Network Security (NetSec)

Summer 2013
Chapter 07: Selected Topics in Network Security
Module 03: Firewalls

SECURE MOBILE NETWORKING

Prof. Dr.-Ing. Matthias Hollick

Technische Universität Darmstadt
Secure Mobile Networking Lab - SEEMOO
Department of Computer Science
Center for Advanced Security Research Darmstadt - CASED

Mornewegstr. 32
D-64293 Darmstadt, Germany
Tel.+49 6151 16-70922, Fax. +49 6151 16-70921
http://seemoo.de or http://www.seemoo.tu-darmstadt.de

CASED

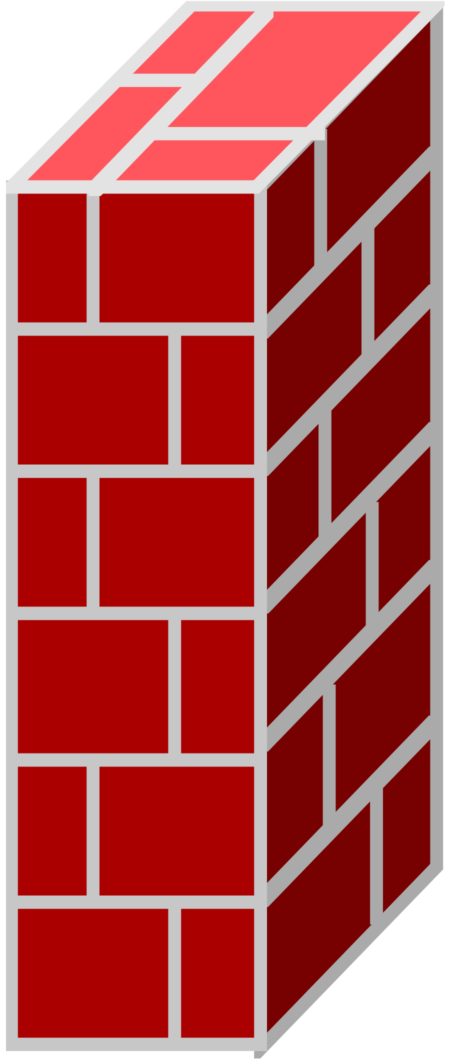
Learning Objectives

Discuss components/systems to implement network security

- What is a firewall?
  - Understand purpose, role, and architecture of firewalls
  - Detailed breakdown of firewall components
  - Limitations of firewalls
- Different types of firewalls
  - Packet filters
  - Stateful packet filters
  - Application firewalls (gateways, proxies)
  - Personal and distributed firewalls

- Outline of the chapter follows above storyline

- Chapter 07, Module 03



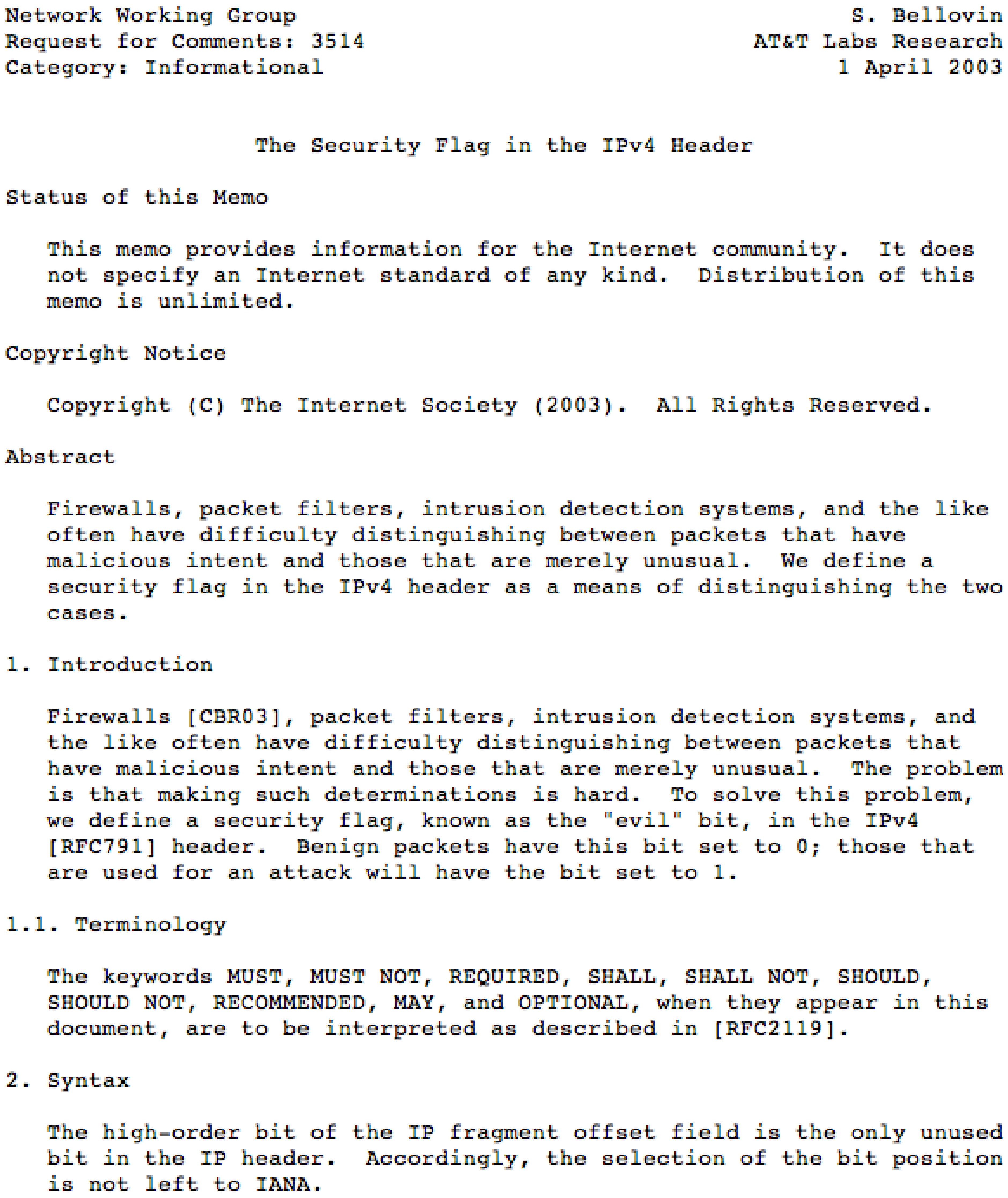
Goals of Firewalls

What are the goals of firewalls?

Goals of Firewalls

How to find out if packets are evil?

http://www.rfc-editor.org/rfc/rfc3514.txt



Network Working Group S. Bellovin
Request for Comments: 3514 AT&T Labs Research
Category: Informational 1 April 2003

The Security Flag in the IPv4 Header

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

Firewalls, packet filters, intrusion detection systems, and the like often have difficulty distinguishing between packets that have malicious intent and those that are merely unusual. We define a security flag in the IPv4 header as a means of distinguishing the two cases.

1. Introduction

Firewalls [CBR03], packet filters, intrusion detection systems, and the like often have difficulty distinguishing between packets that have malicious intent and those that are merely unusual. The problem is that making such determinations is hard. To solve this problem, we define a security flag, known as the "evil" bit, in the IPv4 [RFC791] header. Benign packets have this bit set to 0; those that are used for an attack will have the bit set to 1.

1.1. Terminology

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in [RFC2119].

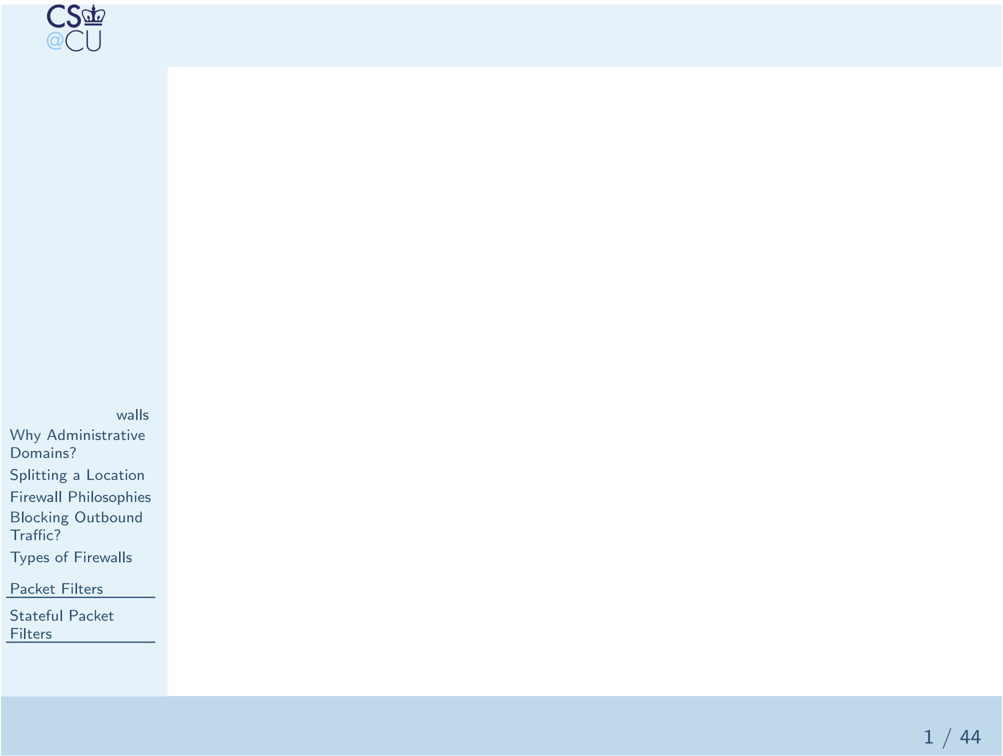
2. Syntax

The high-order bit of the IP fragment offset field is the only unused bit in the IP header. Accordingly, the selection of the bit position is not left to IANA.

Material & Recommended Reading

This year, we use an excellent slideset of Steven M. Bellovin

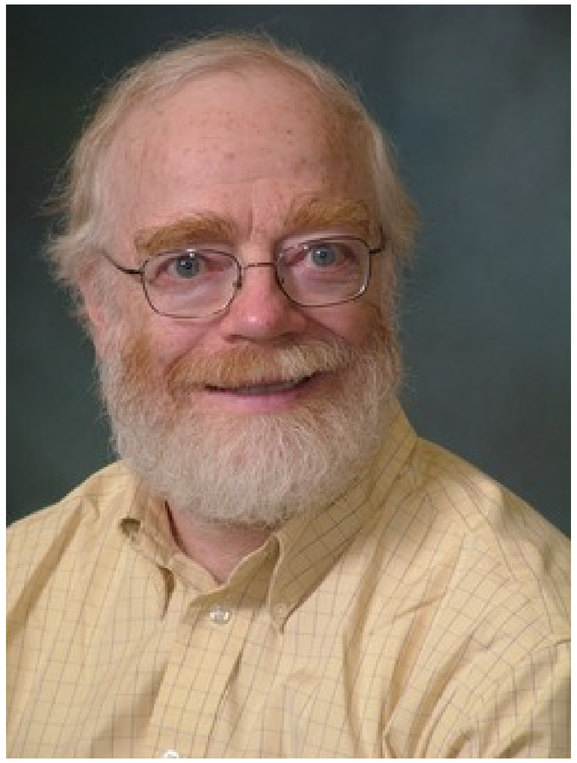
- https://www.cs.columbia.edu/~smb



CS@CU

walls
Why Administrative Domains?
Splitting a Location
Firewall Philosophies
Blocking Outbound Traffic?
Types of Firewalls
Packet Filters
Stateful Packet Filters

1 / 44



Recommended reading

- [ChBeRu] William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin. Firewalls and Internet Security: Repelling the Wily Hacker. Addison-Wesley. 2003.
- Selected papers on the homepage of Steven M. Bellovin

Firewalls

What’s a Firewall

Firewalls
What’s a Firewall
Limitations
Why Use Firewalls?
Tradtitonal Firewalls by Analogy
Should We Fix the Network Protocols Instead?
Firewall Advantages
Schematic of a Firewall
Conceptual Pieces
The DMZ
Positioning Firewalls
Why Administrative Domains?
Splitting a Location
Firewall Philosophies
Blocking Outbound Traffic?
Types of Firewalls
Packet Filters
Stateful Packet Filters

- 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.

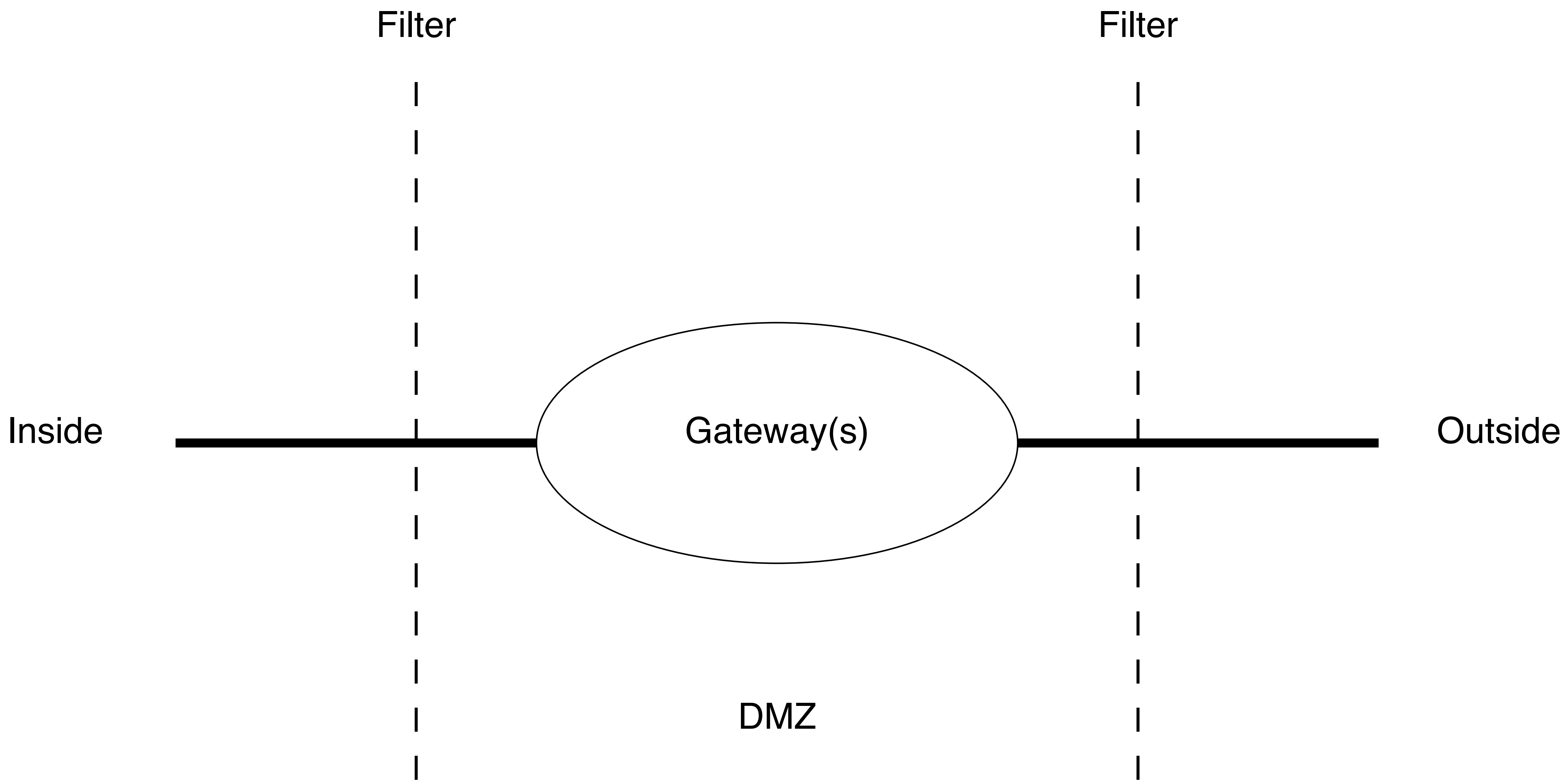
Firewall Advantages

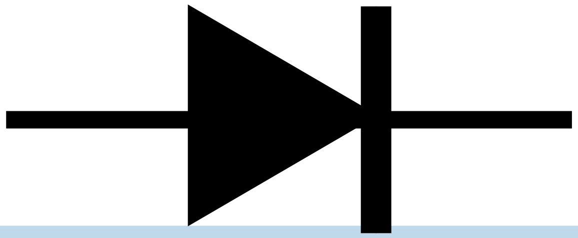
If you don't need it, get rid of it.

- 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

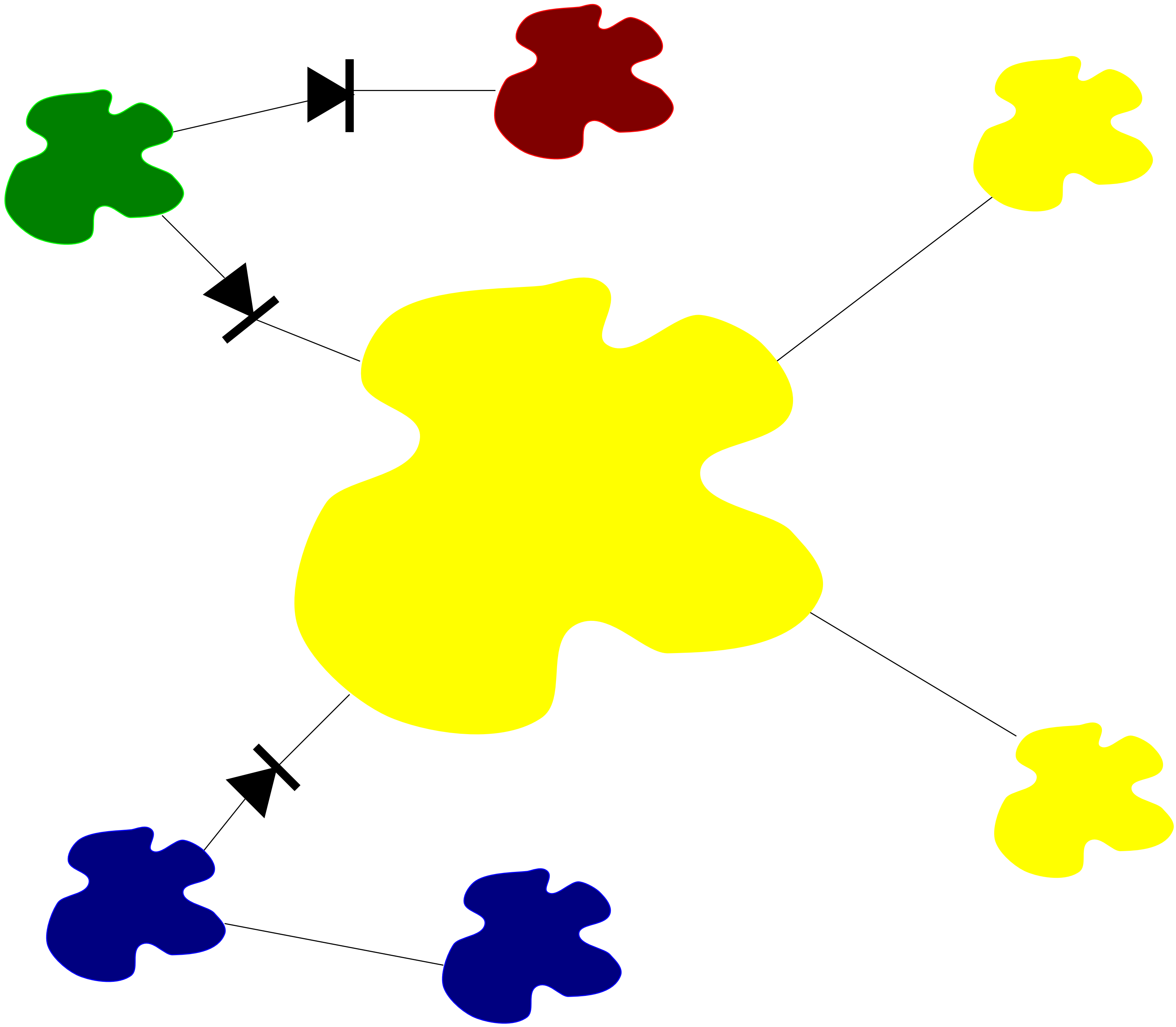
- An “inside” — everyone on the inside is presumed to be a good guy
- An “outside” — bad guys live there
- A “DMZ” (Demilitarized Zone) — put necessary but potentially dangerous servers there

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

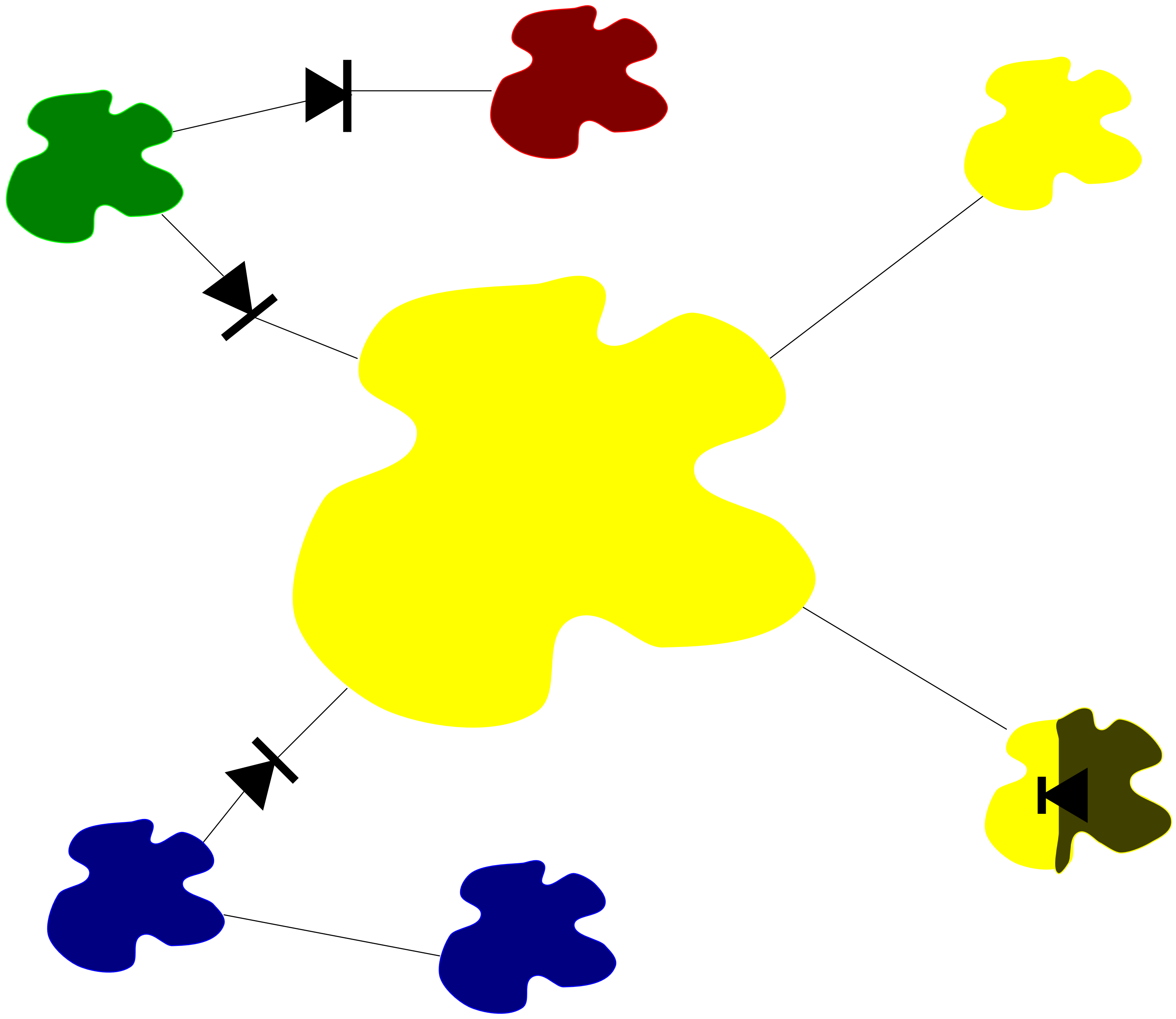
Firewalls protect *administrative* divisions.



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.

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

Incorrect Rule Set

Firewalls
Packet Filters
Packet Filters
Running Without State
Sample Rule Set
Incorrect Rule Set
The Right Choice
Locating Packet Filters
Filtering Inbound Packets
Packet Filters and UDP
UDP Example: DNS
ICMP Problems
The Problem with RPC
A Failed Approach
FTP, SIP, et al.
Saving FTP
The Role of Packet Filters
Simplicity
Point Firewalls
Address Filtering
Sample Configuration
Sample Rules
Stateful Packet Filters

We want to allow all conversations with remote mail gateways.

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

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

The Right Choice

Firewalls

Packet Filters

Packet Filters Running Without State
Sample Rule Set
Incorrect Rule Set
The Right Choice
Locating Packet Filters
Filtering Inbound Packets
Packet Filters and UDP
UDP Example: DNS
ICMP Problems
The Problem with RPC
A Failed Approach
FTP, SIP, et al.
Saving FTP
The Role of Packet Filters
Simplicity
Point Firewalls
Address Filtering
Sample Configuration
Sample Rules

Stateful Packet Filters

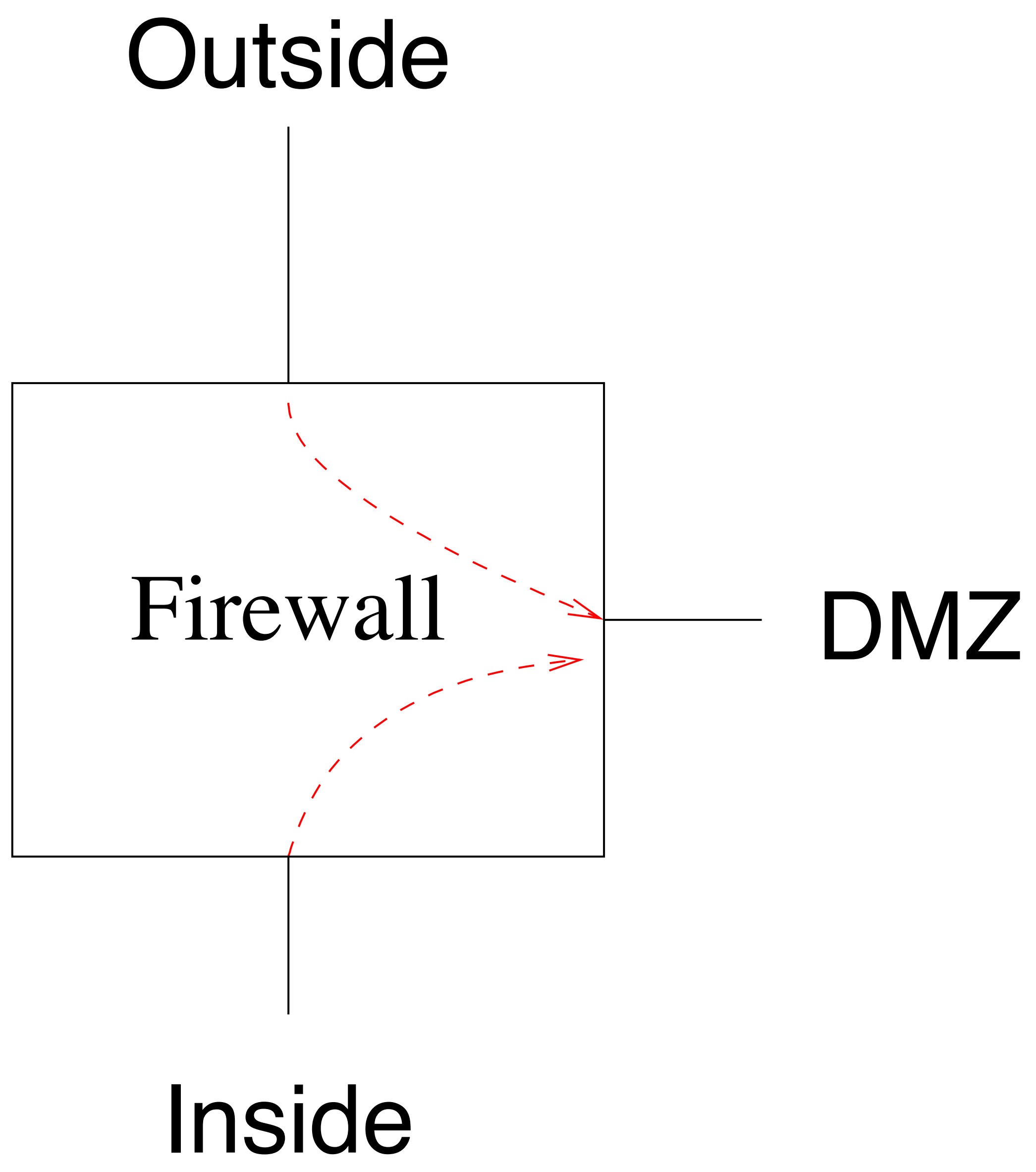
$$\begin{array}{lrcl}
\textbf{allow:} & \mathit{theirhost} & = & \textit{any}\ \textbf{and} \\
 & \mathit{theirport} & = & 25\ \textbf{and} \\
 & \mathit{ourhost} & = & \textit{any}\ \textbf{and} \\
 & \mathit{ourport} & = & \textit{any}\ \textbf{and} \\
 & \mathit{bitset}(\mathtt{ACK}) & &
\end{array}$$

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

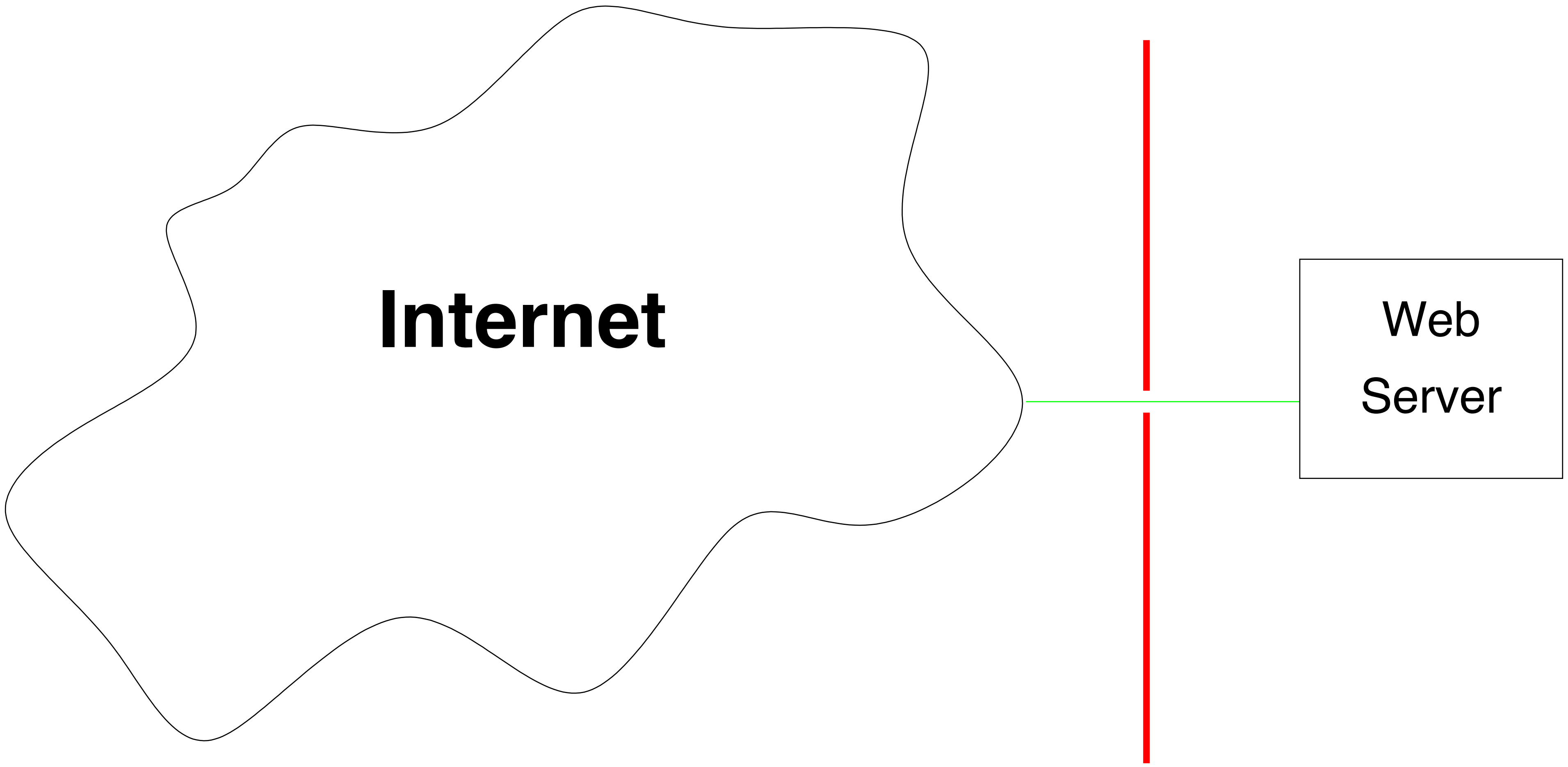
- Packet filters are not very useful as general-purpose firewalls
- That said, they have their place
- Several special situations where they're perfect

Simplicity

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

CS@CU

Point Firewalls



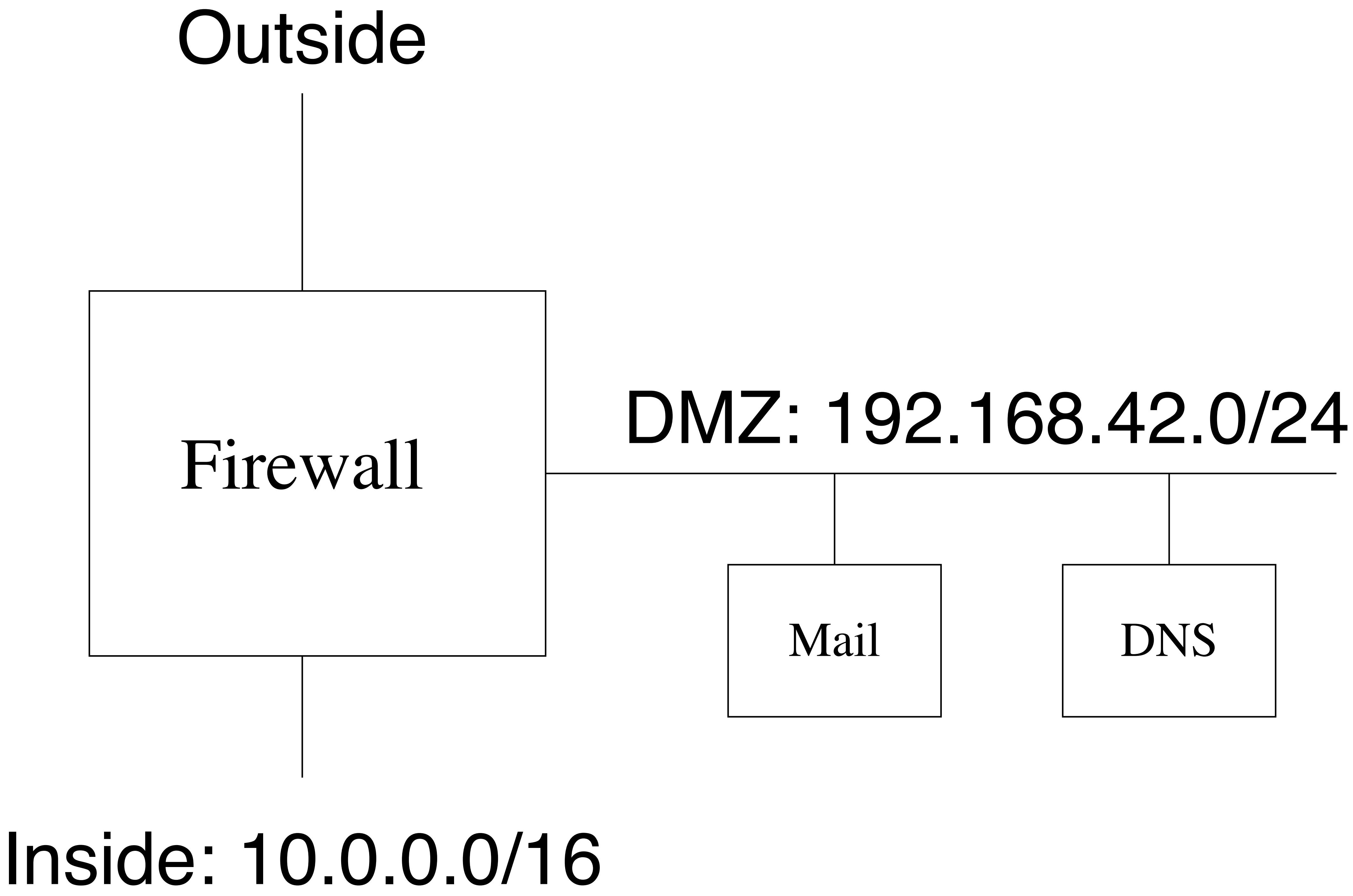
Allow in ports 80 and 443. Block *everything* else. This is a Web server appliance — it shouldn't do anything else! But — it may have necessary internal services for site administration.

CS@CU

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

Firewalls

Packet Filters

Packet Filters Running Without State
Sample Rule Set
Incorrect Rule Set
The Right Choice
Locating Packet Filters
Filtering Inbound Packets
Packet Filters and UDP
UDP Example: DNS
ICMP Problems
The Problem with RPC
A Failed Approach
FTP, SIP, et al.
Saving FTP
The Role of Packet Filters
Simplicity
Point Firewalls
Address Filtering
Sample Configuration
Sample Rules

Stateful Packet Filters

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.

Application Firewalls

Moving Up the Stack

- Why move up the stack?
- Apart from the limitations of packet filters discussed last time, *firewalls are inherently incapable of protecting against attacks on a higher layer*
- IP packet filters (plus port numbers. . .) can't protect against bogus TCP data
- A TCP-layer firewall can't protect against bugs in SMTP
- SMTP proxies can't protect against problems in the email itself, etc.

Advantages

- Protection can be tuned to the individual application
- More context can be available
- You only pay the performance price for that application, not others

Disadvantages

- Application-layer firewalls don’t protect against attacks at *lower* layers!
- They require a separate program per application
- These programs can be quite complex
- They may be very intrusive for user applications, user behavior, etc.

Example: Protecting Email

- Do we protect inbound or outbound email? Some of the code is common; some is quite different
- Do we work at the SMTP level (RFC 2821) or the mail content level (RFC 2822)?
- What about MIME?
- (What about S/MIME- or PGP-protected mail?)
- What are the threats?

Email Threats

- The usual: defend against protocol implementation bugs
- Virus-scanning
- Anti-spam?
- Javascript? Web bugs in HTML email?
- Violations of organizational email policy?
- Signature-checking?

Inbound Email

- Email is easy to intercept: MX records in the DNS route inbound email to an arbitrary machine
- Possible to use "*" to handle entire domain
- Example: DNS records exist for `att.com` and `*.att.com`
- Net result: all email for that domain is sent to a front end machine

Different Sublayers

- Note that are are multiple layers of protection possible here
- The receiving machine can run a hardened SMTP, providing protection at that layer
- Once the email is received, it can be scanned at the content layer for any threats
- The firewall function can consist of either or both

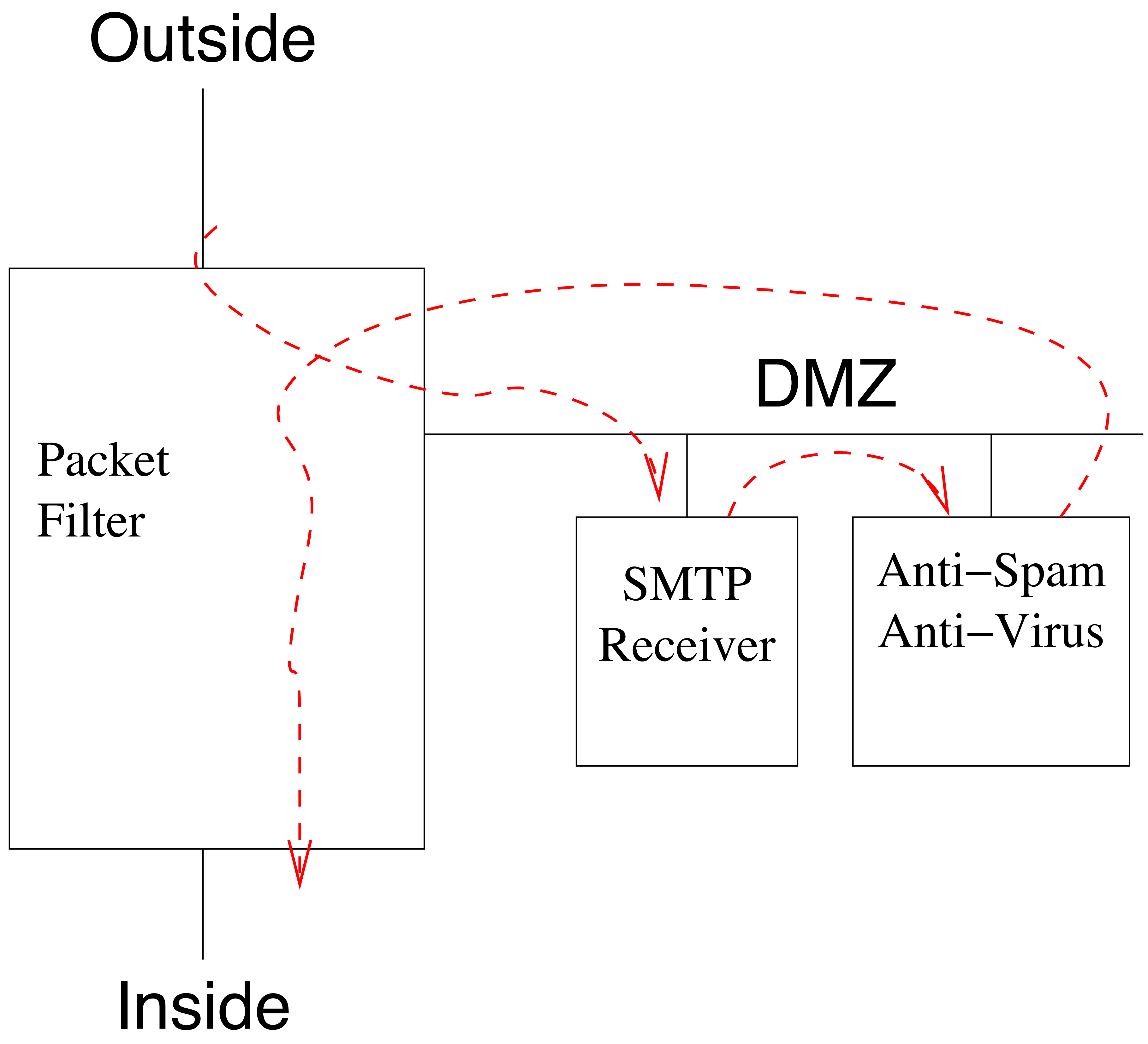
Outbound Email

- No help from the protocol definition here
- But — most MTAs have the ability to forward some or all email to a relay host
- Declare by administrative fiat that this must be done
- (Remember: in a large organization, some groups will run their own MTA.)
- Enforce this with a packet filter. . .

Combining Firewall Types

- Use an application firewall to handle inbound and outbound email
- Use a packet filter to enforce the rules

Firewalling Email



Enforcement

- Email can't flow any other way
- The only SMTP server the outside can talk to is the SMTP receiver
- It forwards the email to the anti-virus/anti-spam filter, via some arbitrary protocol
- That machine speaks SMTP to some inside mail gateway
- Note the other benefit: if the SMTP receiver is compromised, it can't speak directly to the inside

Outbound Email

- Again, we use a packet filter to block direct outbound connections to port 25
- The only machine that can speak to external SMTP receivers is the dedicated outbound email gateway
- That gateway can either live on the inside or on the DMZ

Application Proxies

Small Application Gateways

- Some protocols don’t need full-fledged handling at the application level
- That said, a packet filter isn’t adequate
- Solution: examine some of the traffic via an application-specific proxy; react accordingly

FTP Proxy

- Remember the problem with the PORT command?
- Scan the FTP control channel
- If a PORT command is spotted, tell the firewall to open that port temporarily for an incoming connection
- (Can do similar things with RPC — define filters based on RPC applications, rather than port numbers)

Attacks Via FTP Proxy

- Downloaded Java applets can call back to the originating host
- A malicious applet can open an FTP channel, and send a PORT command listing a vulnerable port on a nominally-protected host
- The firewall will let that connection through
- Solution: make the firewall smarter about what host and port numbers can appear in PORT commands. . .

Web Proxies

- Again, built-in protocol support
- Provide performance advantage: caching
- Can enforce site-specific filtering rules

Circuit Gateways

Circuit Gateways

- Circuit gateways operate at (more or less) the TCP layer
- No application-specific semantics
- Avoid complexities of packet filters
- Allow controlled inband connections, i.e., for FTP
- Handle UDP
- Most common one: SOCKS. Supported by many common applications, such as Firefox and Pidgin.

Application Modifications

- Application must be changed to speak the circuit gateway protocol instead of TCP or UDP
- Easy for open source
- Socket-compatible circuit gateway libraries have been written for SOCKS — use those instead of standard C library to convert application

Adding Authentication

- Because of the circuit (rather than packet) orientation, it's feasible to add authentication
- Purpose: extrusion control

Personal and Distributed Firewalls

Rationale

- Conventional firewalls rely on topological assumptions — these are questionable today
- Instead, install protection on the end system
- Let it protect itself

Personal Firewalls

- Add-on to the main protocol stack
- The “inside” is the host itself; everything else is the “outside”
- Most act like packet filters
- Rules can be set by individual or by administrator

Saying “No”, Saying “Yes”

- It’s easy to reject protocols you don’t like with a personal firewall
- The hard part is saying “yes” safely
- There’s no topology — all that you have is the sender’s IP address
- Spoofing IP addresses isn’t that hard, especially for UDP

Application-Linked Firewalls

- Most personal firewalls act on port numbers
- At least one such firewall is tied to applications — individual programs are or are not allowed to talk, locally or globally
- Pros: don't worry about cryptic port numbers; handle auxiliary ports just fine
- Cons: application names can be just as cryptic; service applications operate on behalf of some other application

Distributed Firewalls

- In some sense similar to personal firewalls, though with central policy control
- Use IPsec to distinguish “inside” from “outside”
- Insiders have inside-issued certificates; outsiders don’t
- Only trust other machines with the proper certificate
- No reliance on topology; insider laptops are protected when traveling; outsider laptops aren’t a threat when they visit

The Problems with Firewalls

Problems

- Corrupt insiders
- IPsec versus Firewalls
- Connectivity
- Laptops
- Evasion

IPsec versus Firewalls

- Suppose hosts routinely use IPsec to talk to the outside world.
- An inbound, ESP-protected packet arrives at the firewall.
- Should it be allowed in? Does it conform to security policies?
- The destination port number is encrypted. The ACK flag is encrypted. It might even be a tunnel mode packet.
- There is no way to for the firewall to make a decision!

Corrupt Insiders

- Firewalls assume that everyone on the inside is good
- Obviously, that's not true
- Beyond that, active content and subverted machines mean there are bad actors on the inside

Connectivity

- Firewalls rely on topology
- If there are too many conections, some will bypass the firewall
- Sometimes, that's even necessary; it isn't possible to effectively firewall all external partners
- A large company may have hundreds or even thousands of external links, most of which are unknown to the official networking people

Laptops

- Laptops, more or less by definition, travel
- When they're outside the firewall, what protects them?
- At one conference, I spotted at least a dozen other attendee machines that were infected with the Code Red virus
- (Code Red only infected web servers. Why were laptops running web servers?)

Evasion

- Firewalls and firewall administrators got too good
- Some applications weren't able to run
- Vendors started building things that ran over HTTP
- HTTP usually gets through firewalls and even web proxies. . .

Contact

Prof. Dr.-Ing. Matthias Hollick
Department of Computer Science

SEEMOO
Mornewegstr. 32
64293 Darmstadt/Germany
matthias.hollick@seemoo.tu-darmstadt.de

Phone +49 6151 16-70920
Fax +49 6151 16-70921
www.seemoo.tu-darmstadt.de

APPENDIX

Same story told in different words … the following slides are courtesy of G. Schäfer (TU Ilmenau) and provide a different angle on the same topic.

As usual, the appendices are NOT relevant for the exam, but only given for easing your access to further information.

Please be aware, that some of the terminology used here might be slightly different compared to the first part of the lecture

Introduction to Network Firewalls (1)

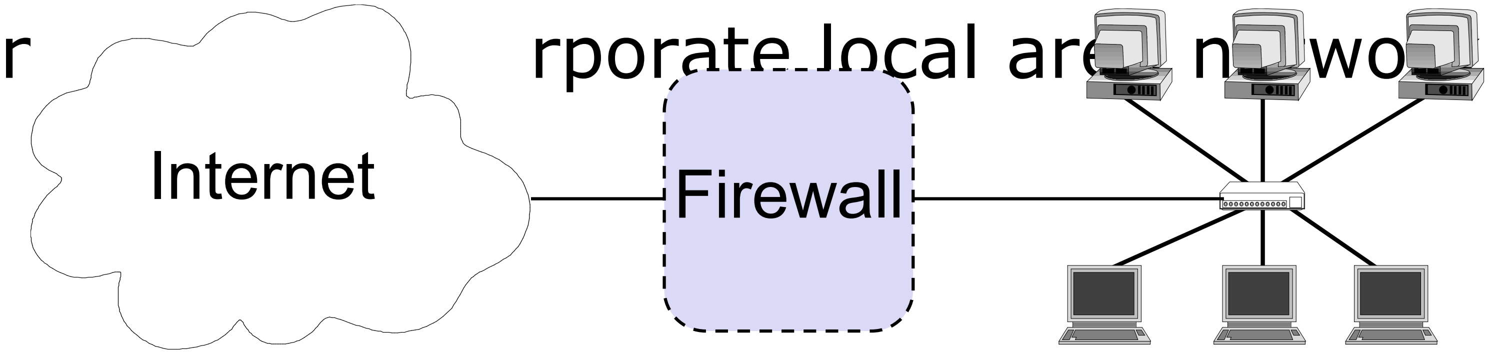
In building construction, a firewall is designed to keep a fire from spreading from one part of the building to another

A network firewall, however, can be better compared to a moat of a medieval castle:

- It restricts people to entering at one carefully controlled point
- It prevents attackers from getting close to other defenses
- It restricts people to leaving at one carefully controlled point

Usually, a network firewall is installed at a point where the protected subnetwork is connected to a less trusted network:

- Example: Conr rporate local are n wo to the Internet



- So, basically firewalls realize access control on the subnetwork level

Introduction to Network Firewalls (2)

What firewalls can do:

- A firewall is a focus for security decisions
- A firewall can enforce a security policy, i.e. concerning access control
- A firewall can log Internet activity efficiently
- A firewall can limit exposure to security problems in one part of a network

What firewalls can not do:

- A firewall can’t protect against malicious insiders
- A firewall can’t protect against connections that don’t go through it
  - If, for example, there is a modem pool behind a firewall that provides PPP service to access a subnetwork, the firewall can not provide any protection against malicious traffic from dial-in users
- A firewall can’t protect against completely new threats
- A firewall can’t fully protect against viruses
- A firewall can’t set itself up correctly (⇒ cost of operation)

Two Fundamental Approaches Regarding Firewall Policy

Default deny strategy:

- "Everything that is not explicitly permitted is denied"
- Examine the services the users of the protected network need
- Consider the security implications of these services and how the services can be safely provided
- Allow only those services that can be safely provided and for which there is a legitimate need
- Deny any other service

Default permit strategy:

- "Everything that is not explicitly forbidden is denied"
- Permit every service that is not considered dangerous
- Example:
  - Network file system (NFS) and X-Windows is not permitted across the firewall
  - Incoming telnet connections are only allowed to one specific host

What Internet Services & Protocols are to be Considered?

Electronic mail: simple mail transfer protocol (SMTP)

File exchange: file transfer protocol (FTP), network file system (NFS)

Remote terminal access and command execution: telnet, rlogin, ssh

Usenet news: network news transfer protocol (NNTP)

World wide web: hypertext transfer protocol (HTTP)

Information about people: finger

Real-time conferencing services: CUseeMe, Netmeeting, Netscape conference, MBone tools, ...

Name services: domain name service (DNS)

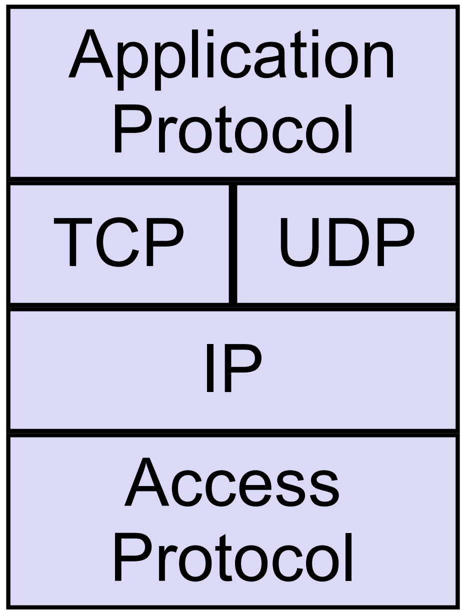
Network management: simple network management protocol (SNMP)

Time service: network time protocol (NTP)

Window systems: X-Windows

Printing systems: line printing protocols (LPR/LPD)

Some Background on Internet Services, IP, TCP & UDP



Internet services are usually realized with client and server programs and application protocols that are run by those programs

The application protocol data units are most often transported in either segments of a TCP connection or UDP datagrams

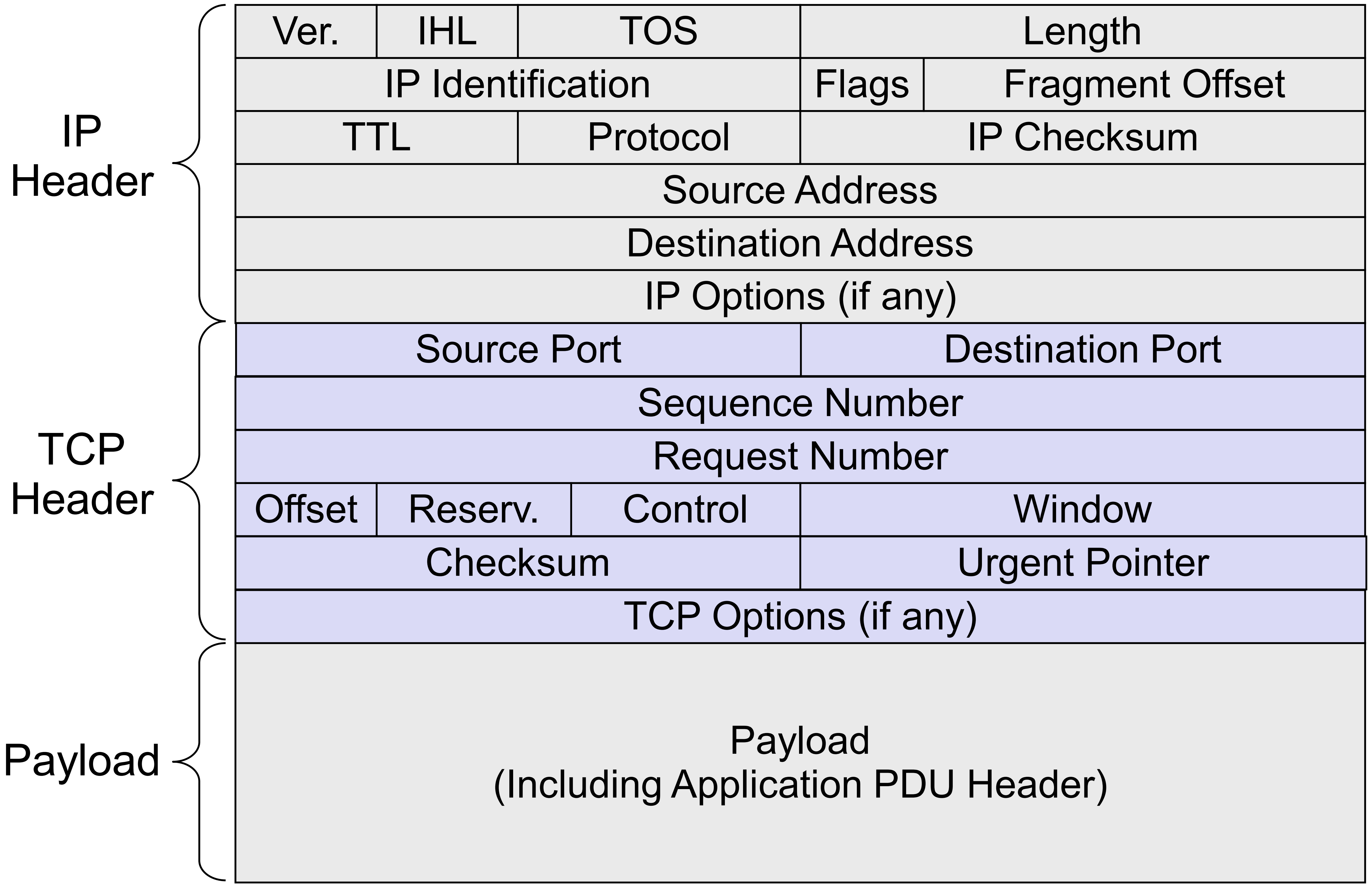
The TCP segments / UDP datagrams are transported in IP packets which themselves are transported in the PDUs of the data link technology used on the links between source and destination

- Examples: Ethernet, FDDI, ATM, etc.

The addressing of application processes (like clients, servers) is realized by the tupels:

- Source IP address, source port
- Destination IP address, destination port
- A port is a two-byte number that identifies what application process the application PDU is coming from / going to

Example: An IP Packet Carrying a TCP Segment



IP Header

Ver. | IHL | TOS | Length

IP Identification | Flags | Fragment Offset

TTL | Protocol | IP Checksum

Source Address

Destination Address

IP Options (if any)

TCP Header

Source Port | Destination Port

Sequence Number

Request Number

Offset | Reserv. | Control | Window

Checksum | Urgent Pointer

TCP Options (if any)

Payload

Payload
(Including Application PDU Header)

Protocol Fields Important for Firewalls (1)

Access Protocol:

- Network Layer Protocol: IP, Appletalk, IPX (Novell), DecNet, etc.
- Access Protocol Addresses: Ethernet MAC Address, E.164 Address, etc.
  - These addresses either refers to the final source / destination or the addresses of the intermediate nodes of this link

IP:

- Source address
- Destination address
- Flags, especially the indication of an IP fragment
- Protocol type: TCP, UDP, ICMP, ...
- Options:
  - Source routing:
    - the sender explicitly specifies the route an IP packet will take
    - as this is often used for attacks most firewalls discard these packets
  - In general, IP options are rarely used

Protocol Fields Important for Firewalls (2)

TCP:

- Source Port, Destination Port:
  - Evaluation of source and destination ports allow to determine (with a limited degree of confidence) the sending / receiving application, as most Internet services use well-known port numbers
- Control:
  - ACK: this bit is set in every segment but the very first one transmitted in a TCP connection, it therefore helps to identify connection requests
  - SYN: this bit is only set in the first two segments of a connection, so it can be used to identify connection confirmations
  - RST: if set this bit indicates an ungraceful close of a connection, it can be used to shut peers up without returning helpful error messages

Application Protocol:

- In some cases a firewall might even need to peek into application protocol header fields
- However, as this is application-dependent this class will not go into detail...

Firewall Terminology & Building Blocks for Firewalls (1)

Firewall:

- A component or a set of components that restricts access between a protected network and the Internet or between other sets of networks

Packet Filtering:

- The action a device takes to selectively control the flow of data to and from a network
- Packet filtering is an important technique to implement access control on the subnetwork-level for packet oriented networks, e.g. the Internet
- A synonym for packet filtering is screening

Bastion Host:

- A computer that must be highly secured because it is more vulnerable to attacks than other hosts on a subnetwork
- A bastion host in a firewall is usually the main point of contact for user processes of hosts of internal networks with processes of external hosts

Dual homed host:

- A general purpose computer with at least two network interfaces

Firewall Terminology & Building Blocks for Firewalls (2)

Proxy:

- A program that deals with external servers on behalf of internal clients
- Proxies relay approved client requests to real servers and also relay the servers answers back to the clients
- If a proxy interprets and understands the commands of an application protocol it is called an application level proxy, if it just passes the PDUs between the client and the server it is called a circuit level proxy

Network Address Translation (NAT):

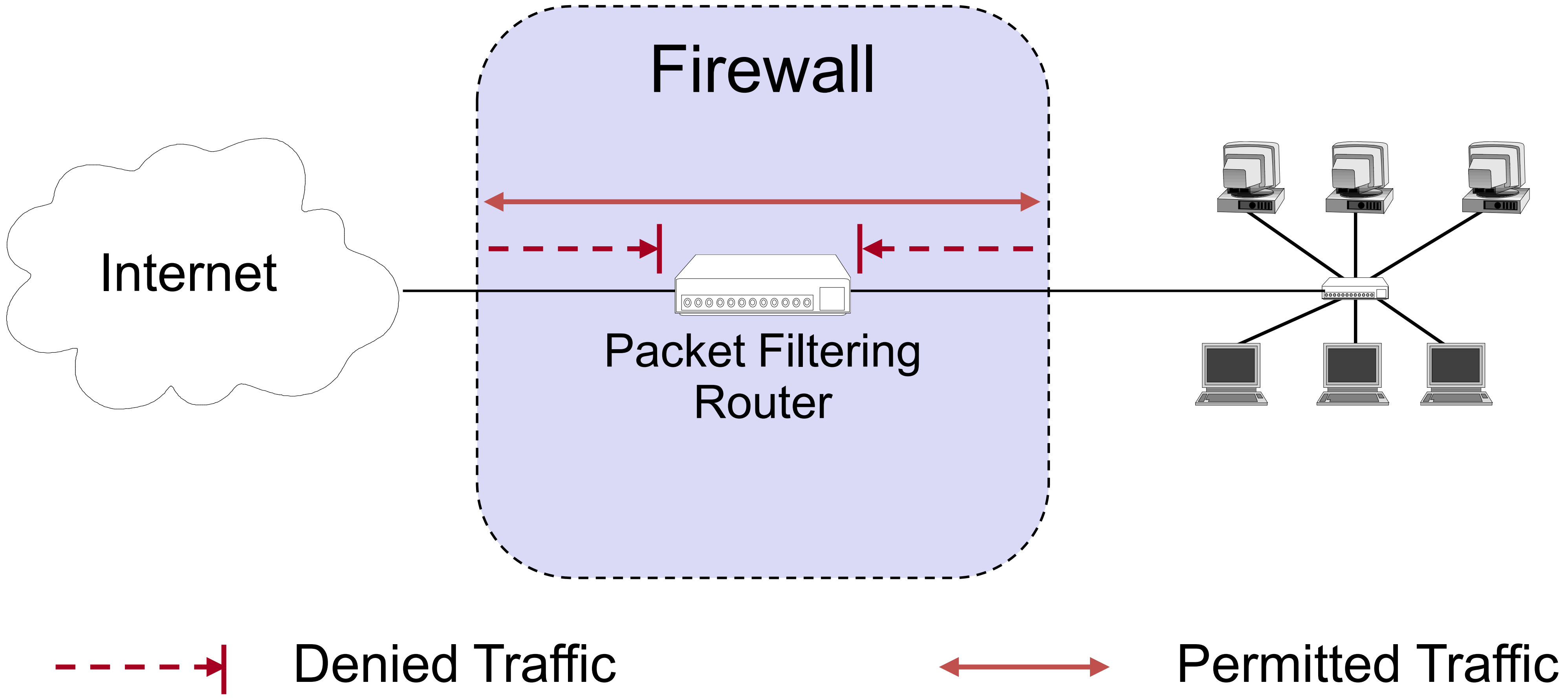
- A procedure by which a router changes data in packets to modify the network addresses
- This allows to conceal the internal network addresses (even though NAT is not actually a security technique)

Perimeter Network:

- A subnetwork added between an external and an internal network, in order to provide an additional layer of security
- A synonym for perimeter network is de-militarized zone (DMZ)

Firewall Architectures (1)

The Simple Packet Filter Architecture



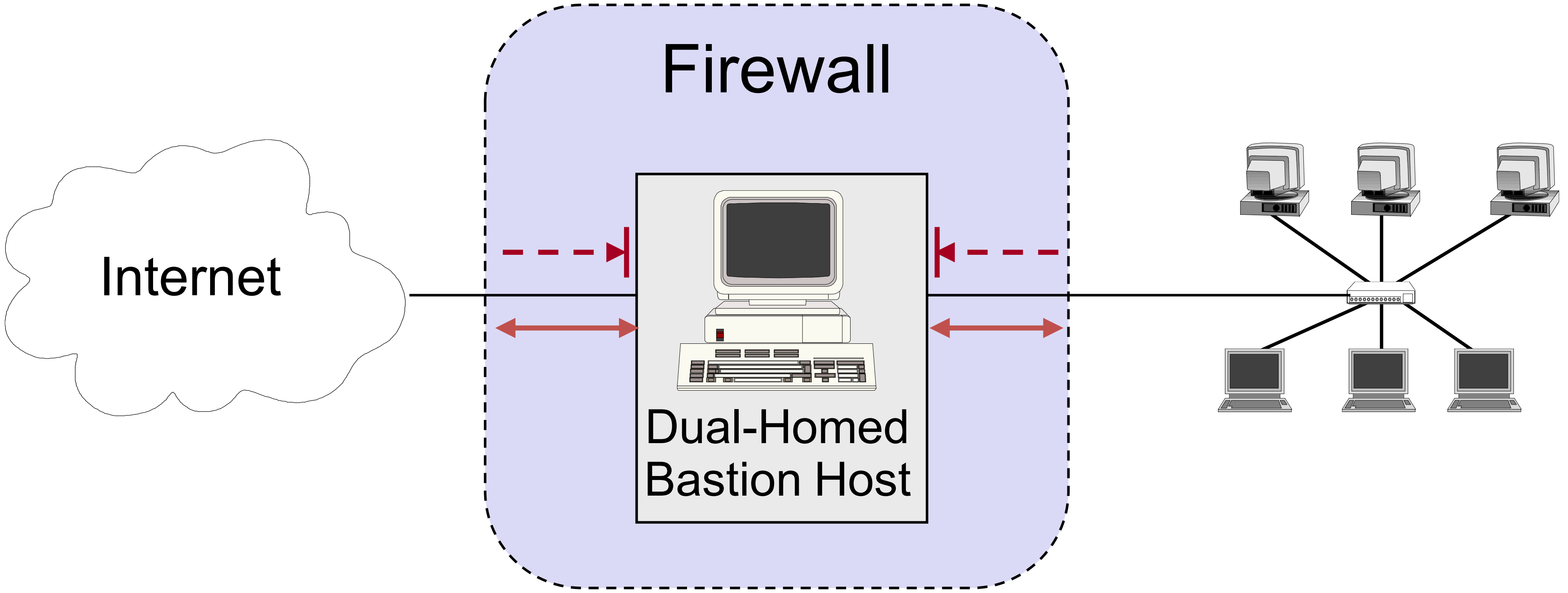
The most simple architecture just consists of a packet filtering router

It can be either realized with:

- A standard workstation (e.g. Linux PC) with at least two network interfaces plus routing and filtering software
- A dedicated router device, which usually also offers filtering capabilities

Firewall Architectures (2)

The Dual-Homed Host Architecture



The dual-homed host provides:

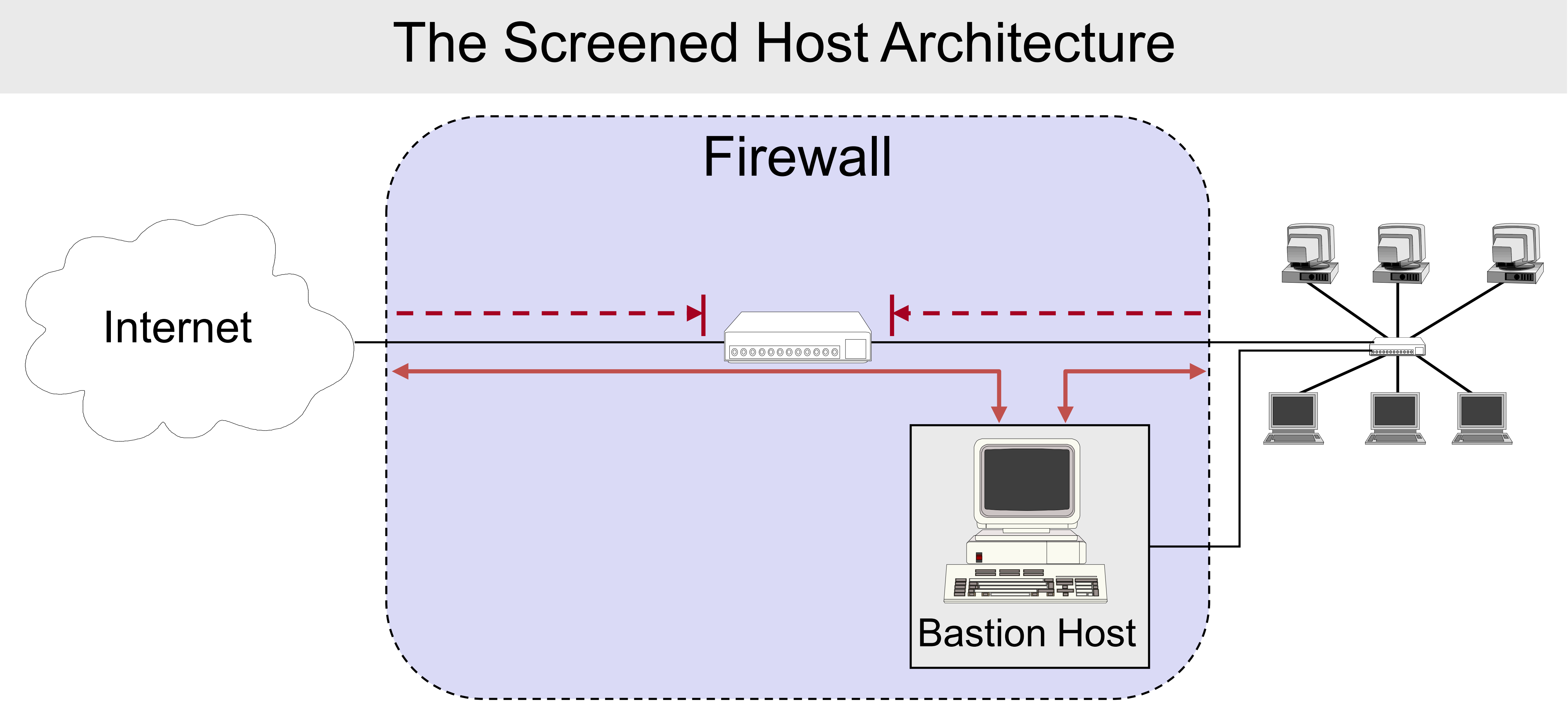
- Proxy services to internal and / or external clients
- Eventually packet filtering capabilities if it is also acting as a router

Properties of the dual-homed host:

- It has at least two network interfaces

Drawback: As all permitted traffic passes through the bastion host, this might introduce a performance bottleneck

Firewall Architectures (3)



The packet filter:

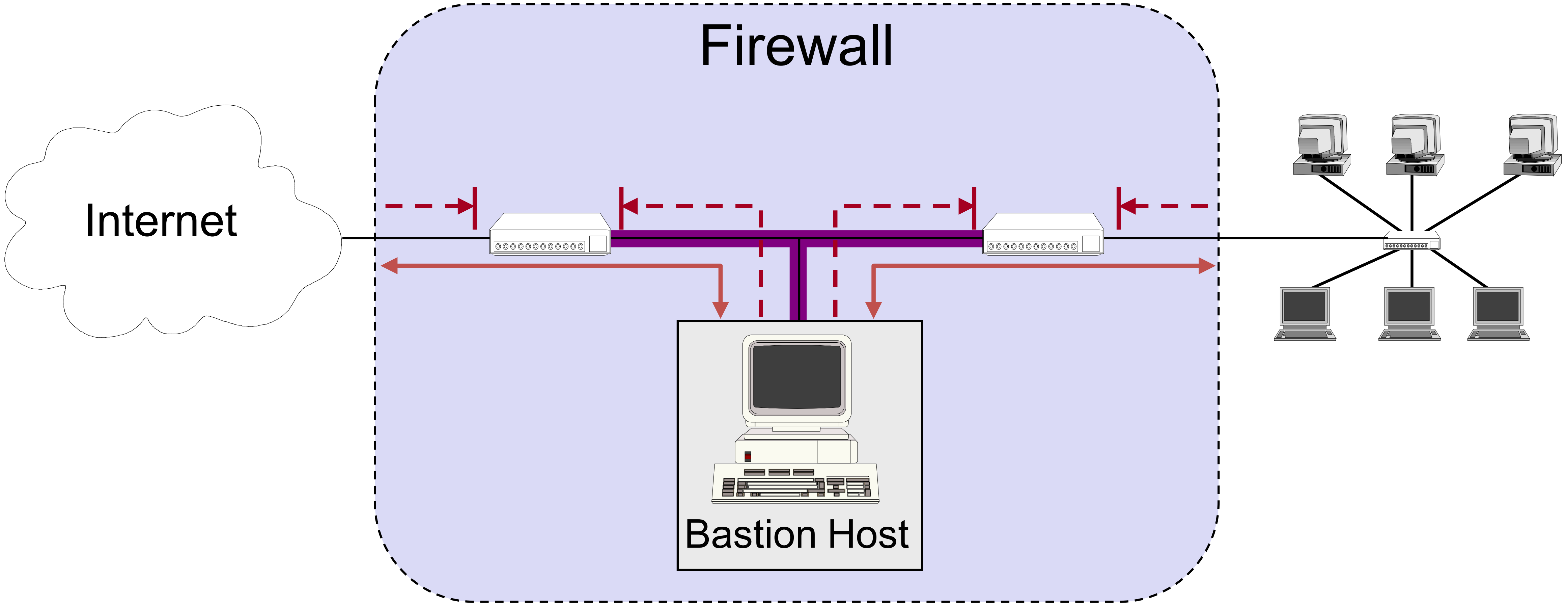
- Allows permitted IP traffic to flow between the screened host and the Internet
- Blocks all direct traffic between other internal hosts and the Internet

The screened host provides proxy services:

- Despite partial protection by the packet filter the screened host acts as a bastion host

Firewall Architectures (4)

The Screened Subnet Architecture



A perimeter network is created between two packet filters

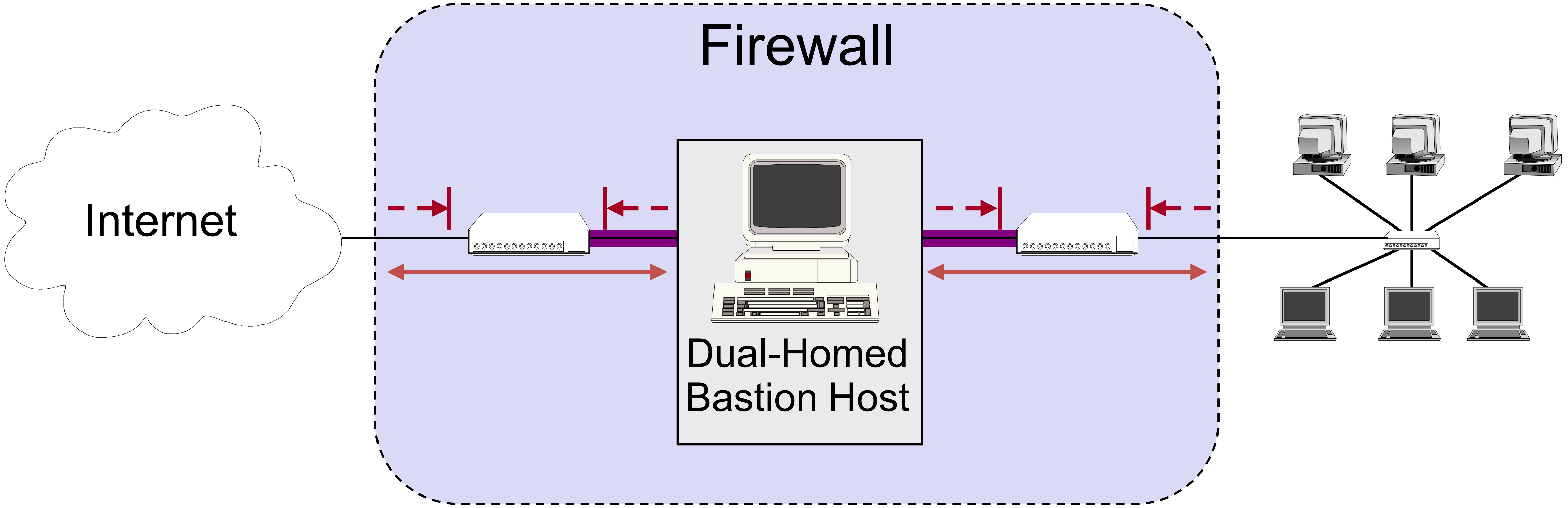
The inner packet filter serves for additional protection in case the bastion host is ever compromised:

- For example, this avoids a compromised bastion host to sniff on internal traffic

The perimeter network is also a good place to host a publicly accessible information server, e.g. a www-server

Firewall Architectures (5)

The Split Screened Subnet Architecture



A dual-homed bastion host splits the perimeter network in two distinct networks

This provides defense in depth, as:

- The dual-homed bastion host provides finer control on the connections as his proxy services are able to interpret application protocols
- The bastion host is protected from external hosts by an outer packet filter
- The internal hosts are protected from the bastion host by an inner packet filter

Packet Filtering (1)

What can be done with packet filtering?

- Theoretically speaking everything, as all information exchanged in a communication relation is transported via packets
- In practice, however, the following observations serve as a guide:
  - Operations that require quite detailed knowledge of higher layer protocols or prolonged tracking of past events are easier to realize in proxy systems
  - Operations that are simple but need to be done fast and on individual packets are easier to do in packet filtering systems

Basic packet filtering enables to control data transfer based on:

- Source IP Address
- Destination IP Address
- Transport protocol
- Source and destination application port
- Eventually, specific protocol flags (e.g. TCP’s ACK- and SYN-flag)
- The network interface a packet has been received on

Packet Filtering (2)

More elaborate packet filtering:

- Stateful or dynamic packet filtering:
  - Example 1: “Let incoming UDP packets through only if they are responses to outgoing UDP packets that have been observed”
  - Example 2: “Accept TCP packets with the SYN bit set only as part of TCP connection initiation”
- Protocol checking:
  - Example 1: “Let in packets bound for the DNS port, but only if they are formatted like DNS packets”
  - Example 2: “Do not allow HTTP transfers to these sites”
- However, more elaborate packet filtering consumes more resources!

Actions of a packet filter:

- Pass the packet
- Drop the packet
- Eventually, log the passed or dropped packet (entirely or parts of it)
- Eventually, pass an error message to the sender (may help an attacker!)

Packet Filtering (3)

Specifying packet filtering rules:

- As a packet filter protects one part of a network from another one, there is an implicit notion of the direction of traffic flow:
  - Inbound: The traffic is coming from an interface which is outside the protected network and its destination can be reached on an interface which is connected to the protected network
  - Outbound: the opposite of inbound
  - For every packet filtering rule this direction is specified as either “inbound”, “outbound”, or “either”
- Source and destination address specifications can make use of wildcards, e.g. 125.26.*.* denotes all addresses starting with 125.26.
  - In our examples, we denote often simply denote addresses as “internal” or “external” when we want to leave exact network topology out of account
- For source and destination ports we sometimes write ranges, e.g. “>1023”
- We assume filtering rules to be applied in the order of specification, that means the first rule that matches a packet is applied

An Example Packet Filtering Ruleset (1)

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP		25		Permit
B	Outbound	Internal	External	TCP		>1023		Permit
C	Outbound	Internal	External	TCP		25		Permit
D	Inbound	External	Internal	TCP		>1023		Permit
E	Either	Any	Any	Any		Any		Deny

This first ruleset aims to specify, that incoming and outgoing email should be the only allowed traffic into and out of a protected network

Email is relayed between two servers by transferring it to an SMTP-daemon on the target server (server port 25, client port > 1023)

Rule A allows incoming email to enter the network and rule B allows the acknowledgements to exit the network

Rules C and D are analogous for outgoing email

Rule E denies all other traffic

An Example Packet Filtering Ruleset (2)

Consider, for example, a packet which “wants” to enter the protected subnet and has a forged IP source address from the internal network:

- As all allowed inbound packets must have external source and internal destination addresses (A, D) this packet is successfully blocked
- The same holds for outbound packets with external source addresses (B, C)

Consider now telnet traffic:

- As a telnet server resides usually at port 23, and all allowed inbound traffic must be either to port 25 or to a port number > 1023, incoming packets to initiate an incoming telnet connection are successfully blocked
- The same holds for outgoing telnet connections

However, the ruleset is flawed as, for example, it does not block the X11-protocol for remote operation of X-Windows applications:

- An X11-server usually listens at port 6000, clients use port numbers > 1023
- Thus, an incoming X11-request is not blocked (B), neither is any answer (D)
- This is highly undesirable, as the X11-protocol offers many vulnerabilities to an attacker, like reading and manipulating the display and keystrokes

An Example Packet Filtering Ruleset (3)

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP	>1023	25		Permit
B	Outbound	Internal	External	TCP	25	>1023		Permit
C	Outbound	Internal	External	TCP	>1023	25		Permit
D	Inbound	External	Internal	TCP	25	>1023		Permit
E	Either	Any	Any	Any	Any	Any		Deny

The above flaw can be fixed by including the source ports into the ruleset specification:

- As now outbound traffic to ports >1023 is allowed only if the source port is 25 (B), traffic from internal X-clients or -servers (port >1023) will be blocked
- The same holds for inbound traffic to ports >1023 (D)

However, it can not be assumed for sure, that an attacker will not use port 25 for his attacking X-client:

- In this case the above filter will let the traffic pass

An Example Packet Filtering Ruleset (4)

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Internal	TCP	>1023	25	Any	Permit
B	Outbound	Internal	External	TCP	25	>1023	Yes	Permit
C	Outbound	Internal	External	TCP	>1023	25	Any	Permit
D	Inbound	External	Internal	TCP	25	>1023	Yes	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

This problem can be addressed by also specifying TCP's ACK-bit in rules B and D:

- As the ACK-bit is required to be set in rule B, it is not possible to open a new TCP connection in the outbound direction to ports >1023, as TCP's connect-request is signaled with the ACK-bit not set
- The same holds for the inbound direction, as rule D requires the ACK bit to be set

As a basic rule, any filtering rule that permits incoming TCP packets for outgoing connections should require the ACK-bit be set

An Example Packet Filtering Ruleset (5)

If the firewall comprises a bastion host, the packet filtering rules should further restrict traffic flow (→ screened host architecture):

- As in the modified rules above only traffic between the Internet and the bastion host is allowed, external attackers can not attack SMTP on arbitrary internal hosts any longer

In a screened subnet firewall, two packet filtering routers are set up:

- one for traffic allowed between the Internet and the bastion host, and
- one for traffic allowed between the bastion host and the internal network

Rule	Direction	Src. Addr.	Dest. Addr.	Protocol	Src. Port	Dest. Port	ACK	Action
A	Inbound	External	Bastion	TCP	>1023	25	Any	Permit
B	Outbound	Bastion	External	TCP	25	>1023	Yes	Permit
C	Outbound	Bastion	External	TCP	>1023	25	Any	Permit
D	Inbound	External	Bastion	TCP	25	>1023	Yes	Permit
E	Either	Any	Any	Any	Any	Any	Any	Deny

Bastion Hosts (1)

A bastion host is defined as a host that is more exposed to the hosts of an external network than the other hosts of the network it protects

A bastion host may serve for different purposes:

- Packet filtering
- Providing proxy services
- A combination of both

The principles for building a bastion hosts are extensions of those for securing any mission critical host:

- Keep it simple
- Prepare for the bastion host to be compromised:
  - Internal hosts should not trust it any more than is absolutely required
  - If possible, it should be connected in a way to the network so that it can not sniff on internal traffic
  - Provide extensive logging for incident detection / analysis, if possible such that it can not be easily tampered with even when the host is compromised

Bastion Hosts (2)

Further guidelines:

- Make the bastion host unattractive:
  - Slower machines are less appealing targets and are less useful if compromised
  - However, if the bastion host offers some resource consuming service, e.g. WWW-service, it may be wiser not to make it too slow
  - The fewer tools are available on the bastion host, the less useful the machine is to an attacker
- Get a reliable hardware configuration (no leading / bleeding edge)
- The bastion host should be placed at a physically secure location
- Disable any user accounts on the bastion host
- Secure the system logs (by writing them directly to a printer or even better using the printer port to a dedicated PC which is not networked)
- Do regular backups of the system logs and the configuration (using a dedicated backup device)
- Monitor the machine closely (reboots, usage / load patterns, etc.)
- If possible, restore the machine regularly from a prepared installation

Proxy Services

Proxying provides access to a specific Internet service for a single host, while appearing to provide it for all hosts of a protected network

Candidate services for proxying:

- FTP, Telnet, DNS, SMTP, HTTP

Proxy servers usually run on (possibly dual-homed) bastion hosts

The use of a proxy service usually leads to the following situation:

- The user of a proxy service has the illusion of exchanging data with the actual server host
- The actual server has the illusion of exchanging data with the proxy host

In order to instruct the proxy service to which server it should connect to, one of the following approaches can be taken:

- Proxy-aware client software
- Proxy-aware operating system
- Proxy-aware user procedures
- Proxy-aware router