



Firewalls

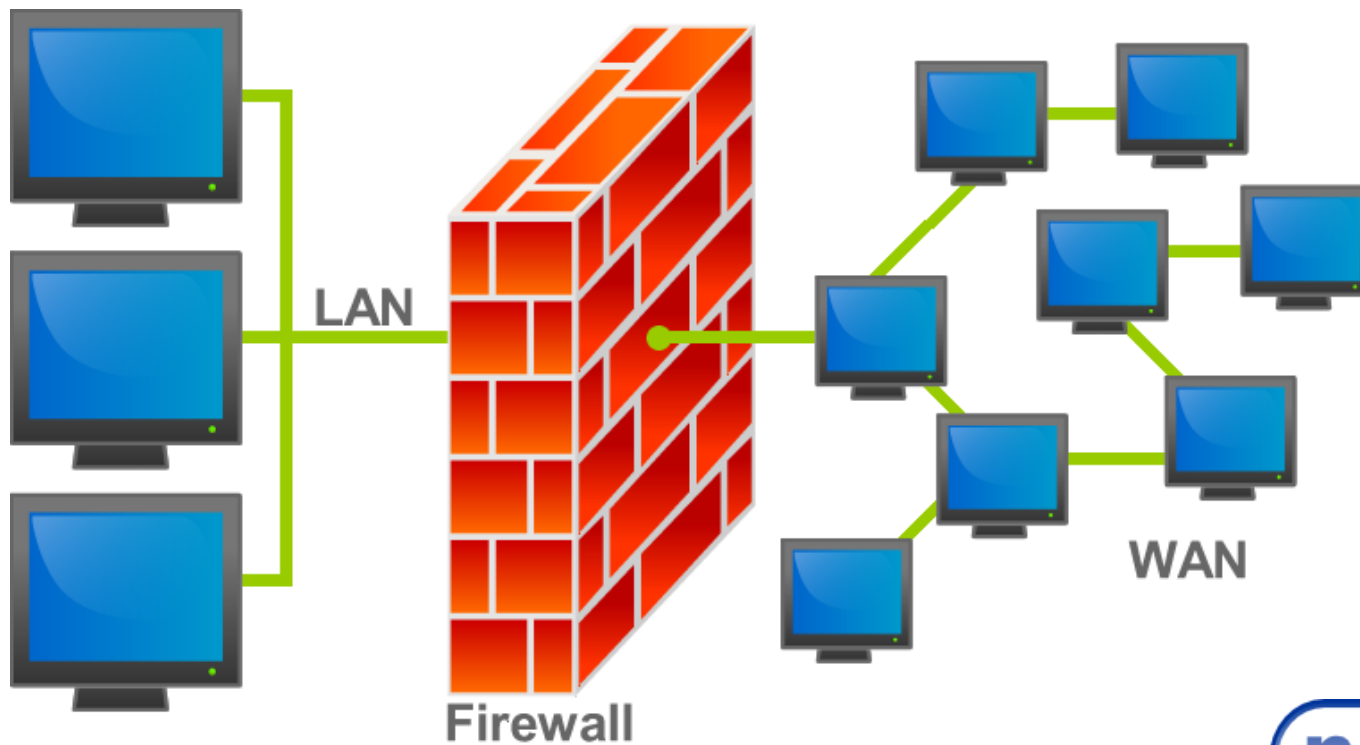
Tamás Holczer

Laboratory of Cryptography and System Security

Department of Networked Systems and Services

Holczer@CrySyS.hu

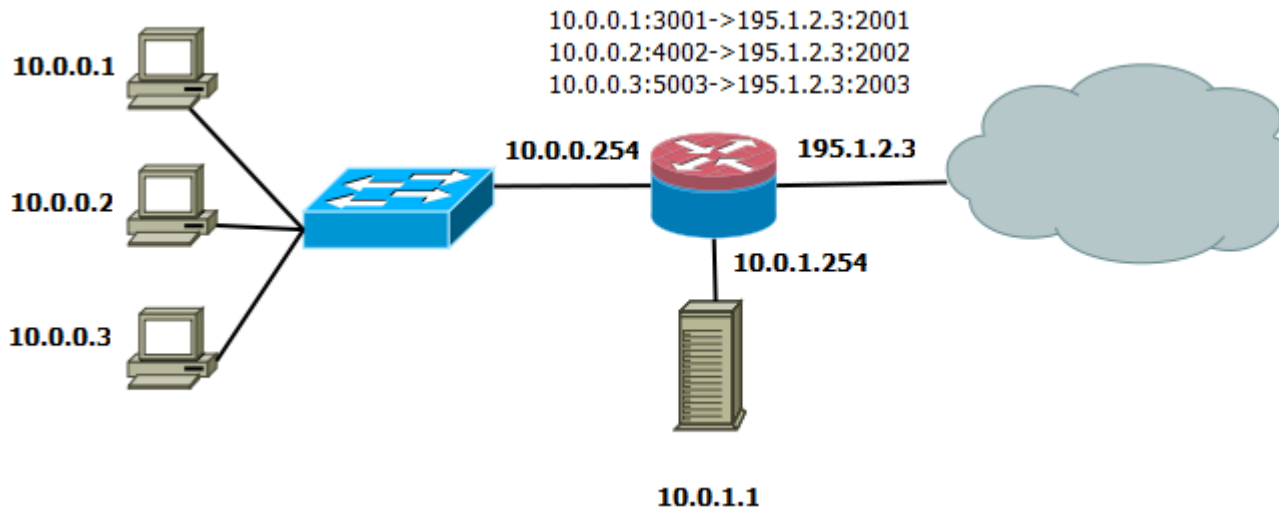
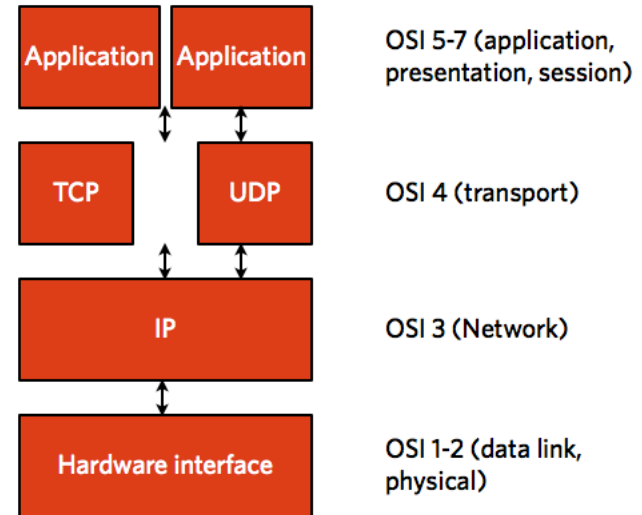
Introduction



Background

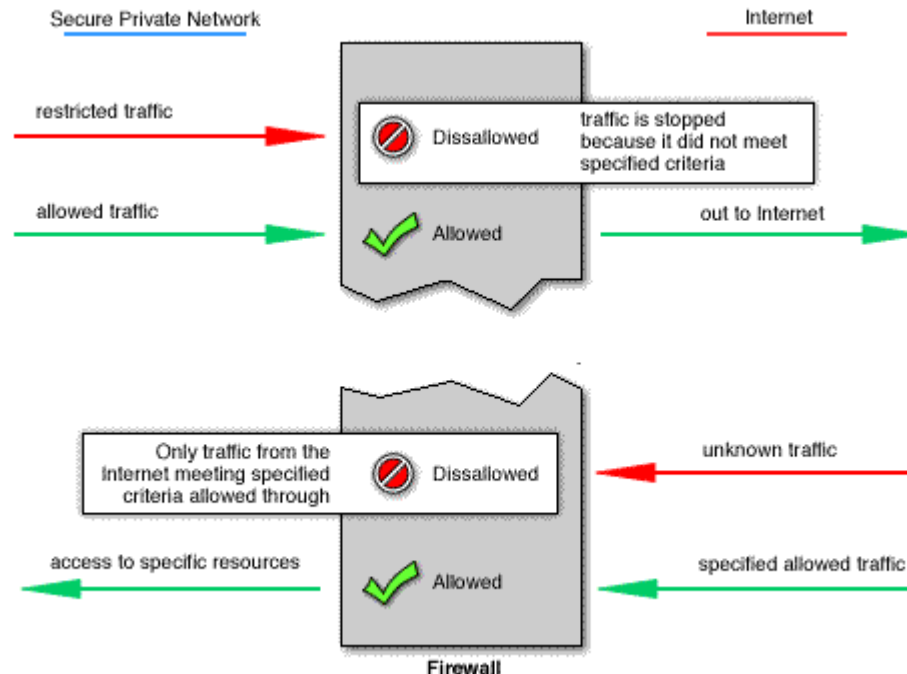
- TCP/IP Stack
- Private/Public Addresses
 - 10.X.X.X
 - 172.16-31.X.X
 - 192.168.X.X
- NAT

TCP/IP stack



What is a firewall?

- a firewall is a system or group of systems that enforces some network access control policy
 - usually, the policy is defined by filtering rules
 - each incoming and outgoing packet is matched against the filtering rules and it is either allowed to pass or dropped



High level design goals for firewalls

- all traffic from inside to outside, and vice versa, must pass through the firewall
 - ensured by the appropriate design of the topology of the network and the placement of the firewall
 - various topologies and placements are possible
- only authorized traffic, as defined by the access control policy, must be allowed to pass (allow list instead of block list)
 - ensured by the appropriate filtering rule set that enforces the given policy
 - filtering at various layers are possible
- the firewall itself must be immune to penetration
 - secure OS, limited set of allowed features, systematic maintenance

Why it is hard to design a good firewall?

- The topology is
 - constantly changing
 - not (well) documented (who is the backend server? The one used to be act like a backend server)
- No one cares (or dares) to delete an existing rule
- Do not touch if it is working!? (lack of ticket is good?)
- Lack of precise specification
 - „Make a secure firewall” is not a specification
 - „Let only in the required traffic” is not a specification
 - The best you can get is a list of high level goals
- Takes some time to realize problems (limit based rules)
- At large corporates:
 - Distributed topology (several firewalls)
 - Distributed tasks (tier 1-2... specialists)

Classification 1

- Packet filtering firewall - Typically is a router with the capability to filter some packet content, such as Layer 3 and sometimes Layer 4 information
- Stateful firewall - Monitors the state of connections, whether the connection is in an initiation, data transfer, or termination state.
- Application gateway firewall (proxy firewall) - A firewall that filters information at Layers 3, 4, 5, and 7 of the OSI reference model. Most of the firewall control and filtering is done in software.
- Network address translation (NAT) firewall - A firewall that expands the number of IP addresses available and hides network addressing design (it is rather a functionality in FWs).

Classification 2

Type	Advantage	Disadvantage
Packet filtering	<ul style="list-style-type: none">• Simple• Efficient• Hardware support feasible	<ul style="list-style-type: none">• Stateless• Fragmented packets• Dynamic ports
Stateful filtering	<ul style="list-style-type: none">• Simple• Efficient• Easier to realize policy• Better against DoS	<ul style="list-style-type: none">• Not good against Application layer attacks• Requires more resources
Application level	<ul style="list-style-type: none">• Sophisticated rules• Can realize complex policies	<ul style="list-style-type: none">• Requires lot of resources• Slows down traffic

- no perfect firewall exists
- choice depends on the policies to be realized
- in most cases a mix is used

Classification 3

- Host-based (server and personal) firewall - A PC or server with firewall software running on it.
- Transparent firewall - A firewall that filters IP traffic between a pair of bridged interfaces.

- PC based firewall (i.e.: Linux with iptables)
 - Low throughput
 - Highly configurable
- Router based firewall (i.e.: Cisco 1800 series)
 - Medium throughput
- Hardware based firewall (i.e.: Cisco ASA, FortiGate, Sophos UTM)
 - High throughput
 - Medium/Highly configurable
 - Linux/Custom OS inside

Packet filtering / Access Control Lists 1.

- Basic (standard) format:

```
protocol source-addr [source-wildcard] destination-addr  
[destination-wildcard] {permit | deny}
```

- protocol: ip/tcp/udp(/icmp)
- source-addr, source-wildcard: range of senders
- Destination-addr, destination-wildcard: range of receivers
- permit | deny: decision

- Advanced format:

```
protocol source-addr [source-wildcard] sport destination-addr  
[destination-wildcard] dport {permit | deny} [log]
```

- Port numbers
- logging

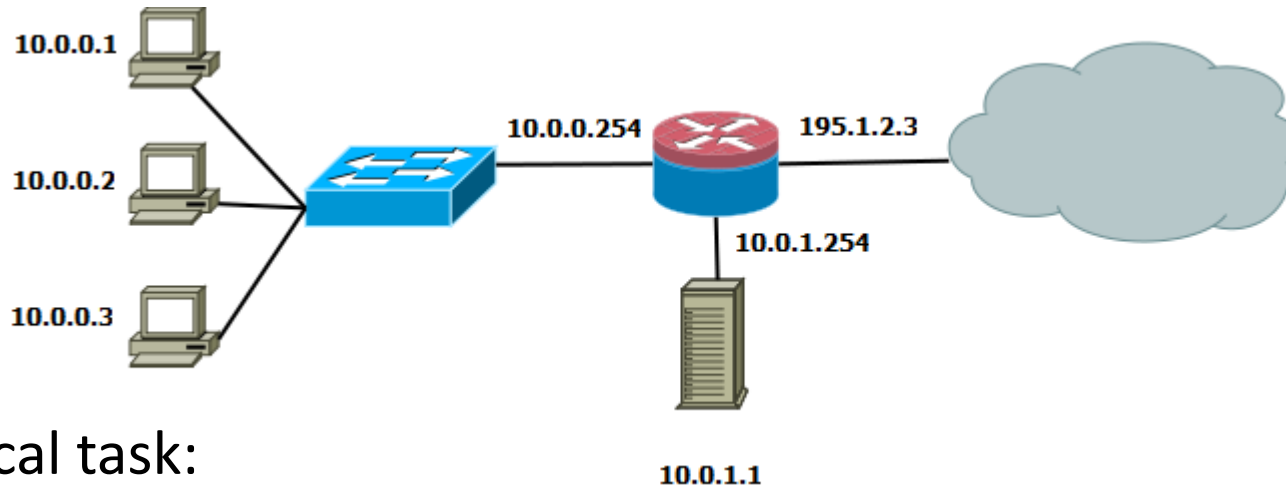
Packet filtering / Access Control Lists 2

- destination IP address (subnet)
- destination port (range)
- source IP address (subnet)
- source port (range)
- protocol (TCP, UDP, ...)
- TCP flags
 - SYN – used in the connection setup phase
 - ACK – indicates that the message contains acknowledgment information
 - FIN – indicates that the sender wants to close the connection
 - RST – used to signal that a transmission failure occurred (no ack arrived for some segment) → state of connection is reset
 - ...
- ICMP “type” and “code” fields
- interface ID and direction

Packet filtering / Access Control Lists 3

- Design considerations of ACLs
 - Processed top-down
 - » First match determines outcome (or last or best match)
 - » When inserting new statement, it can be inserted as first or last in most cases
 - Implicit deny at the end of list (sometimes default accept)
 - Consider where to put
 - » Close to source → efficient bandwidth management
 - » Close to destination → can be more specific
 - » Decide: Firewall (Interface, Direction)
 - Inconsistency, inefficiency
 - » Shadowing (more general before specific)
 - » Generalization (more specific before general)
 - » Correlation (partly overlapping)
 - » Redundancy (total overlapping)

Packet filtering / Access Control Lists 4



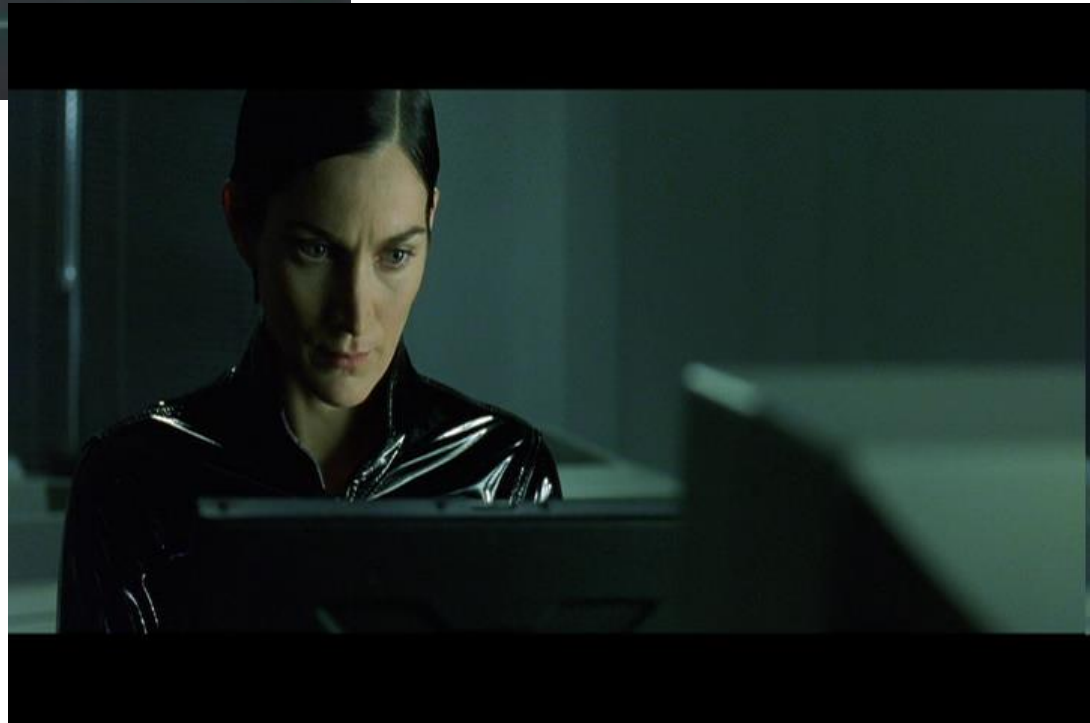
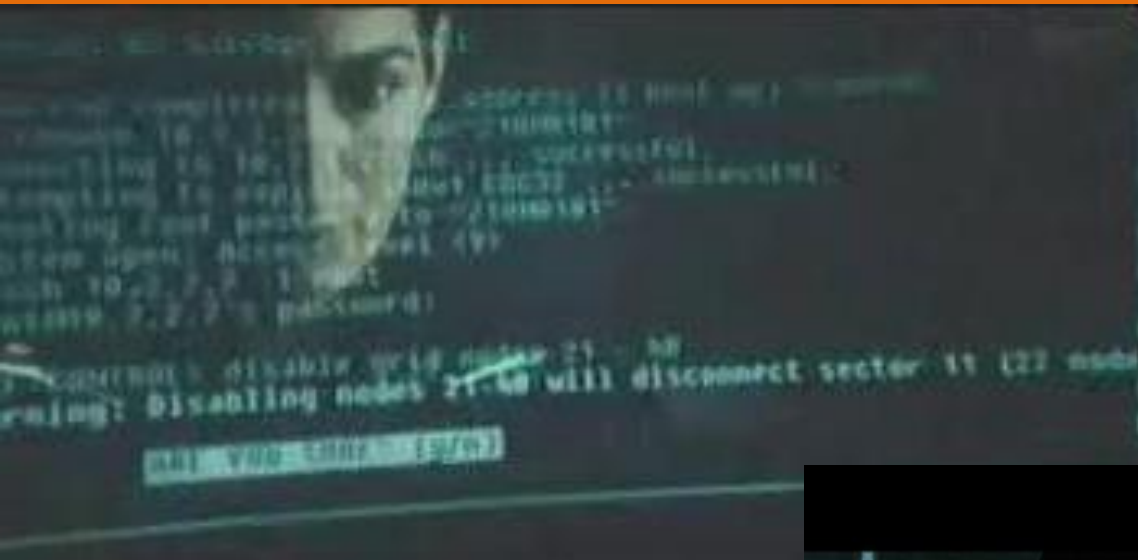
- Typical task:
 - Allow HTTP access to web server from anywhere
 - Allow HTTP access from inside to Internet
 - Allow FW access from inside
 - Allow DNS, email, ntp...
 - Discard everything else

Packet filtering: Port scanning

- Initiate a TCP connection
- No answer? ICMP unreachable message– Maybe our packet was filtered out (“discarded”): maybe a firewall protects the host
- The answer is an RST packet? – Most likely a closed port
- The answer is a SYN packet? – Open port
- Details on port scanning:

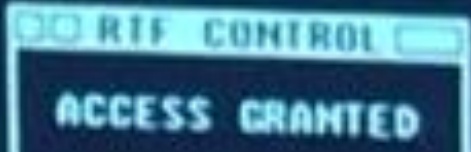
<http://nmap.org/book/man-port-scanning-techniques.html>

Nmap port scanner / as seen in the movies



Screenshot from the Matrix

```
80/tcp      open       http
81/tcp      open       hosts2-ns
10.0.0.0 [nobile]
11 # nmap -v -ss -O 10.2.2.2
11
13 Starting nmap V. 2.54BETA25
13 Insufficient responses for TCP sequencing (3). OS detection i
13 accurate
14 Interesting ports on 10.2.2.2:
44 (The 1539 ports scanned but not shown below are in state: cl
51 Port      State      Service
51 22/tcp     open      ssh
58
68 No exact OS matches for host
68
24 Nmap run completed -- 1 IP address (1 host up) scanned
50 # sshnuke 10.2.2.2 -rootpw-"210M0101"
Connecting to 10.2.2.2:ssh ... successful.
Re Attempting to exploit SSHv1 CRC32 ... successful.
IP Resetting root password to "210M0101".
System open: Access Level <9>
Hn # ssh 10.2.2.2 -l root
root@10.2.2.2's password: █
```



Firewall example: Prevention of port scanning

- different ways to scan open ports
 - send a SYN packet to a TCP port, if port is used, you receive a SYN/ACK response, otherwise you receive an RST/ACK response
 - send a FIN packet to a TCP port, if port is not used, you receive an RST/ACK response , otherwise a FIN/ACK may be sent or the FIN packet is silently ignore
 - ...
- countermeasure:
 - filter incoming TCP connection requests
 - (but allow TCP connections to the web server and TCP connections initiated from inside)

src = any, sport = any, dst = <web server IP>, dport = 80, prot = tcp → ALLOW

src = any, sport = any, dst = <internal IP range>, dport = any, prot = tcp, SYN = 1, ACK = 0 → DROP

src = <internal IP range>, sport = any, dst = any, dport = any, prot = tcp, RST = 1, ACK = 1 → DROP

src = <internal IP range>, sport = any, dst = any, dport = any, prot = tcp → ALLOW

src = any, sport = any, dst = <internal IP range>, dport = any, prot = tcp, ACK = 1 → ALLOW

Example: Detection of IP address spoofing

- source IP address in IP packets is not authenticated, spoofing is very easy
- countermeasures:

- ingress filtering can drop all packets where source IP address is an internal address

src = <internal IP range>, dst = any, prot = any, dir = inbound → DROP

- egress filtering can drop all packets where destination IP address is an internal address

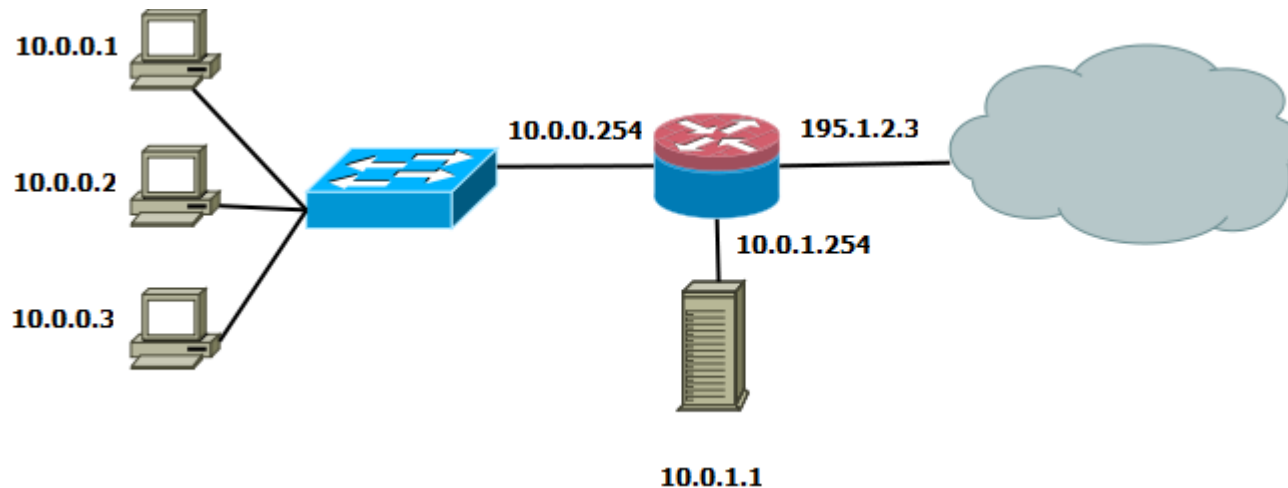
src = any, dst = <internal IP range>, prot = any, dir = outbound → DROP

src != <internal IP range>, dst = any, prot = any, dir = outbound → DROP

(some internal host may be compromised and used to spoof IP addresses)

- Reverse Path Filtering
- Bogon Filtering

Packet filtering exercise



■ Task:

- Allow HTTP access to web server from anywhere
- Allow HTTP access from inside to Internet
- Discard everything else

Problems:

No answer to outside

Attacker from port 80 can attack

Can be partly mitigated by TCP flags

■ Solution:

- src=any, sport=any, dst=10.0.1.1, dport=80 → ALLOW
- src=10.0.0.0/24, sport=any, dst=any, dport=80 → ALLOW
- src=any, sport=80, dst=10.0.0.0/24, dport=any → ALLOW
- src=any, sport=any, dst=any, dport=any → DROP

Dynamic/stateful packet filters

- motivation
 - if we want to allow internal hosts to use external services, we should make sure that the responses from those external servers are let in
 - in case of a stateless packet-filter, we allow more than that
src = <internal IP range>, sport = any, dst = any, dport = any, prot = tcp → ALLOW
src = any, sport = any, dst = <internal IP range>, dport = any, prot = tcp, ACK = 1 → ALLOW
- dynamic packet filters keep track of the existing connections, and an incoming packet is let in only if it can be associated with an already existing connection (based on the addresses and port numbers)
 - connection table is checked first
 - then comes the matching with the static rules
 - new connection is inserted in the connection table after a successful 3-way TCP handshake (“established” state, reflexive ACL)
 - and removed after the FIN exchange

Why are dynamic packet filters better?

- Static firewalls can achieve almost all of the tricks of the dynamic (stateful) packet filters
- However, It is important to have information about individual connections and their state
- With the additional information some spoofing attacks might be less probable
- Less push on the sysadmins on the rules: Easier to configure
- Makes it possible to handle with special protocols (e.g. FTP has a control channel, then it opens a data TCP connection for file transfer / file listing -> easy job in stateful filters)

Stateful example: pfSense

pfSense+ System Interfaces Firewall Services VPN Status Diagnostics Help

Status / Dashboard

System Information

Name: pfSense.centralus.cloudapp.azure.com

User: admin@87.115.241.173 (Local Database)

System: Microsoft Azure
Netgate Device ID: 8ad1a7682b9e719428f6

BIOS: Vendor: American Megatrends Inc.
Version: 090008
Release Date: Fri Dec 7 2018

Version: 21.02-RELEASE (amd64)
built on Tue Feb 16 08:56:32 EST 2021
FreeBSD 12.2-STABLE

The system is on the latest version.
Version information updated at Wed Feb 24 20:10:28 UTC 2021

CPU Type: Intel(R) Xeon(R) CPU E5-2673 v4 @ 2.30GHz
AES-NI CPU Crypto: Yes (inactive)

Kernel PTI: Enabled

MDS Mitigation: Inactive

Uptime: 00 Hour 20 Minutes 06 Seconds

Current date/time: Wed Feb 24 20:29:13 UTC 2021

DNS server(s):

- 127.0.0.1
- 168.63.129.16

Last config change: Wed Feb 24 20:26:31 UTC 2021

State table size: 0% (224/45000) [Show states](#)

MBUF Usage: 3% (760/26584)

Load average: 0.57, 0.66, 0.57

CPU usage: 5%

Memory usage: 75% of 451 MiB

Disk usage:

- /: 18% of 7.7GiB - ufs
- /var/run: 3% of 3.4MiB - ufs in RAM
- /mnt/resource: 0% of 3.9GiB - ufs

Interfaces

Interface	Speed	MAC
WAN	10Gbase-T <full-duplex>	10.1.7.9
WG0	25Gbase-ACC <full-duplex>	172.56.89.89

Gateways

Name	RTT	RTTsd	Loss	Status
WAN_DHCP	12.9ms	1.6ms	0.0%	Online
WG0_WGV4	0.1ms	0.2ms	0.0%	Online

Traffic Graphs

WAN

Interface Statistics

	WAN	WG0
Packets In	560352	0
Packets Out	567591	9
Bytes In	763.06 MiB	0 B
Bytes Out	38.97 MiB	776 B
Errors In	0	6
Errors Out	0	6
Collisions	0	6

Installed Packages

Name	Version	Actions
acme	✓ 0.6.9_3	Refresh Install Uninstall
aws-wizard	✓ 0.9	Refresh Install Uninstall
Cron	✓ 0.3.7_5	Refresh Install Uninstall
iperf	✓ 3.0.2_5	Refresh Install Uninstall
ipsec-profile-wizard	✓ 1.0_2	Refresh Install Uninstall
nmap	✓ 1.4.4_2	Refresh Install Uninstall

Packages may be added/managed here: [System -> Packages](#)

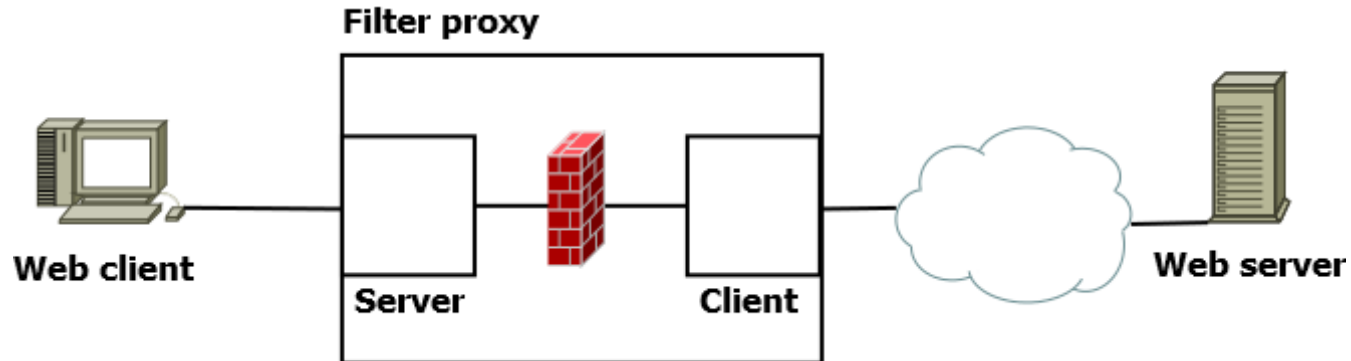
Services Status

Service	Description	Action
✓ dpinger	Gateway Monitoring Daemon	Refresh Install Uninstall
✗ iperf	iperf Network Performance Testing Daemon/Client	Refresh Install Uninstall
✓ pcscd	PC/SC Smart Card Daemon	Refresh Install Uninstall
✓ sshd	Secure Shell Daemon	Refresh Install Uninstall
✓ syslogd	System Logger Daemon	Refresh Install Uninstall
✓ unbound	DNS Resolver	Refresh Install Uninstall

Firewall Logs

Act	Time	IF	Source	Destination
✗	Feb 24 20:24	WAN	40.87.168.8	10.1.7.9:25524
✗	Feb 24 20:24	WAN	52.136.0.0	10.1.7.9:28235
✗	Feb 24 20:26	WAN	52.136.0.0	10.1.7.9:28241
✗	Feb 24 20:26	WAN	52.136.0.0	10.1.7.9:28243
✗	Feb 24 20:27	WAN	40.87.168.8	10.1.7.9:25538

Application Level Filtering / Proxies



- Hides internal users from the external network by hiding them behind the IP of the proxy (similar to NAT)
- Prevents low level network protocols from going through the firewall eliminating some of the problems with NAT
- Restricts traffic to only the application level protocols being proxied
- Proxy is a combination of a client and a server; internal users send requests to the server portion of the proxy which then sends the internal users requests out through its client (keeps track of which users requested what, returns data back to appropriate user)
- Address seen by the external network is the address of the proxy

Application Level Filtering / Proxies

Problems:

- SSL MitM (install root CA)
- HPKP (delete header and use vpn OR no alert in Chrome or Firefox if manual root CA is installed by proxy)

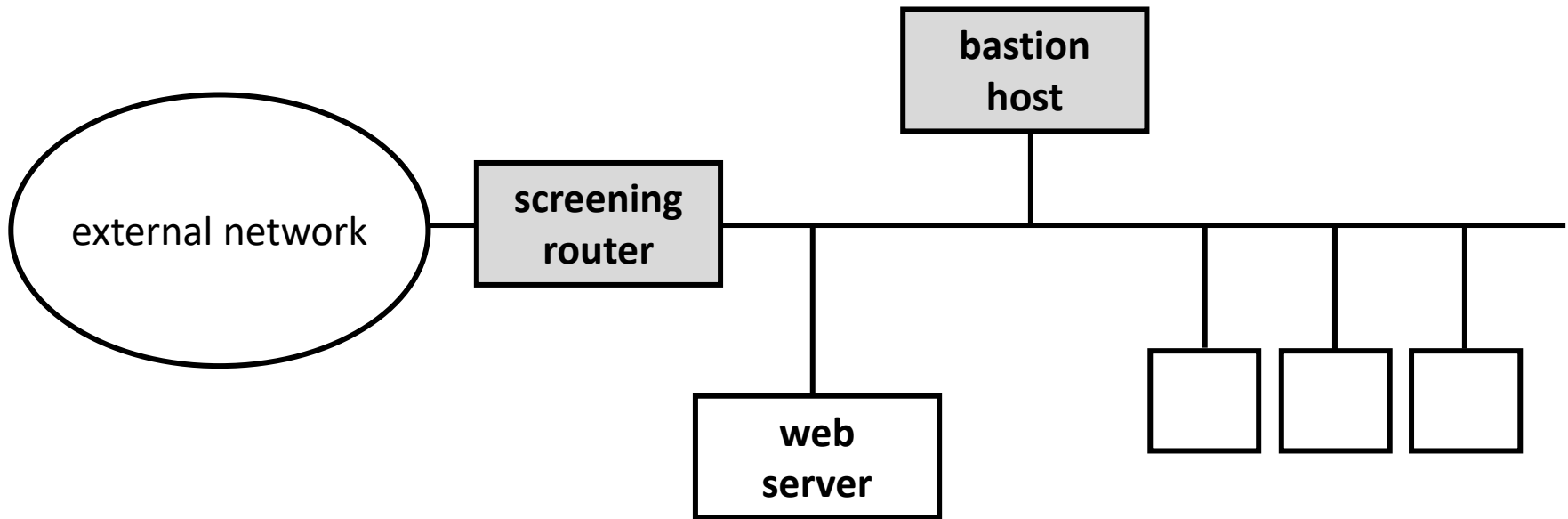
Companies like to filter content:

- Since the proxy server is a natural bottle neck for observing all of the external requests being made from the internal network it is the natural place to check content
- This is usually done by subscription to a vendor that specializes in categorizing websites into content types based on observation
- Usually an agent is installed into the proxy server that compares URL requests to a database of URLs to reject
- All access are then logged and reported, most companies then review the reported access violations and usually a committee reviews and decides whether or not any personnel action should be taken (letter of reprimand, dismissal, ect)
- Sites can be selectively filtered (e.g.:Facebook allowed only for HR)

Place of the firewall

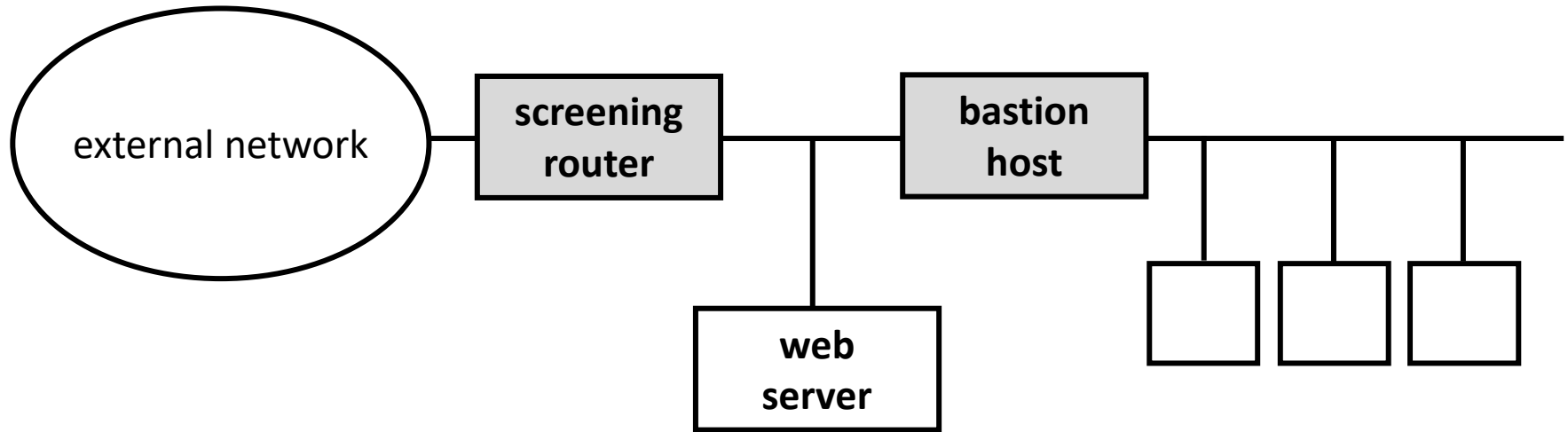
- a firewall architecture uses combinations of different devices to create an “architecture” for firewall defense
 - usually used in larger networks
 - multiple firewalls
 - different strengths to complement each other
 - architecture can be configured in different ways
 - These topologies are the basics, a common language, a philosophy, not too much more.
-
- screened host firewalls
 - dual-homed screened host
 - demilitarized zone (DMZ)

Screened host firewall



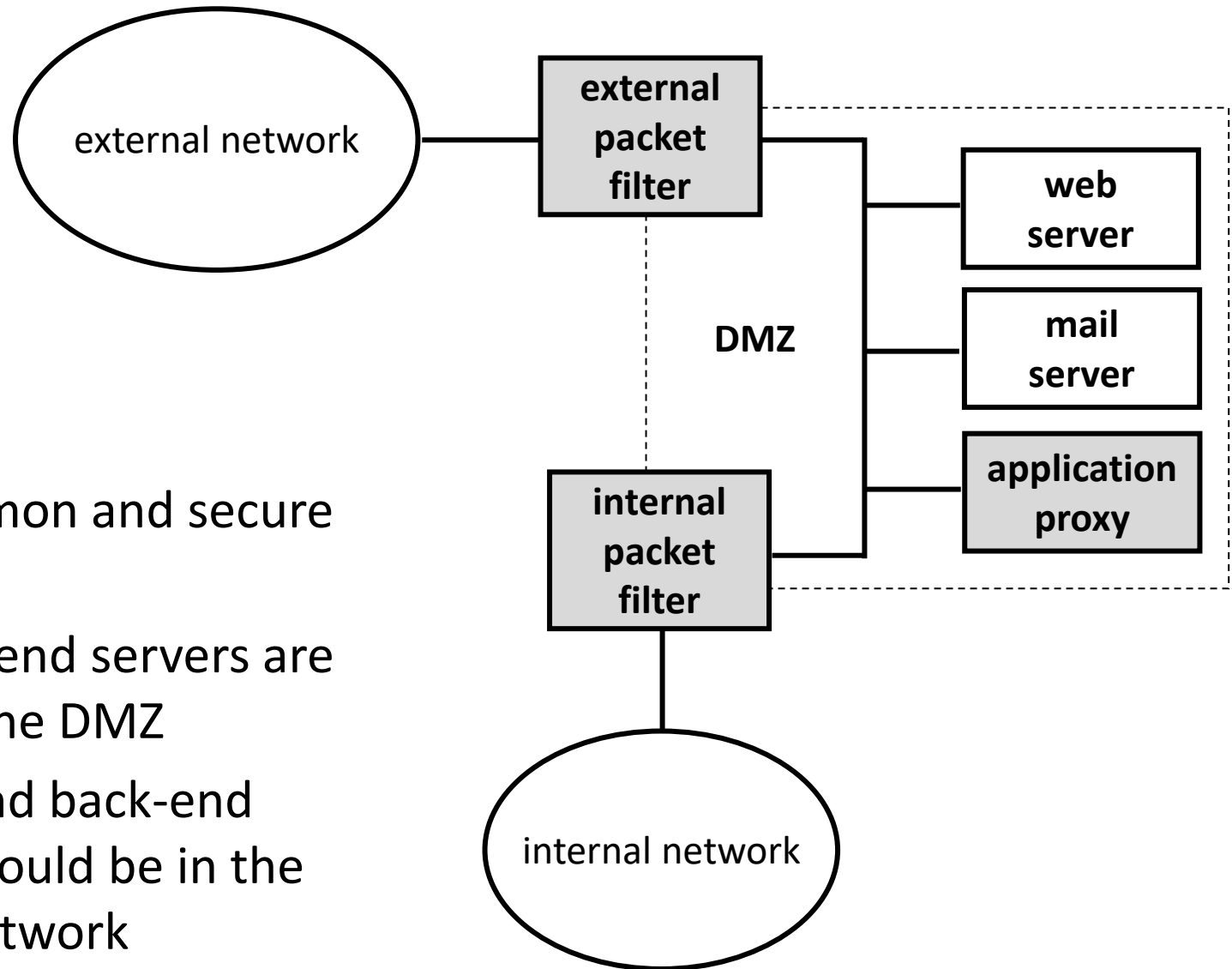
- screening router acts as a packet filter
- bastion host acts as an application proxy (or circuit level gateway)
- packet filter ensures that all incoming/outgoing traffic of the monitored applications (e.g., HTTP) is sent to/from the bastion host
- a potential weakness is the lack of physical separation between the internal hosts and the packet filter
 - e.g., corrupted internal host can spoof IP address of the bastion host

Dual-homed screened host



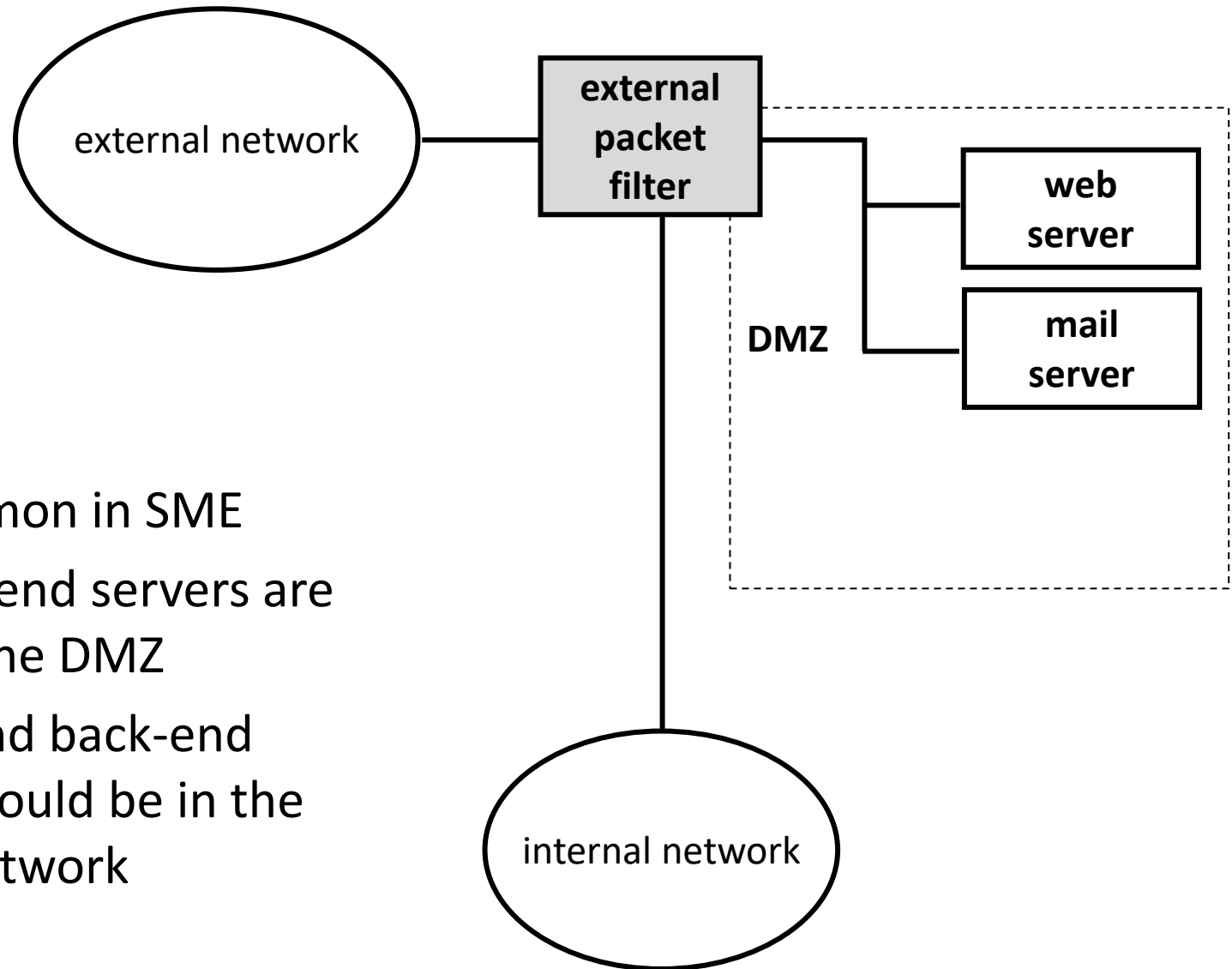
- this architecture ensures complete physical separation

DMZ architecture / DeMilitarized Zone



- most common and secure topology
- only front-end servers are placed in the DMZ
- all hosts and back-end systems should be in the internal network

Simplified DMZ architecture



- most common in SME
- only front-end servers are placed in the DMZ
- all hosts and back-end systems should be in the internal network

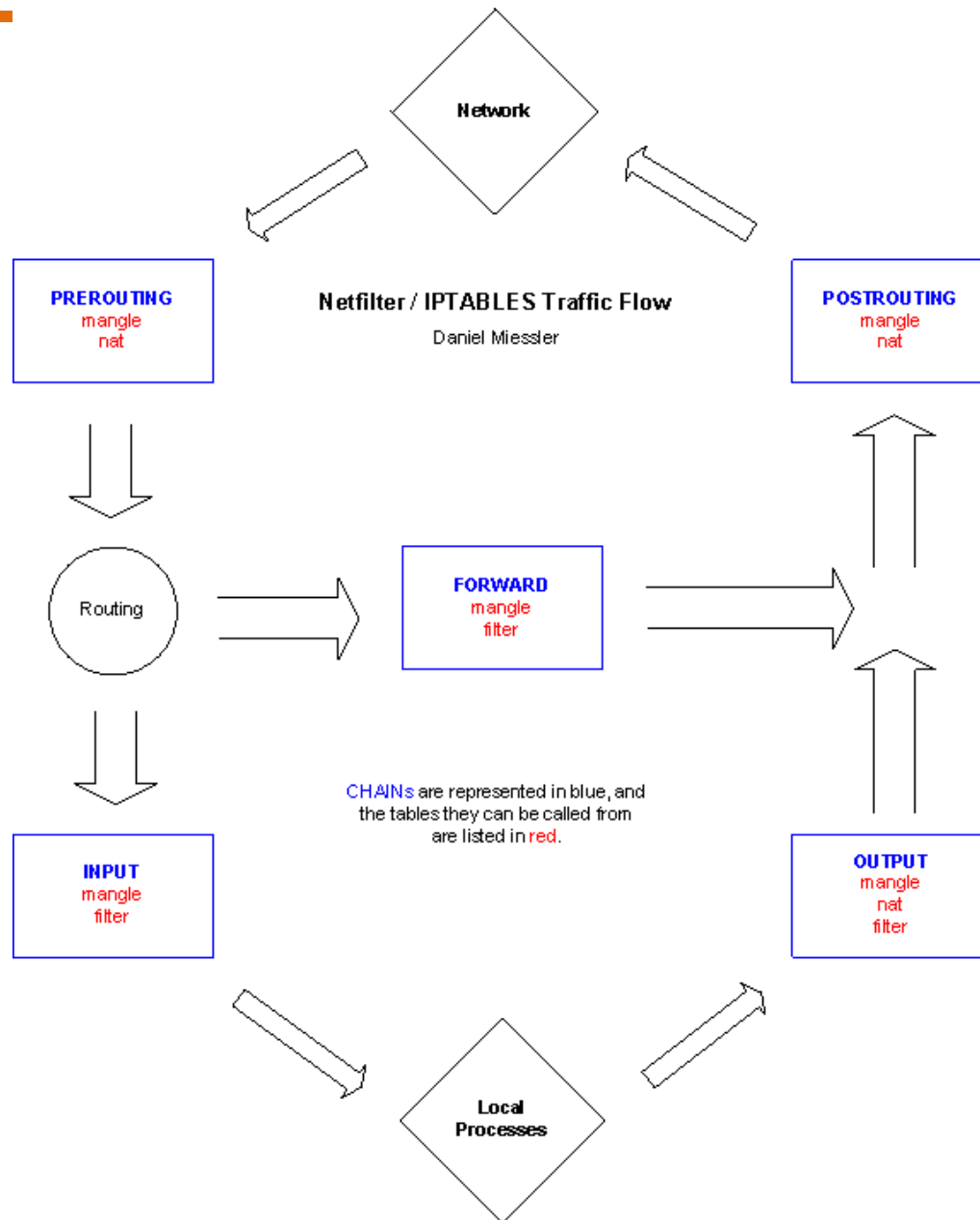
Limitations of firewalls

- firewalls may have software vulnerabilities, they may be hacked and penetrated
- even if correct, firewalls do not solve all security problems → one may have a false sense of security
- firewalls cannot protect against attacks that don't go through the firewall
 - data flow through CD, USB key [-> See Stuxnet]
- firewalls alone are not very efficient against new viruses and other forms of malware
- firewalls cannot protect against insider attackers and social engineering
- problems with email (urls, attachements)

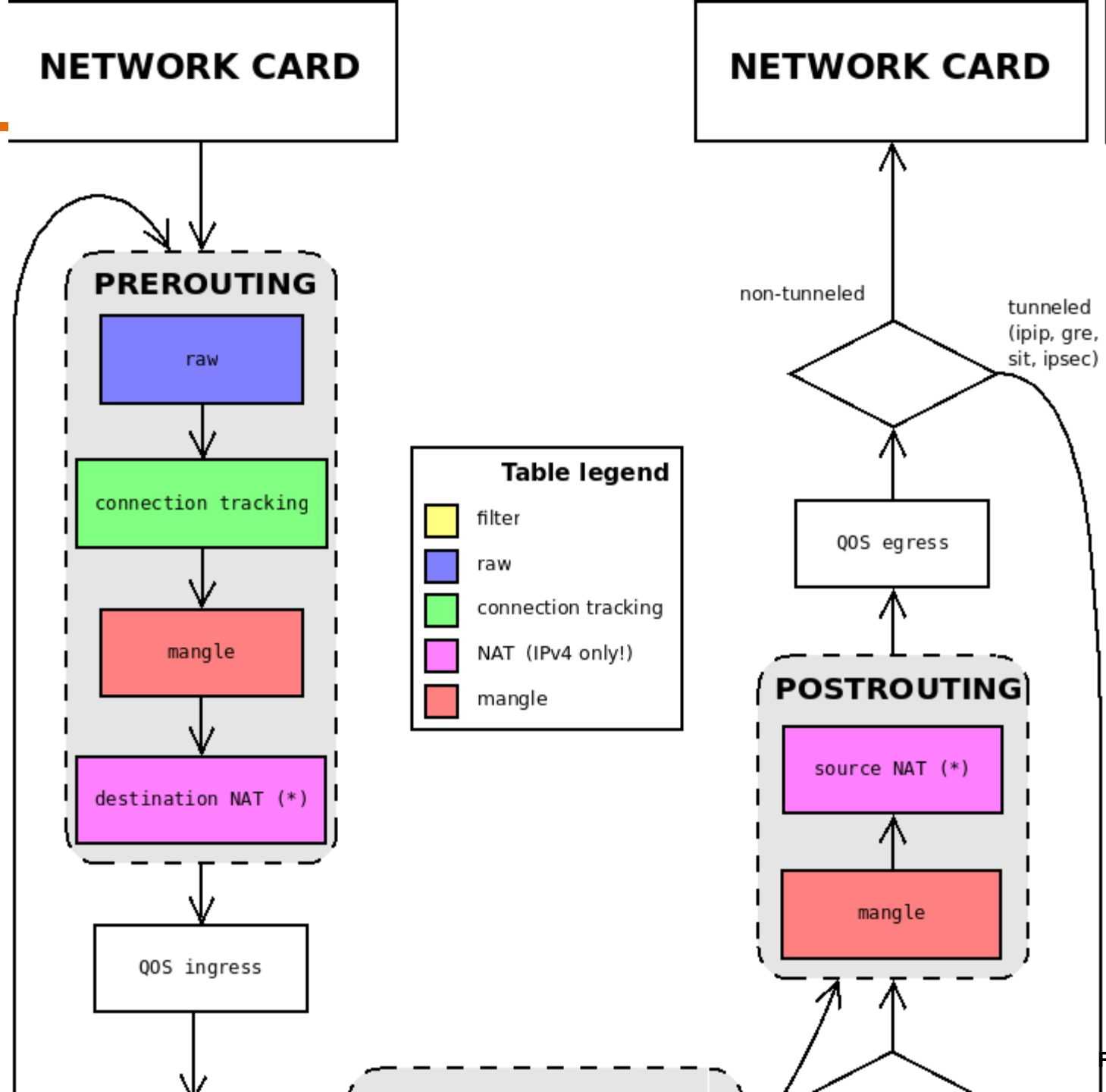
Example: Netfilter/IPTables

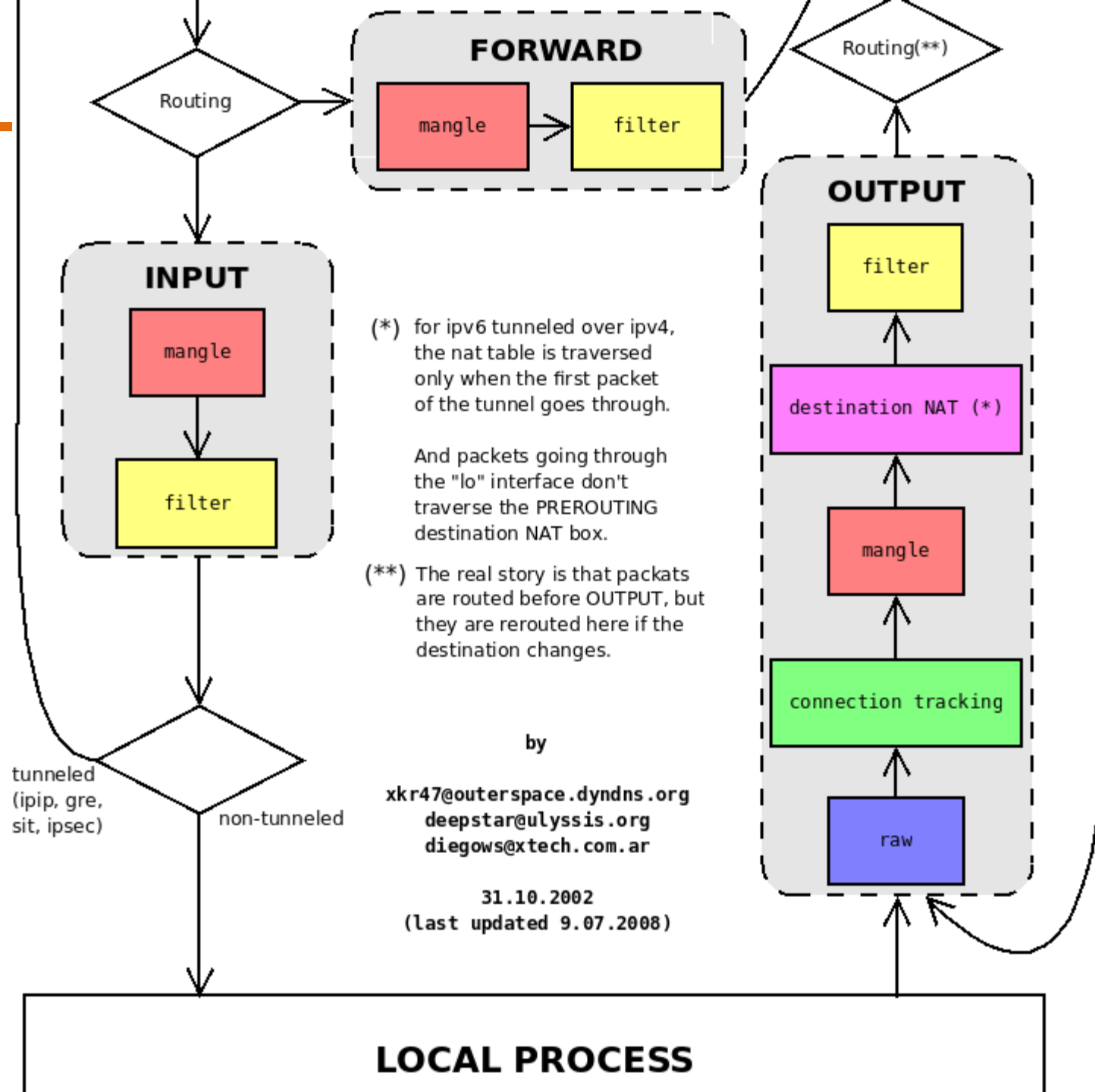
- Part of Linux kernel
- Can be managed by iptables commands (others: ufw, graphical interfaces)
- Close interaction with routing
- Provides (tables):
 - Filtering (accept, reject, drop, log), default table
 - NAT
 - Mangle (packet header modification)
 - Raw access
- Table = set of chains (PREROUTING, INPUT, FORWARD, OUTPUT and POSTROUTING.)
- Chain = ordered list of rules

Example: Netfilter/IPTables



- http://hydra.geht.net/tino/howto/linux/netfilter/packet_flow10.png





Example: Netfilter/IPTables

Some examples:

- Deny all traffic except http from outside on eth0
 - `iptables -A INPUT -i eth0 -p tcp --dport 80 -j ACCEPT`
 - `iptables -A INPUT -i eth0 -j DROP`
- Forward http traffic to web server in DMZ to special port
 - `iptables -t nat -A PREROUTING -j DNAT -d 1.2.3.4 -p tcp --dport 80 --to 10.0.1.1:8080`
- Forward special traffic to web server in DMZ
 - `iptables -t nat -A PREROUTING -j DNAT -d 1.2.3.4 -p tcp --dport 8080 --to 10.0.1.1:80`
- Simple NAT
 - `iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE`
 - `iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT`
 - `iptables -A FORWARD -i eth0 -o eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT`

Example: Netfilter/IPTables

Some examples:

- Log the forwarded p2p traffic (Napster)
 - `iptables -A FORWARD -j LOG --log-prefix "Peer2Peer" -p tcp --dport 6699`
- Log and drop heartbleed attack
 - `iptables -t filter -A INPUT -p tcp --dport 443 -m u32 --u32 \"52=0x18030000:0x1803FFFF\" -j LOG --log-prefix \"BLOCKED: HEARTBLEED\"`
 - `iptables -t filter -A INPUT -p tcp --dport 443 -m u32 --u32 \"52=0x18030000:0x1803FFFF\" -j DROP`

Example: Netfilter/IPTables

Some advanced examples (>100 extensions):

- Custom chains
- Mark: internal tagging of packets; marks can be used in routing or filtering
- Ulog: save whole packet to pcap, requires user space ulogd
- Netflow: create and send netflow data to collector (filter possible)
- Load balancing: nth (every nth packet) and random (given probability) module; do not forget to handle all packets (25 + 25 + 25 + 25 != 100)
- Limit: time based limits to avoid DDoS
- NFQUEUE (decision on userspace)
- Recent, string, time, quota, ttl, ...
- Further tips: <http://linuxgazette.net/108/odonovan.html>

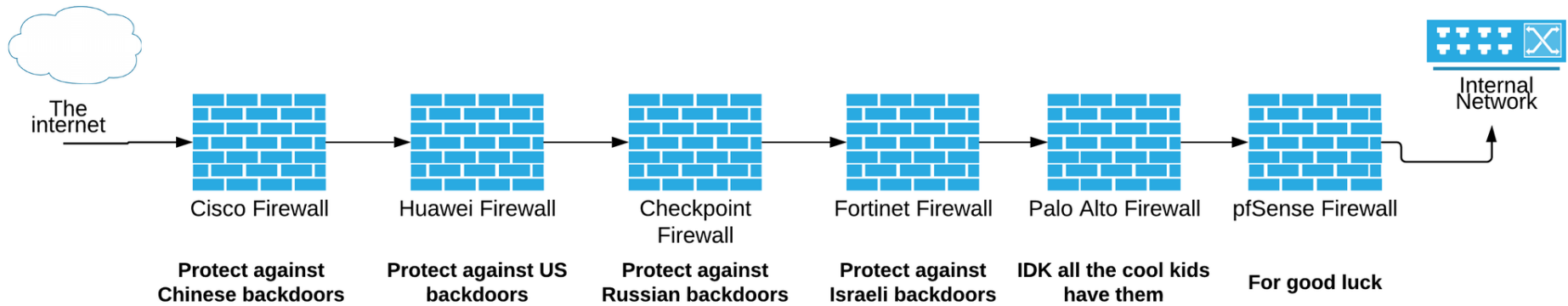
nftables

- New alternative of iptables (since 3.13 kernel, 2014)
- New syntax, *nft* command
- Iptables command still supported (compatibility layer)
- Less rules required compared to iptables
- Only the required tables and chains are registered (counters are also optional)
- Multi-action rules (e.g. log and drop)
- Unified IPv4 and IPv6 rules possible
- Concatenation, sets, maps, e.g.

```
nft add rule ip filter input ip saddr . ip daddr . ip protocol { 1.1.1.1 . 2.2.2.2 . tcp, 1.1.1.1 . 3.3.3.3 . udp } counter accept
```

```
nft add map filter whitelist { type ipv4_addr . inet_service : verdict \; }  
nft add element filter whitelist { 1.2.3.4 . 22 : accept }  
nft add rule filter input ip saddr . tcp dport vmap @whitelist
```

Which firewall to use?



Firewall summary

- Firewall is a must in any secure network
- Can prevent many known and unknown attacks
- Saves network resources
- But have limitations:
 - Firewalls may have software vulnerabilities, they may be hacked and penetrated
 - Even if correct, firewalls do not solve all security problems → one may have a false sense of security
 - Firewalls cannot protect against attacks that don't go through the firewall: data flow through CD, USB key [-> See Stuxnet]
 - Firewalls alone are not very efficient against new viruses and other forms of malware
 - Malicious traffic can be similar to benign traffic (most C&C is web based)
 - Firewalls cannot protect against insider attackers and social engineering

Control questions

- What is the main goal of a firewall?
- What is the difference between a packet filter and a stateful firewall?
- How an application layer firewall works?
- What is a chain in netfilter/iptables?
- What is a table in netfilter/iptables?
- What are the benefits of nftables?