TECHNISCHE UNIVERSITÄT DRESDEN

Operating Systems Group

Distributed Operating Systems

Firewalls

Dresden, 2008-05-21

Agenda

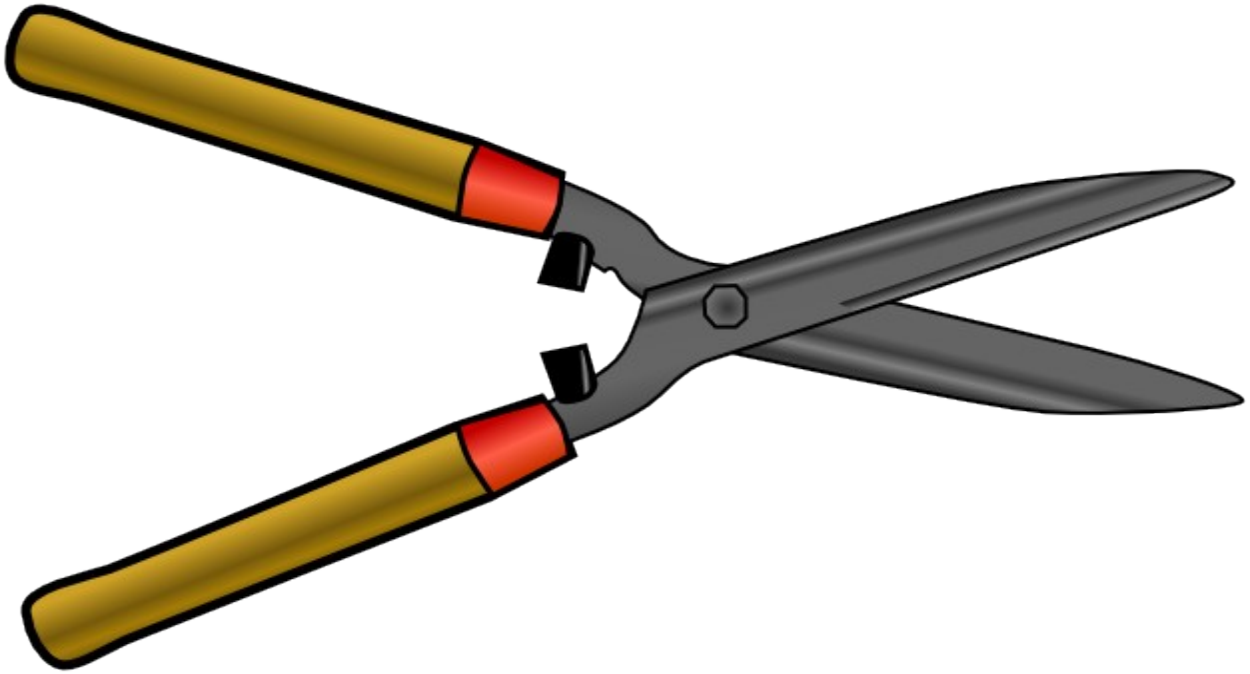
- Introduction
- What to protect? Where to intercept?
- Firewalls:
  - Packet filters
  - Application firewalls
- Firewall practices
  - Security/Network policies
  - NAT
- Further techniques
- Example: Setting up a simple scenario with iptables

Basic Idea

- Protect different networks from each other.
- A Firewall is a device that connects different networks and is configured to permit, deny or proxy data between those.
- Primary usage: The Internet
  - Protect your local network from the (evil) Internet

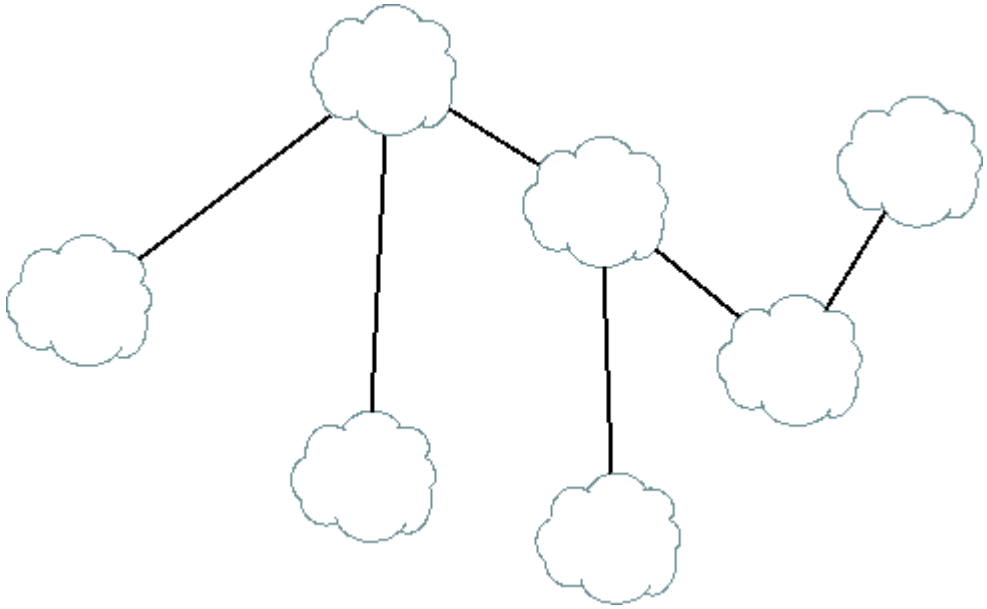
The most Secure Firewall?

The most Secure Firewall:



The Internet?

- Network of networks, interconnected, directly or indirectly.



How Does it Work?

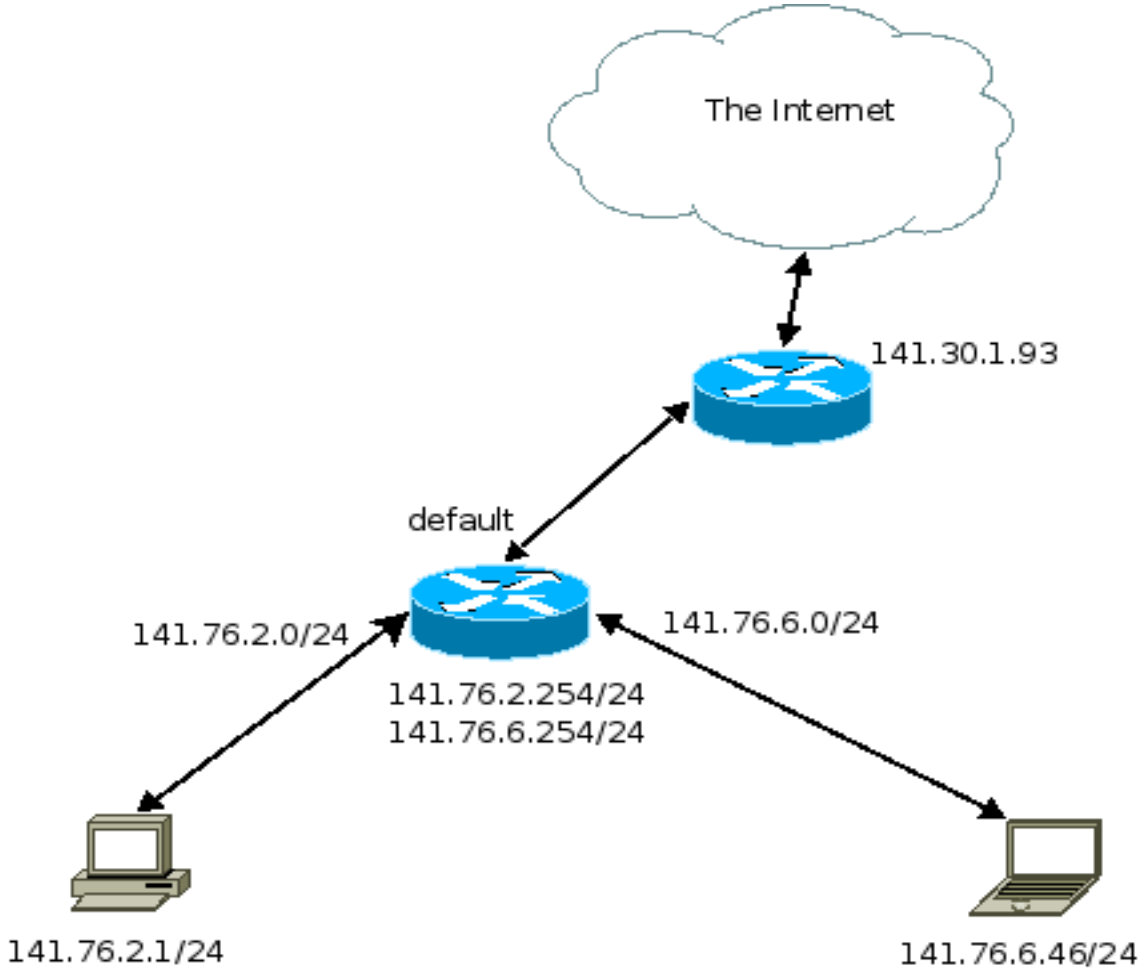
- Each node on the Internet has a unique ID, the IPv4 address.
  - 32 bit
  - Notation: dotted decimals of 8 bits:
    - e.g. 141.76.2.1
  - IPv6: 128 bits, notation: 2001:6b0:e:2018::137

How do Packets Travel Through the net?

- Netmask, associated to each IP address
  - Specifies which packets are transferred locally and which packets must be sent via the router
  - Example1: 141.76.2.1/24
    - 141.76.2.X are local, all others not
  - Example2: 192.168.5.37/28
    - 192.168.5.32-47 are local

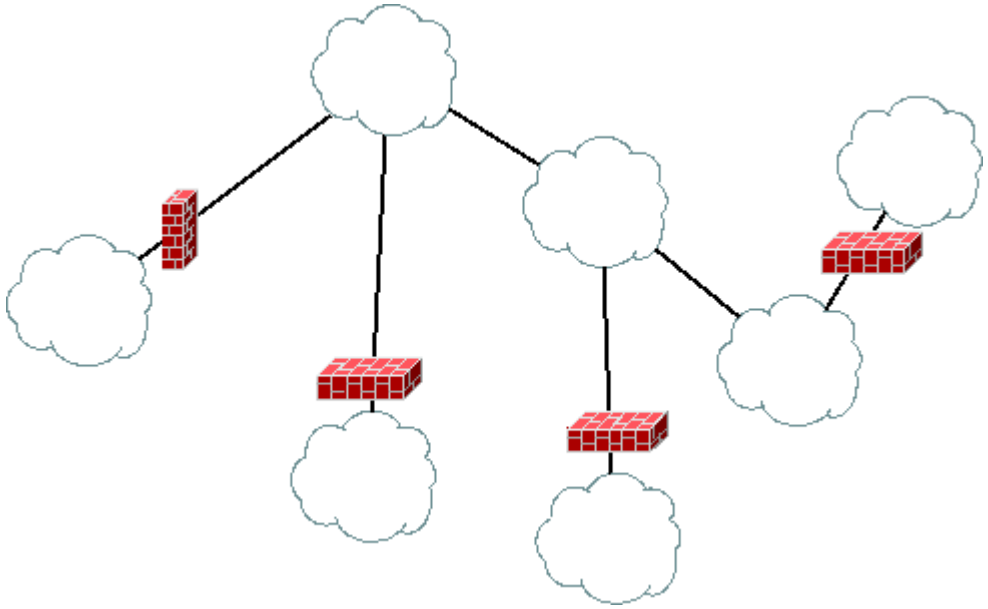
A Router

- Also known as gateway
- Has multiple network interfaces
- Has multiple IP addresses
- Connects different networks
- Has a routing table that instructs where packets go
  - The big routers on the internet have tables with 100,000s of entries (using BGP)
- Can also be a firewall/packet filter


The Internet
141.30.1.93
default
141.76.2.0/24
141.76.6.0/24
141.76.2.254/24
141.76.6.254/24
141.76.2.1/24
141.76.6.46/24

A Firewall

- Packet filters grant or deny network packets
- Application gateways grant or deny content



Sending Data

- TCP and UDP are the most common used protocols to send and receive data on the Internet
- TCP/UDP are based on IP
  - TCP: session based protocol, defines a stream between two nodes, preserves order, retransmission upon errors, checksums
  - UDP: session-less protocol, no error correction, no retransmission etc.
- Both use 'ports' to distinguish services offered by hosts

Services using TCP/UDP

Protocols built on top of TCP/UDP and their typical ports:

TCP:

- HTTP: 80
- HTTPS: 443
- SMTP: 25
- IMAP: 143
- IMAPS: 993
- SSH: 22

UDP:

- NTP: 123
- TFTP: 69

Where can Packets be Intercepted?

In a router (mostly).

But on which network level?
Possibilities:

- Data-link level (e.g. Ethernet)
- Network level (e.g. IP, ICMP)
- Transport level (e.g. TCP and UDP)
- above (e.g. HTTP)

Data-Link Level

- Example: Ethernet
- Packet type: Ethernet frames
- Device: Ethernet-Switch (or PC)
- Filter for:
  - Source and destination MAC addresses
  - Might also inspect packets and filter IP, ARP, VLAN and other protocols

Transparent firewall

Network Level

- Example: IP
- Packet type: IP packets
- Device: Router (or PC)
- Filter for:
  - Source and destination IP addresses

Transport Level

- Example: TCP/UDP
- Device: Router (or PC)
- Filter for:
  - Source and destination addresses and ports
  - TCP flags

That's what most packet filters implement.

Basic Rules

Allow SMTP from everywhere to mail server
Allow SSH from within faculty net to web server

TYPE	SRC IP	PORT	DEST IP	PORT	ACTION
tcp	*	*	141.76.2.1	25	allow
tcp	141.76.0.0/17	*	141.76.2.31	22	allow

The rules only cover one way, what about the reply?

Handling the Reply

The node initiating the connection will use a local (arbitrary) port number, e.g.:

- client:48230 → firewall → server:80

The server will reply by sending to port 48230 on the client

Consequence, as the local port numbers are undefined:

  - Firewall needs to have a rule like:

```
TYPE SRC IP        PORT    DEST IP     PORT   ACTION
tcp  server        80      *           >1023  allow
```

Which basically means the internal network is (half) open from outside.

Stateful Filtering

- TCP is a session based protocol
  - multiple TCP packets belong to a TCP stream
- A firewall can detect those!
- Opening connections from outside is not needed
  - this is essential
- The firewall can bypass filter rules if it identifies that a packet belongs to an already existing connection
  - faster
- Also possible for UDP, only open a specific (client:port ↔ server:port) connection
- Close connection with timeout

Special Protocols

Example FTP:

- FTP in active mode uses two connections, a control connection and a data connection, the data connection is initiated from the server(!)
  - Firewall needs to understand the FTP protocol to support active FTP
  - passive FTP available as well

Application Gateways

- 'Firewalls' on the upper levels
- Content filtering
- Analyze content (data) that is being sent via TCP/UDP connections
- Usually referred to as “Proxy”
- Examples:
  - Web-Proxy (HTTP/HTTPS)
  - Mail (SMTP)

Web-Proxy

- between the web-browser (client) and the web-server
- can be configured by clients
  - Your ISP might provide a web-proxy
- can be transparent, i.e. all clients must use the proxy
  - the case in most (bigger) companies
- Used for:
  - Caching (avoid redownloading content from the net)
  - Logging/filtering/modifying/blocking of browsed URLs and/or content

SMTP-Proxy

- Mail is usually filtered to protect against:
  - SPAM and
  - Viruses and other malicious software/content
- Methods:
  - Tag mail, client/user decides
  - Cut out unwanted content
  - Reject

Firewall Practices

- Concentrate on IP, TCP/UDP firewall
- Vulnerabilities?
  - One client
  - Network with router
- Network Address Translation (NAT)
- Methods:
  - White-list
  - Black-list

Single Client

- Block ports from outside connections if there are open services
  - BUT: don't open them $\rightarrow$ no need to block them
  - is good second line of defence
- You may want to give certain hosts/network access to specific services
  - using an IP filter is one way of doing so

Protecting a Network

- Types of Networks:
  - Internal hosts use routed IP addresses
    - Internal hosts can be connected from outside
  - Internal hosts use a private IP space (not routed)
    - Only the router has an official IP

Network with Routed IPs

- External hosts can connect to every single host inside the network
  - Firewall on the Router can prevent this
- Internal to external traffic can also be controlled

Network with Private IP Space

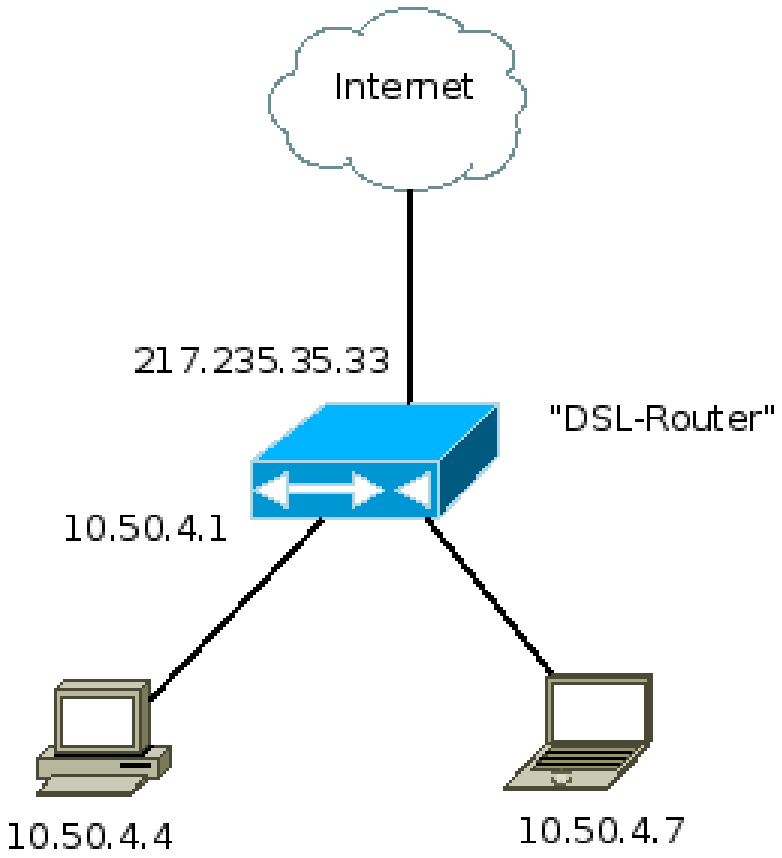
- Private IPs are not routed by public routers
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16
- Only the router has a public IP, internal hosts do not need to have public IPs
- Router must translate IP addresses from internal hosts to its own address and translate back responses from hosts on the net
  - → Network Address Translation — NAT

Network Address Translation (NAT)

- With NAT on the router
  - Internal hosts cannot be addressed from outside
  - Internal hosts are hidden to outside (can be tracked with some effort)
  - Security feature!
- Routers can usually be configured to pass a connection to a specific port through to some internal host

Typical NAT Setup

- Router translates IPs
- Internal host cannot be directly reached



IPv6 and NAT

- NAT was/is primarily used to save IP addresses
  - Typical home setup: People have multiple devices on their DSL line (PCs, notebooks, PDAs, fridges, ...)
- Ipv6 has plenty of them for everyone
- But is it good to let everyone connect to your internal hosts? We do have firewalls anyway?! What about privacy of internal hosts?
  - IPv6 has a privacy feature...

How to do the Rules?

To build a firewall, one needs to specify rules of connections that are allowed or dis-allowed to connect

Two main methods:

- Black list
- White list

Black list

- Default: Open everything, no blocking
- Block only hosts and/or ports that seem necessary

- Not recommended
  - easy to miss something
  - newly opened ports are open to outsiders (and forgotten)

White list

- Default: close everything from outside
- Only open services that are necessary

- Recommended
  - Internal are free to do what they want
  - Internal hosts cannot offer new services without having a hole in the firewall

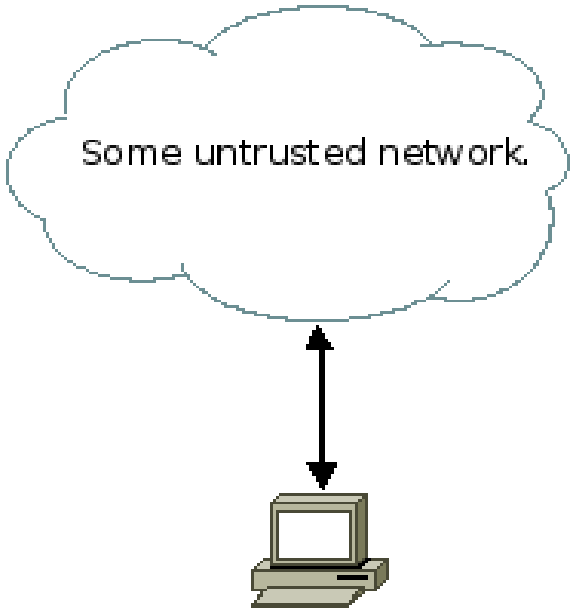
Firewalling Internal Traffic

- Most (bigger) companies
  - Redirecting HTTP(S) to web-proxies
  - close everything else
  - Getting an SSH to outside is often hard
- closing the SMTP port for certain machines might be good to prevent viruses etc. to spread more
- otherwise usually no restrictions

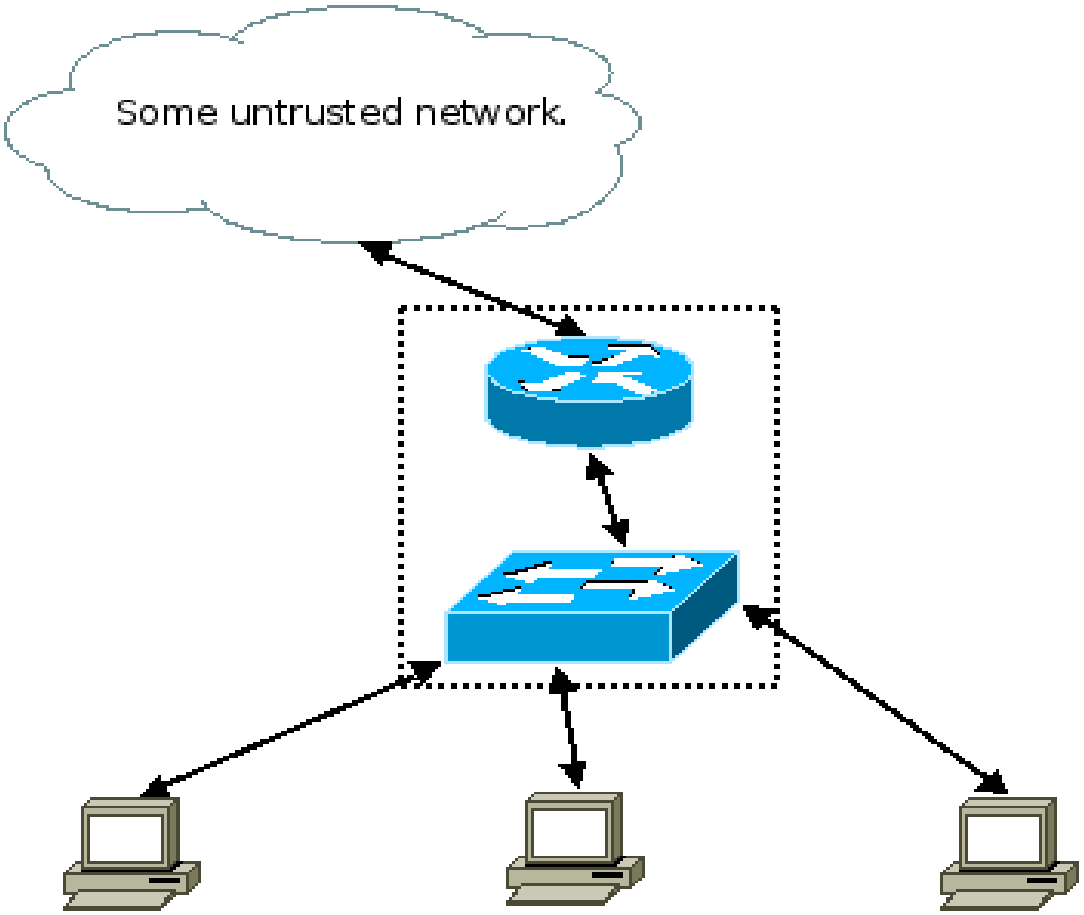
Firewall Setups

There are several possible setups a firewall can be part of.

Single host:



Simple Network



Typical home setup:

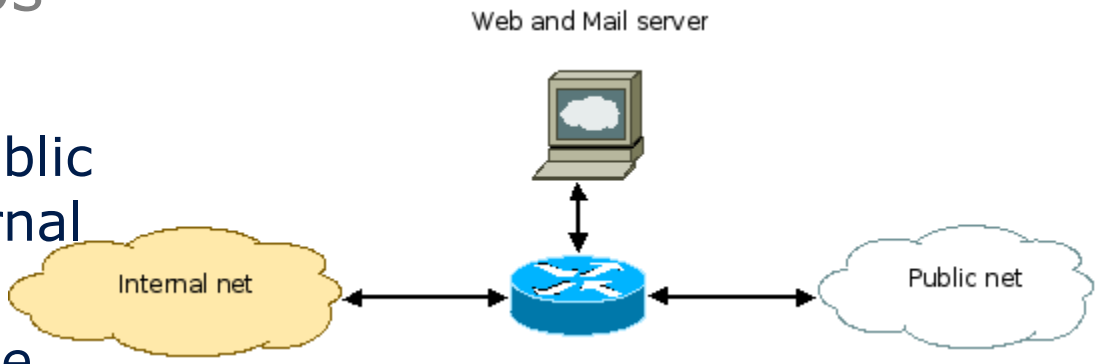
- Router+Switch connects multiple hosts with the Internet

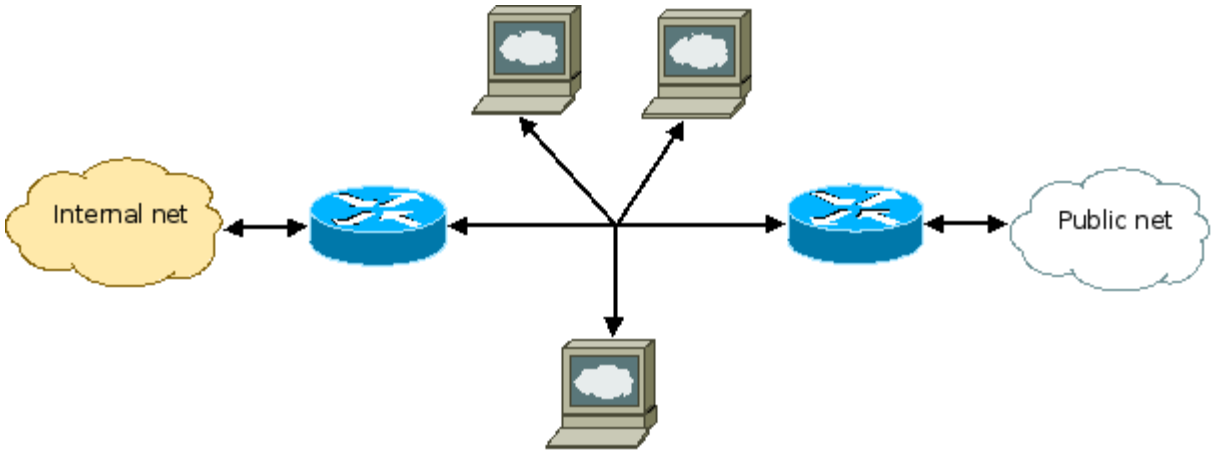
Multi-Level Setups

Separate and protect public services and the internal network.

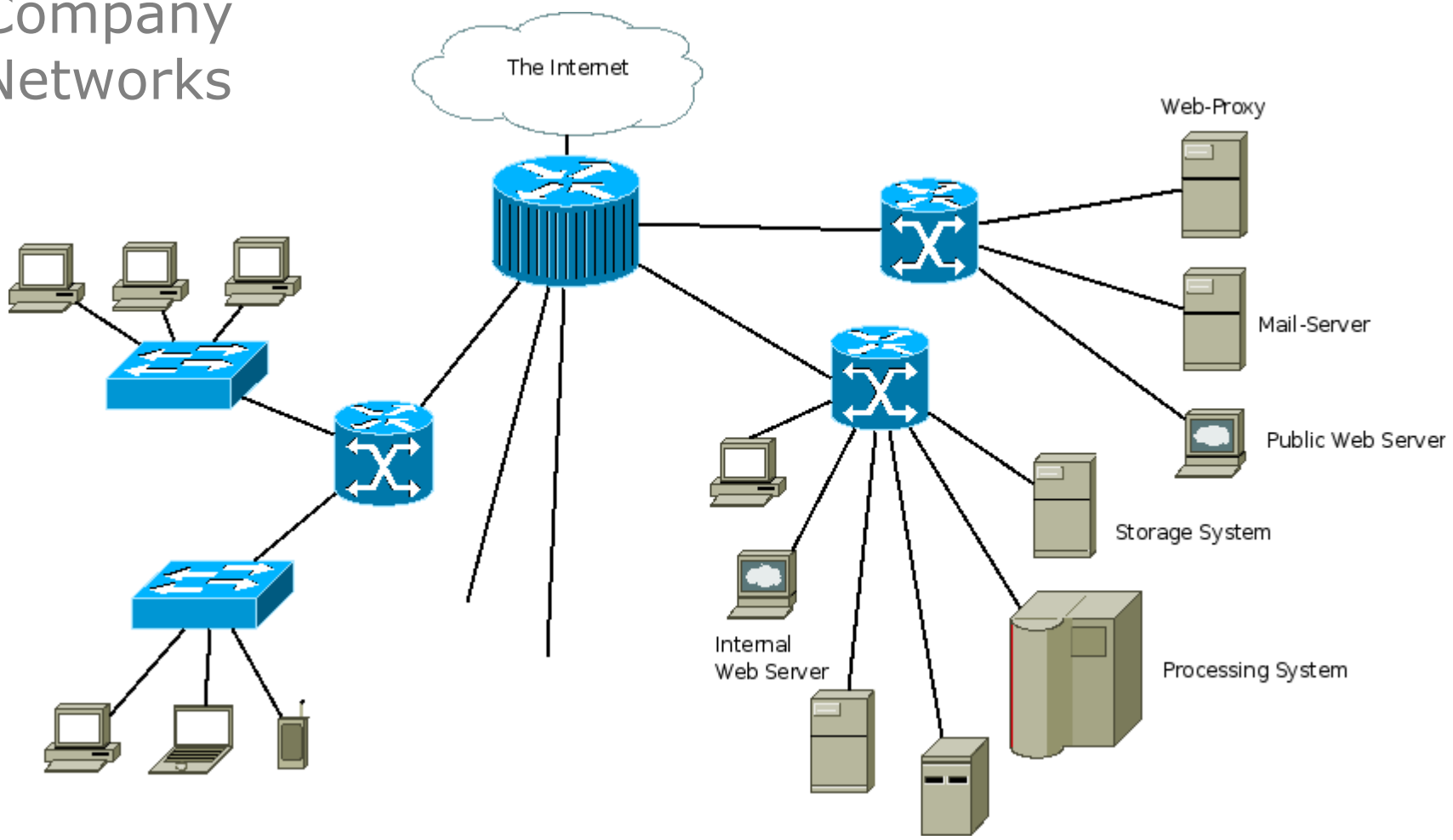
DMZ – demilitarized zone

Perimeter network





Company Networks



Further Techniques

- Intrusion Detection Systems (IDS) and Prevention Systems (IPS)
- Port Knocking

Intrustion Detection and Prevention

- Detect 'malicious' behaviour:
  - port scanning
  - DoS attacks
  - matching signatures
  - 'unusual' behaviour
- IDS: Alert
- IPS: Do something:
  - Block connections
  - Rate-Limit connections
  - ...

Port Knocking

- Selectively open ports for a specific host
- Client 'knocks' on the firewall by connecting to closed ports with a predefined sequence
- Firewall detects this sequence and opens port(s) for the client
- Mostly used for SSH access
- May use crypto techniques in sequences
- Hides offered services
- Voids brute force account cracking attacks

Example: Firewall with Linux iptables

Scenario:

- Small network with one router including a firewall configuration
- The network has a web-Server and a mail server offering SMTP, SSMTP, IMAPS and POP3S
- SSH to the router and mail server is allowed
- All other ports are closed
- Internal clients have no restrictions
- Log every denied packet

iptables

- In Linux, the packet filter sub-system is called 'netfilter'
- The basic configuration tools are 'iptables' and ip6tables' for IPv4 and IPv6 configuration
- iptables defines chains which consist of rules, rules contain other chains that are taken when a rule matches
- Main special chains:
  - INPUT, FORWARD, OUTPUT and ACCEPT, DROP, REJECT

iptables chains

- INPUT: incoming traffic for the local host
- OUTPUT: traffic generated by the local host
- FORWARD: forwarded traffic

- ACCEPT: accept the packet
- DROP: just drop
- REJECT: reject the packet by sending an ICMP message to the sender

Network Interfaces

The router has two physical network interfaces:

- ethi — connected with the internal network
- ethe — connected with the external/public network

IP addresses

To configure the firewall, we need to know the IP network and names/IP addresses of the host offering services:

- Network: 141.76.20.0/24
- Webserver: 141.76.20.9
- Mail-Server: 141.76.20.11

Example

```
# Initialisation
# flush all rules and chains
iptables -F
iptables -X
# select default policies
iptables -P INPUT       DROP
iptables -P FORWARD DROP
iptables -P OUTPUT    DROP
```

Example

```
# allow connection we already know
iptables -A INPUT -j ACCEPT -m state \
   --state ESTABLISHED,RELATED
iptables -A FORWARD -j ACCEPT -m state \
   --state ESTABLISHED,RELATED
```

Example – SSH

```
# allow SSH to the router from everywhere
iptables -A INPUT -j ACCEPT -p tcp --dport 22

# allow SSH to the mail server
iptables -A FORWARD -j ACCEPT -p tcp \
  -d 141.76.20.11 --dport 22
```

Sanity – localhost and block invalid IPs

```
# allow localhost communication
iptables -A INPUT    -j ACCEPT -i lo
iptables -A OUTPUT -j ACCEPT -o lo

# skip private nets (can also be done with routing)
for n in 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16; do
  iptables -A INPUT       -s $n -j DROP
  iptables -A FORWARD -s $n -j DROP
  iptables -A OUTPUT    -s $n -j DROP
done
```

Internal Servers

```
# Mail Server
iptables -A FORWARD -j ACCEPT -p tcp \
  -d 141.76.20.11 -m multiport \
  --dport smtp,ssmtp,imaps,pop3s

# Web server
iptables -A FORWARD -j ACCEPT -p tcp \
  -d 141.76.20.9 -m multiport --dport http,https
```

Example

```
# allow every packet from internal interface to proceed
iptables -A FORWARD -j ACCEPT -i ethi -o ethe

# log everything what's left
iptables -A INPUT -j LOG
iptables -FORWARD -j LOG
```