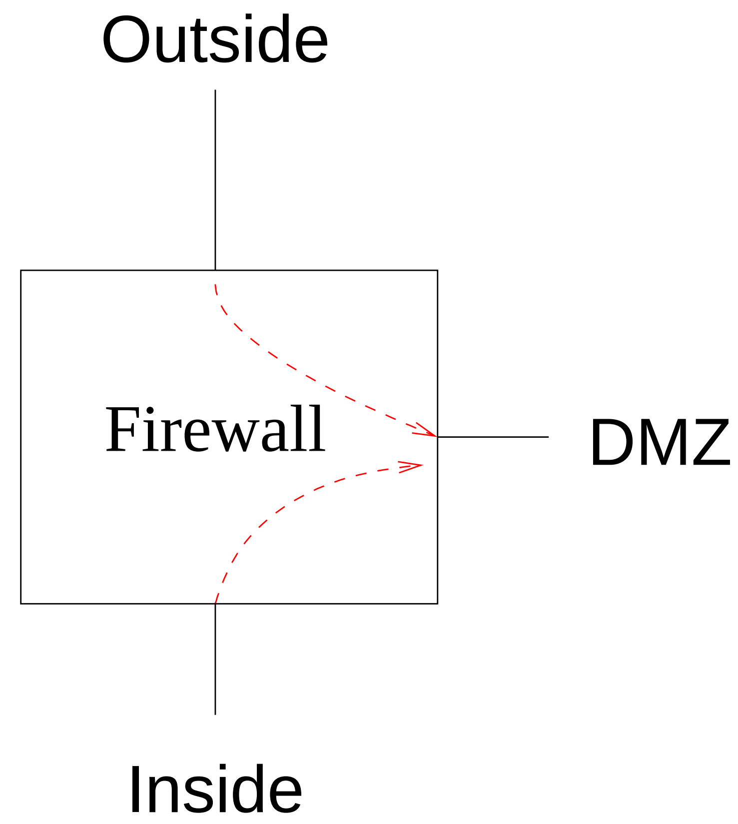
$$
\textbf{allow:} \quad \begin{aligned} theirhost &= \textit{any} \textbf{ and} \\ theirport &= 25 \textbf{ and} \\ ourhost &= \textit{any} \textbf{ and} \\ ourport &= \textit{any} \textbf{ and} \\ bitset(\texttt{ACK}) & \end{aligned}
$$

Permit *outgoing* calls.

Locating Packet Filters

- Generally have per-interface rules
- Rules are further divided to apply to inbound or outbound packets on an interface
- Better to filter inbound packets — less loss of information

Filtering Inbound Packets



If you filter outbound packets to the DMZ link, you can't tell where they came from.

Packet Filters and UDP

- 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 inbound 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

A Failed Approach

One will sometimes read "just block low-numbered UDP ports".

```
$ rpcinfo -p cluster.cs.columbia.edu
    100004    2   udp    1023   ypserv
    100004    1   udp    1023   ypserv
    100005    1   udp   32882   mountd
    100005    2   udp   32882   mountd
    100005    3   udp   32882   mountd
```

The precise patterns are implementation-specific

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

Saving FTP

- By default, FTP clients send a `PORT` command to specify the address for an inbound connection
- If the `PASV` command is used instead, the data channel uses a separate outbound connection
- If local policy permits arbitrary outbound connections, this works well

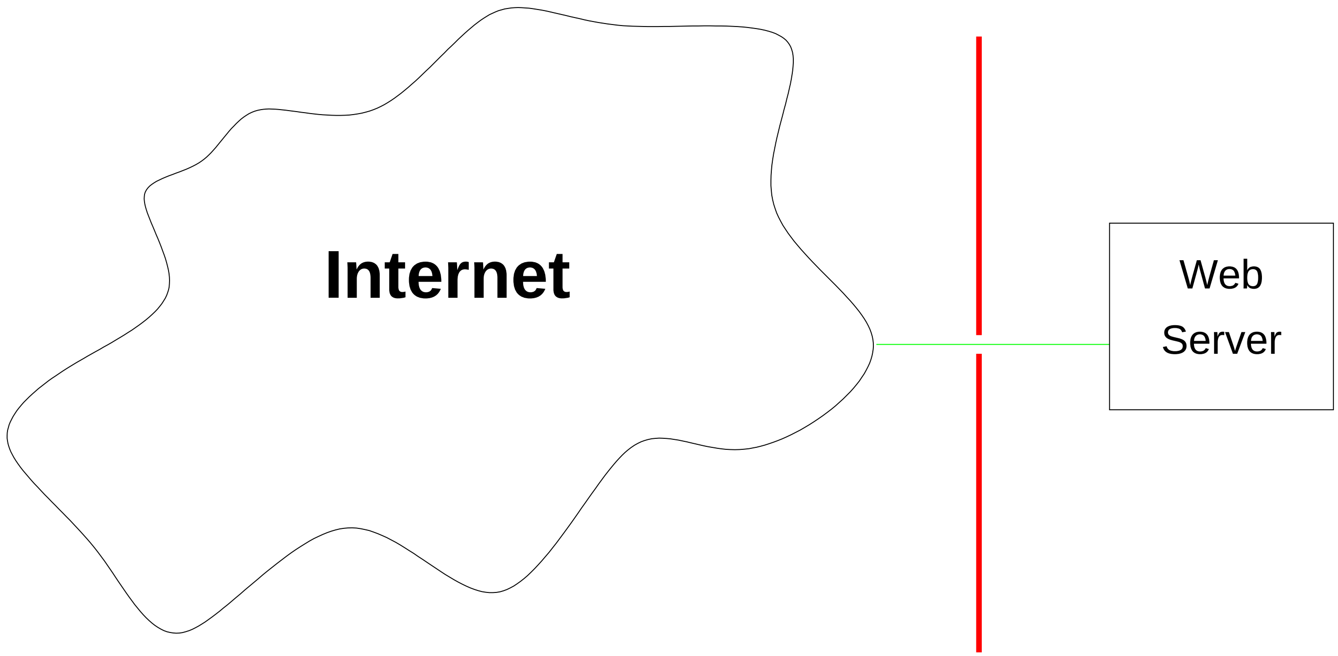
The Role of Packet Filters

- Packet filters are not very useful as general-purpose firewalls
- That said, they have their place
- Several special situations where they're perfect

Simplicity

- Packet filters are very simple, and can protect some simple environments
- Virtually all routers have the facility built in

Point Firewalls

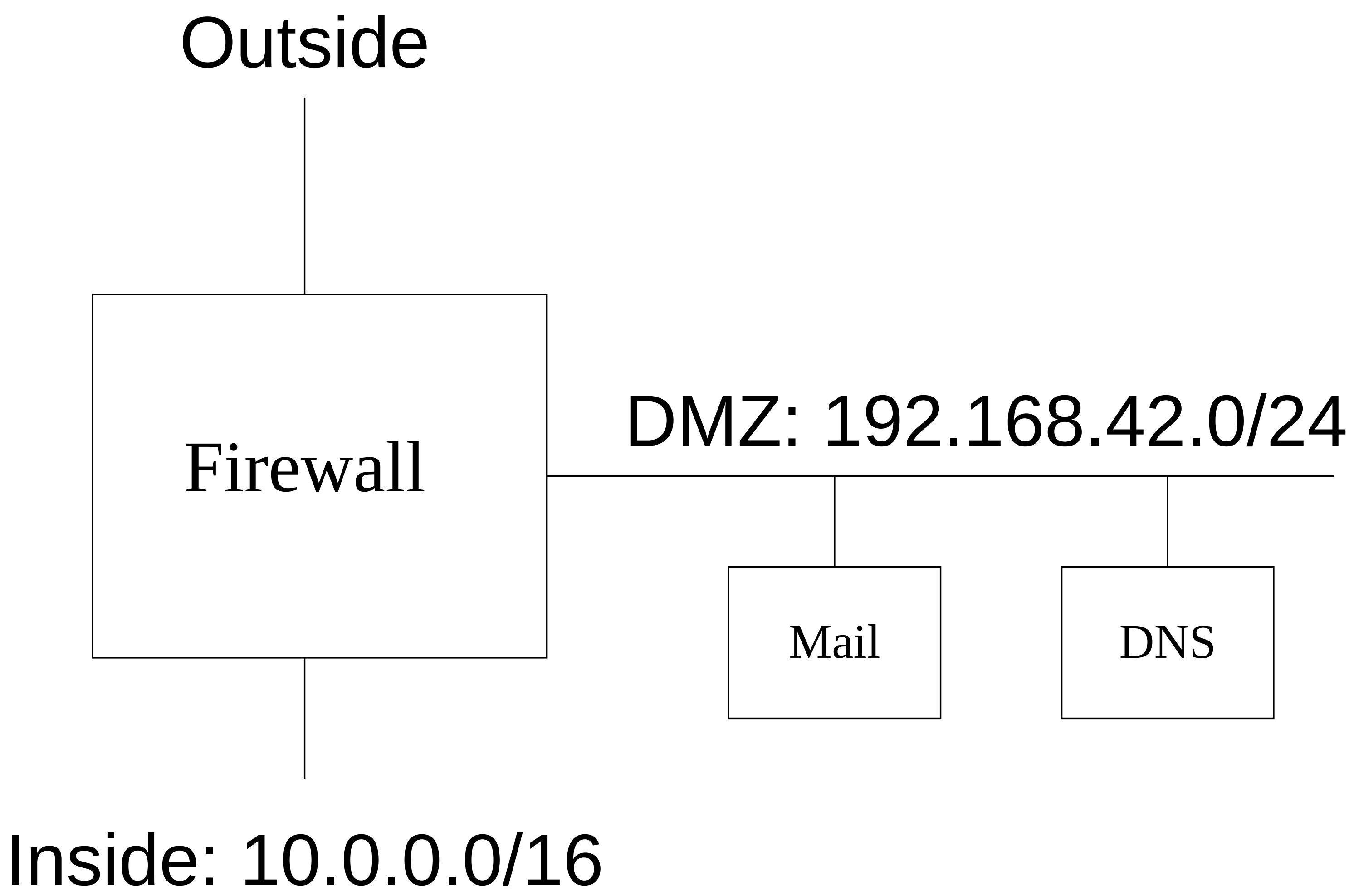


Allow in ports 80 and 443. Block *everything* else. This is a Web server appliance — it shouldn't do anything else! But — it may have necessary internal services for site administration.

Address Filtering

- At the border, block internal addresses from coming in from the outside
- Similarly, prevent fake addresses from going out

Sample Configuration



Sample Rules

Interface	*Action*	*Addr*	*Port*	*Flags*
Outside	Block	src=10.0.0.0/16		
Outside	Block	src=192.168.42.0/24		
Outside	Allow	dst=Mail	25	
Outside	Block	dst=DNS	53	
Outside	Allow	dst=DNS	UDP	
Outside	Allow	Any		ACK
Outside	Block	Any		
DMZ	Block	src≠192.168.42.0/24		
DMZ	Allow	dst=10.0.0.0/16		ACK
DMZ	Block	dst=10.0.0.0/16		
DMZ	Allow	Any		
Inside	Block	src≠10.0.0.0/16		
Inside	Allow	dst=Mail	993	
Inside	Allow	dst=DNS	53	
Inside	Block	dst=192.168.42.0/24		
Insde	Allow	Any		

Stateful Packet Filters

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
- Associate inbound packet with state created by outbound packet

Problems Solved

- Can handle UDP query/response
- Can associate ICMP packets with connection
- Solves some of the inbound/outbound filtering issues — but state tables still need to be associated with inbound 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)

Network Address Translators

- 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

Comparison

Stateful Packet Filter	NAT
Outbound Create state table entry.	**Outbound** Create state table entry. Translate address.
Inbound Look up state table entry; drop if not present .	**Inbound** Look up state table entry; drop if not present. Translate address.

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