

Network Security

Foundations of Cybersecurity

Luca Caviglione

Institute for Applied Mathematics and Information Technologies

National Research Council of Italy

luca.caviglione@cnr.it



University of Pavia – Department of Electrical, Computer and Biomedical Engineering

Outline

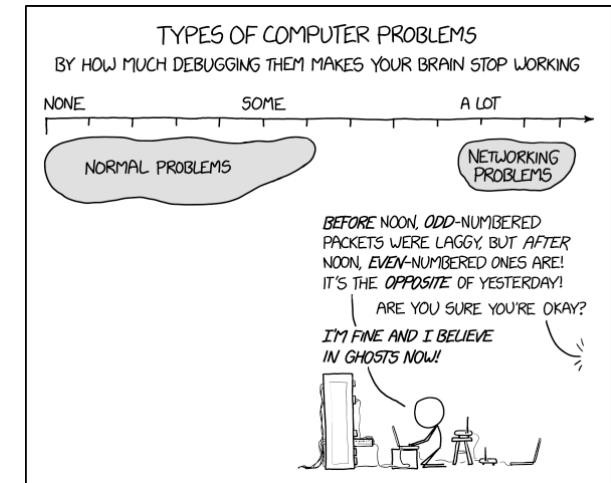
- Why network security in this course?
- Possible Network Attack Types (sniffing and spoofing)
- Denial of Service and Distributed Denial of Service
- Firewalls
- (Network) Vulnerability Scanning
- Policy Enforcing and Hardening
- Network Segmentation
- Network Address Translation - NAT
- Honeypot
- Automatic Scanning tools

Why Network Security in this Course?

- Networks are **pervasively** used in:
 - connected homes
 - Information Technology (IT) and Operational Technology (OT) scenarios
 - industrial and financial settings.
- Attacks against networked devices may result in:
 - loss of data
 - disruption of services and operational continuity
 - reduced trust of users
 - large-scale delivery of malicious software.
- The increasing diffusion of **IoT technologies** and **microservices**:
 - makes the overall attack surface difficult to control
 - offers great opportunities for reconnaissance.
- Any engineer should be aware of **basic** network threats!

Possible Network Attack Types

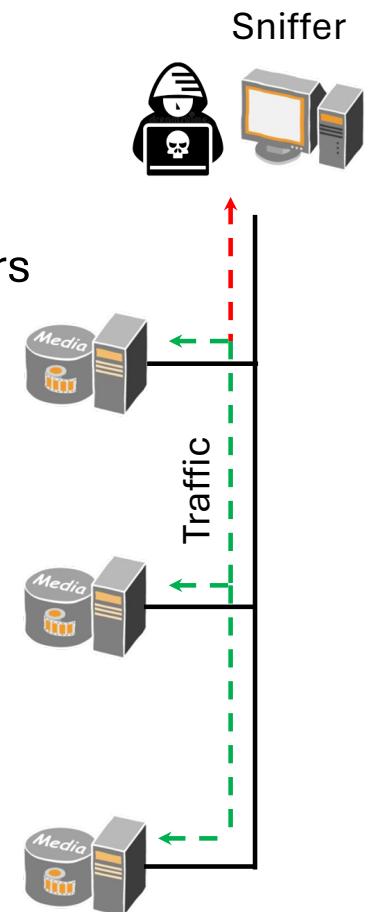
- Networks can be targeted by several attack types:
 - like any other complex systems
 - a **very vast attack surface** to control!
- Main attack types:
 - data interception
 - spoofing
 - redirections and routing attacks
 - violation of authentication/encryption mechanisms
 - DoS/DDoS
 - network vulnerability scanning
 - ...



Source: xkcd

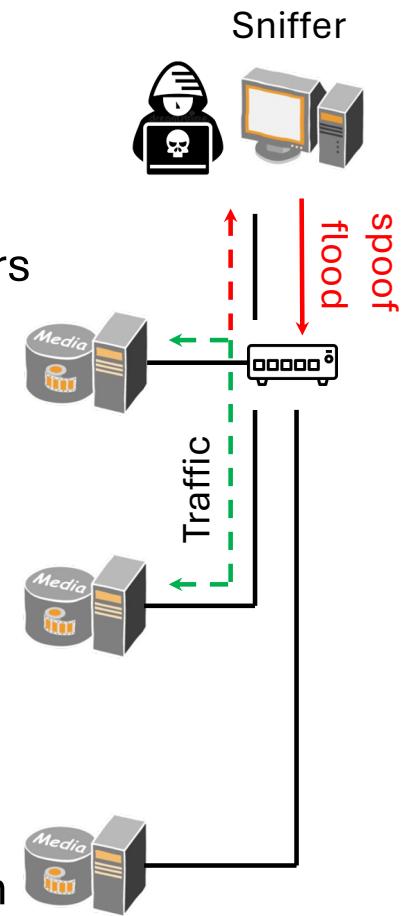
Sniffing (or eavesdropping)

- Network sniffing (eavesdropping):
 - an attacker captures traffic searching for sensitive information
 - mainly implemented via tools called sniffers or protocol analyzers
 - **example:** cleartext passwords, session tokens, confidential information, or “*harvest now, decrypt later*” campaigns.
- Network sniffer:
 - tool for collecting packets
 - can reconstruct streams
 - analyze data.
- Can work in **passive mode**:
 - traffic is simply captured and copied
 - requires the access to the broadcast domain.



Sniffing (or eavesdropping)

- Network sniffing (eavesdropping):
 - an attacker captures traffic searching for sensitive information
 - mainly implemented via tools called sniffer or protocol analyzers
 - **example:** cleartext passwords, session tokens, confidential information, or “*harvest now, decrypt later*” campaigns.
- Network sniffer:
 - tool for collecting packets
 - can reconstruct streams
 - analyze data.
- Traffic can be:
 - **redirected**, e.g., via ARP spoof attacks
 - “obtained” by forcing switched LANs to not filter traffic and act in fail-open mode (broadcast), e.g., **MAC flooding**



Sniffing (or eavesdropping)

- Most common network sniffers (or protocol analyzer) are:
 - Wireshark (cross platform)
 - tcpdump (Unix-like systems)
- Differences between **Windows** and **Unix** worlds:
 - Windows sniffers are typically based on **winpcap** or **npcap** libraries
 - Unix-like sniffers are typically based on the **libpcap** library
 - usually they require root/admin privileges.
- Some general rules for detecting the presence of a sniffer:
 - in general, very hard!
 - check host in promiscuous mode

```
[~] ~ ifconfigl grep PRO
en1: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
en2: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
en3: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
en0: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
```

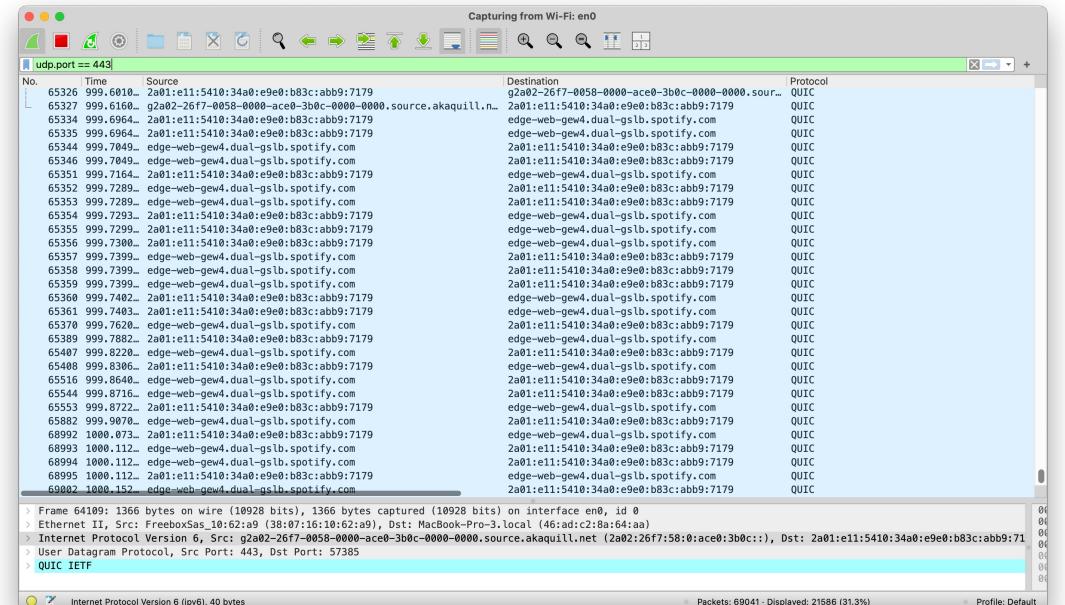


Traffic Sniffer: Wireshark



- Wireshark is a free and open-source network sniffer:
 - based on libpcap, <https://github.com/the-tcpdump-group/libpcap>
 - a huge amount of functionalities
 - <https://www.wireshark.org>
 - **give it a try!**

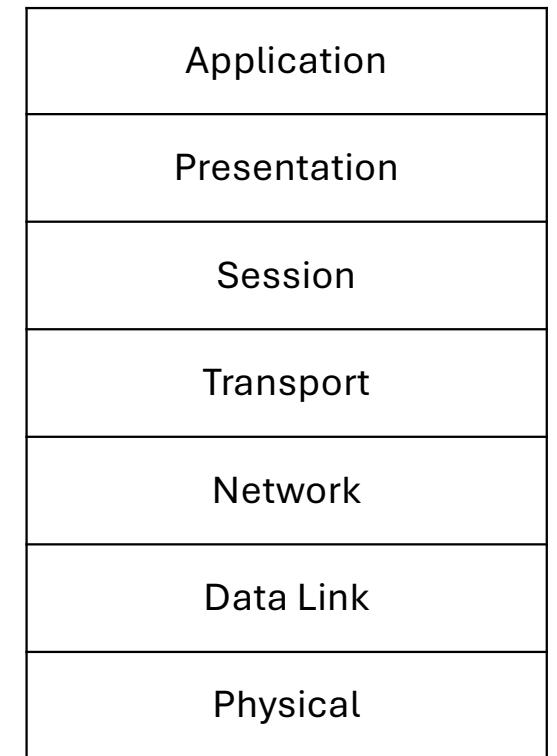
The tool should be considered as “agnostic”, i.e., it can be used for troubleshooting, understanding, as well as to collect traffic for reconnaissance and other malicious purposes (e.g., collecting passwords from weak protocols as in CVE-2021-23846).



...doing slides and listening to “Six Degrees of Inner Turbulence”

Spoofing

- Spoofing is an offensive technique where the attacker masquerades as another entity mainly to falsify the origin of the data.
- This attack can be launched against different layers of the protocol stack:
 - **Layer 2 (Data Link Layer)**: MAC spoofing and ARP spoofing
 - **Layer 3 (Network Layer)**: IP spoofing
 - **Layer 4 (Transport Layer)**: UDP or TCP spoofing
 - **Layer 7 (Application Layer)**: e-mail address spoofing, VoIP Caller ID spoofing, webpage spoofing, etc.



ISO/OSI Reference Model

ARP Spoofing

- This attack targets the Address Resolution Protocol (ARP) used in Local Area Networks (LAN):
 - the attacker sends fake (spoofed) ARP messages through the LAN
 - the goal is to associate the MAC address of the attacker with the IP address of another host
 - its scope is limited to local network segments
 - typically used to start other attacks, e.g., Man in the Middle and session hijacking.
- ARP spoofing may also be used as a non-attacking technique:
 - used in captive portals
 - Mobile IP
 - ...

The Hitchhiker's Guide to ARP

MAC: 00:08:2C:78:58:52

IP: 192.168.1.10

IP: 192.168.1.11

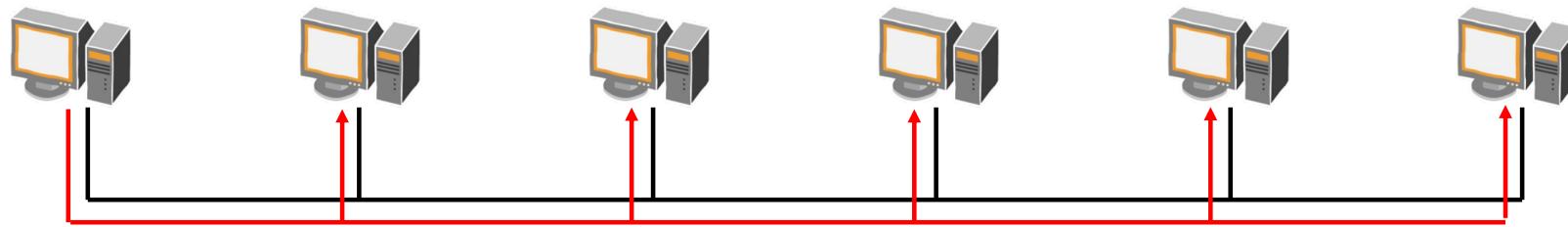
IP: 192.168.1.16

IP: 192.168.1.20

MAC: 00:00:60:AD:87:44

IP: 192.168.1.120

IP: 192.168.1.130



ARP Request: who has 192.168.1.130?

To:	ff:ff:ff:ff:ff:ff
From:	00:08:2C:78:58:52
MAC-T	00:00:00:00:00:00
MAC-S	00:08:2C:78:58:52
IP-T	192.168.1.130
IP-S	192.168.1.10

The Hitchhiker's Guide to ARP

MAC: 00:08:2C:78:58:52

IP: 192.168.1.10

IP: 192.168.1.11

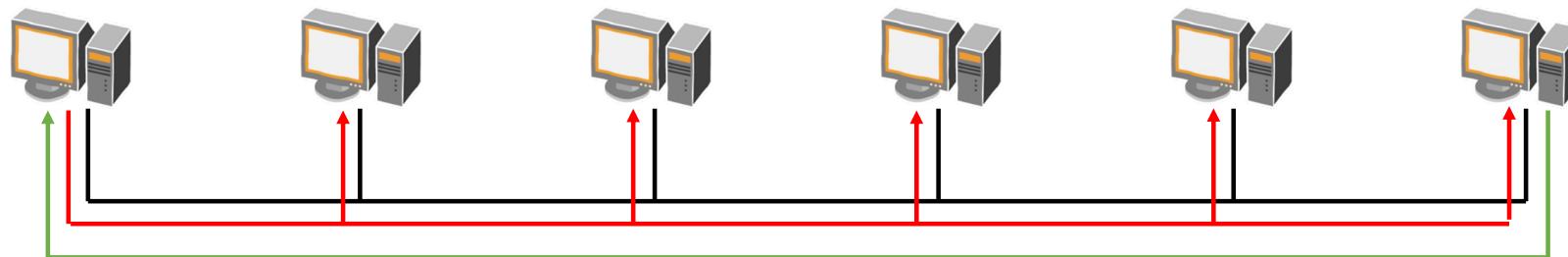
IP: 192.168.1.16

IP: 192.168.1.20

MAC: 00:00:60:AD:87:44

IP: 192.168.1.120

IP: 192.168.1.130



ARP Reply: 192.168.1.130 is at 00:00:60:AD:87:44

ARP Request: who has 192.168.1.130?

To:	ff:ff:ff:ff:ff:ff
From:	00:08:2C:78:58:52
MAC-T	00:00:00:00:00:00
MAC-S	00:08:2C:78:58:52
IP-T	192.168.10.130
IP-S	192.168.10.10

To: 00:08:2C:78:58:52
From: 00:00:60:AD:87:44

MAC-S 00:00:60:AD:87:44

MAC-T 00:08:2C:78:58:52

IP-S 192.168.10.130

IP-T 192.168.10.10

The Hitchhiker's Guide to ARP

MAC: 00:08:2C:78:58:52

IP: 192.168.1.10

IP: 192.168.1.11

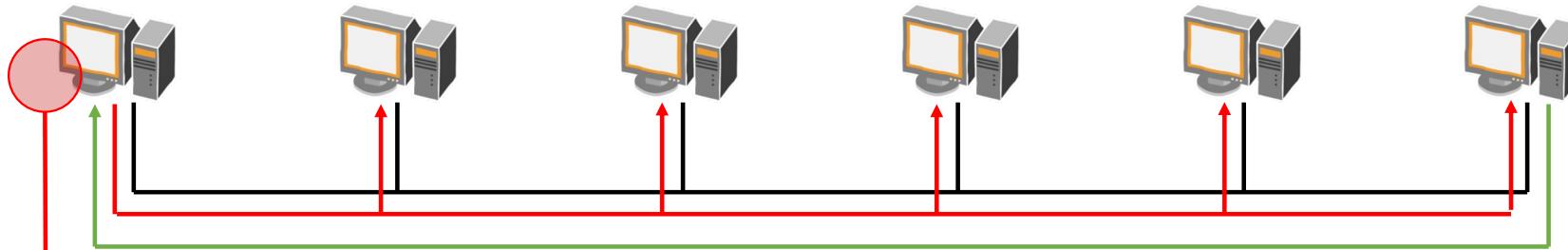
IP: 192.168.1.16

IP: 192.168.1.20

MAC: 00:00:60:AD:87:44

IP: 192.168.1.120

IP: 192.168.1.130



ARP Table is updated with:
192.168.1.130 at 00:00:60:AD:87:44

ARP Reply: 192.168.1.130 is at 00:00:60:AD:87:44

ARP Request: who has 192.168.1.130?

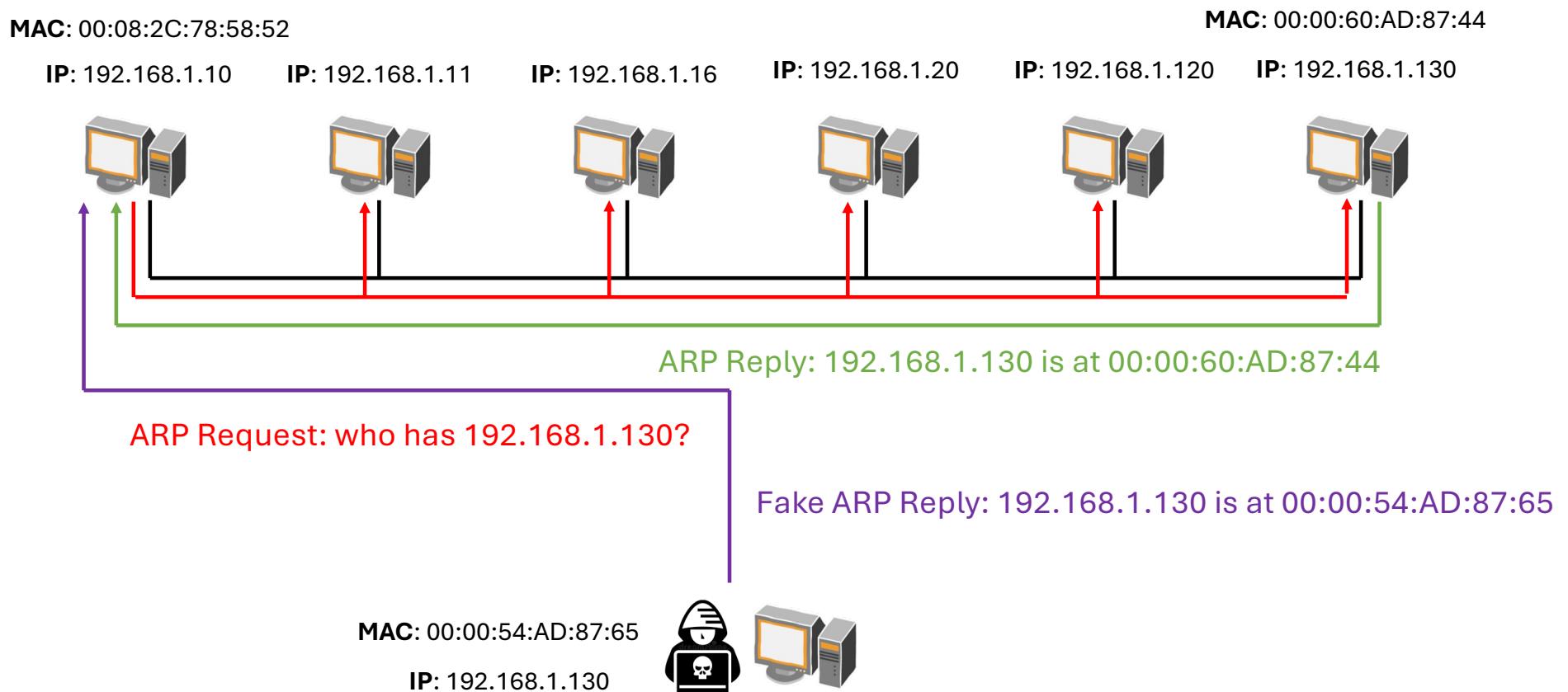
To:	ff:ff:ff:ff:ff:ff
From:	00:08:2C:78:58:52
MAC-T	00:00:00:00:00:00
MAC-S	00:08:2C:78:58:52
IP-T	192.168.10.130
IP-S	192.168.10.10

To: 00:08:2C:78:58:52
From: 00:00:60:AD:87:44

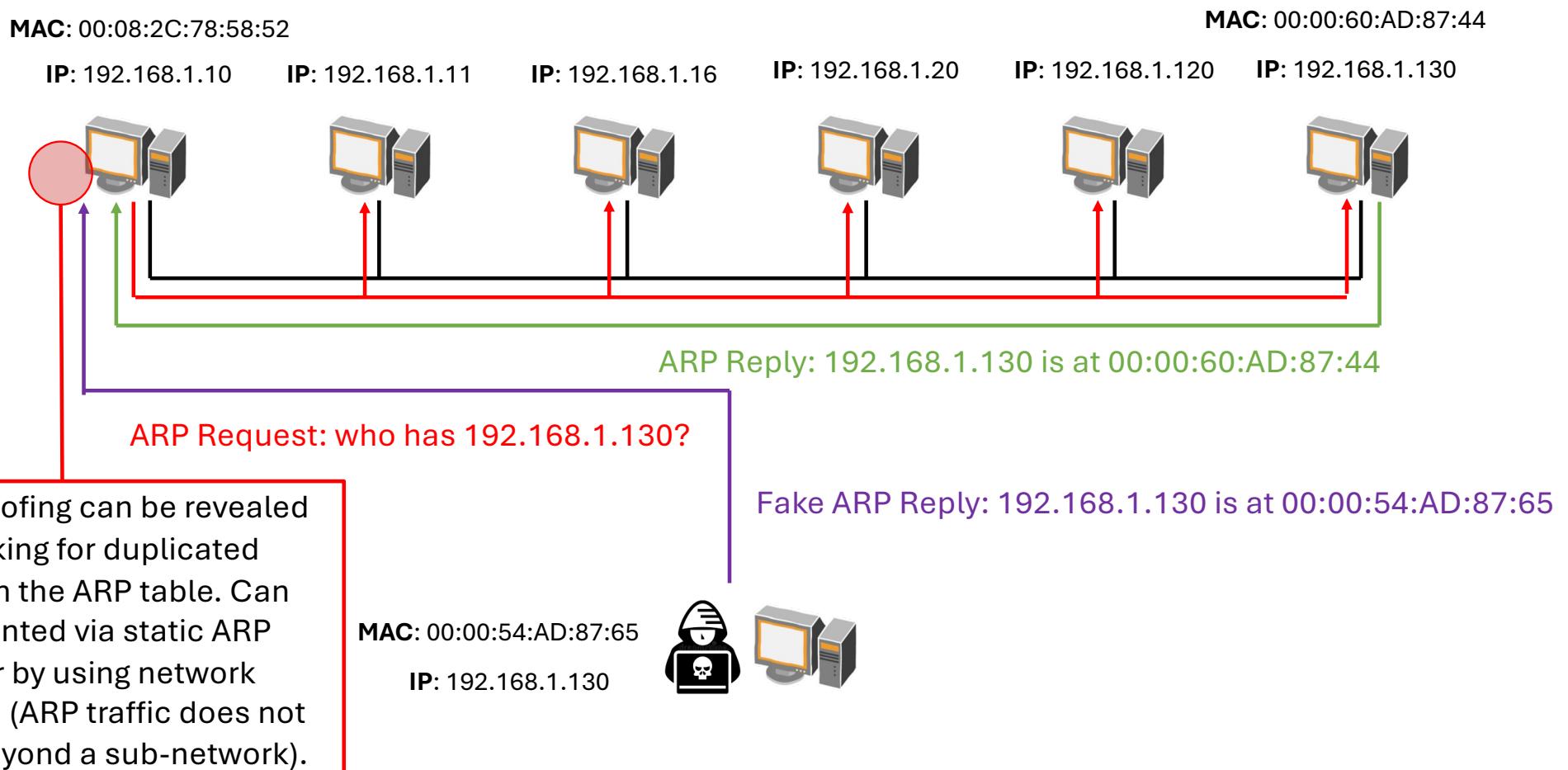
MAC-S 00:00:60:AD:87:44
MAC-T 00:08:2C:78:58:52
IP-S 192.168.10.130
IP-T 192.168.10.10

(Try: arp -a)

ARP Spoofing



ARP Spoofing



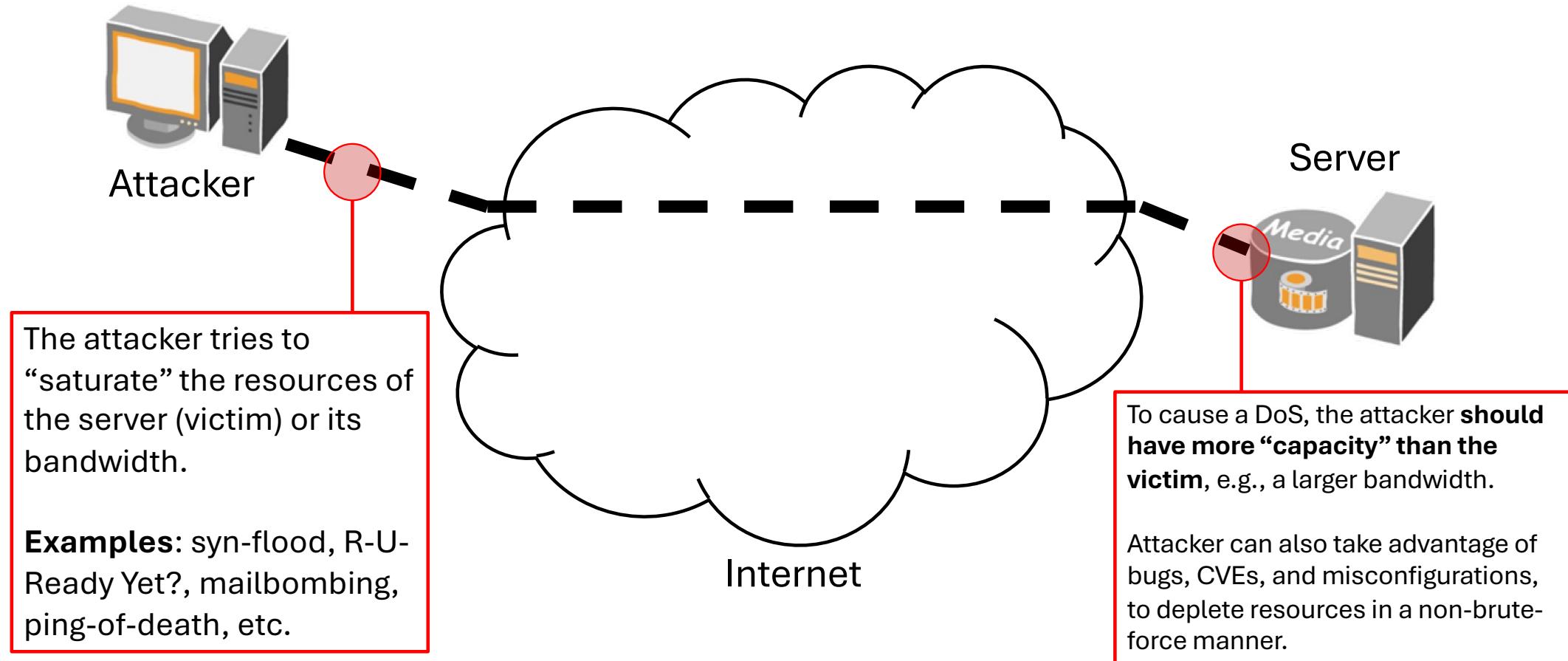
Denial of Service

- Denial of Service (DoS) attacks aim at making a service or a network unavailable.
- Main types of DoS attacks:
 - **volumetric**: “flood” the target with a huge volume of traffic
 - **protocol**: take advantage of vulnerabilities of protocols to quickly deplete resources of the victim
 - **application**: craft the attack against a specific application/service to cause a crash or exhaust resources.
- Scale of attacks:
 - **DoS**: utilizes a single source
 - **Distributed DoS (DDoS)**: uses a network of compromised nodes.

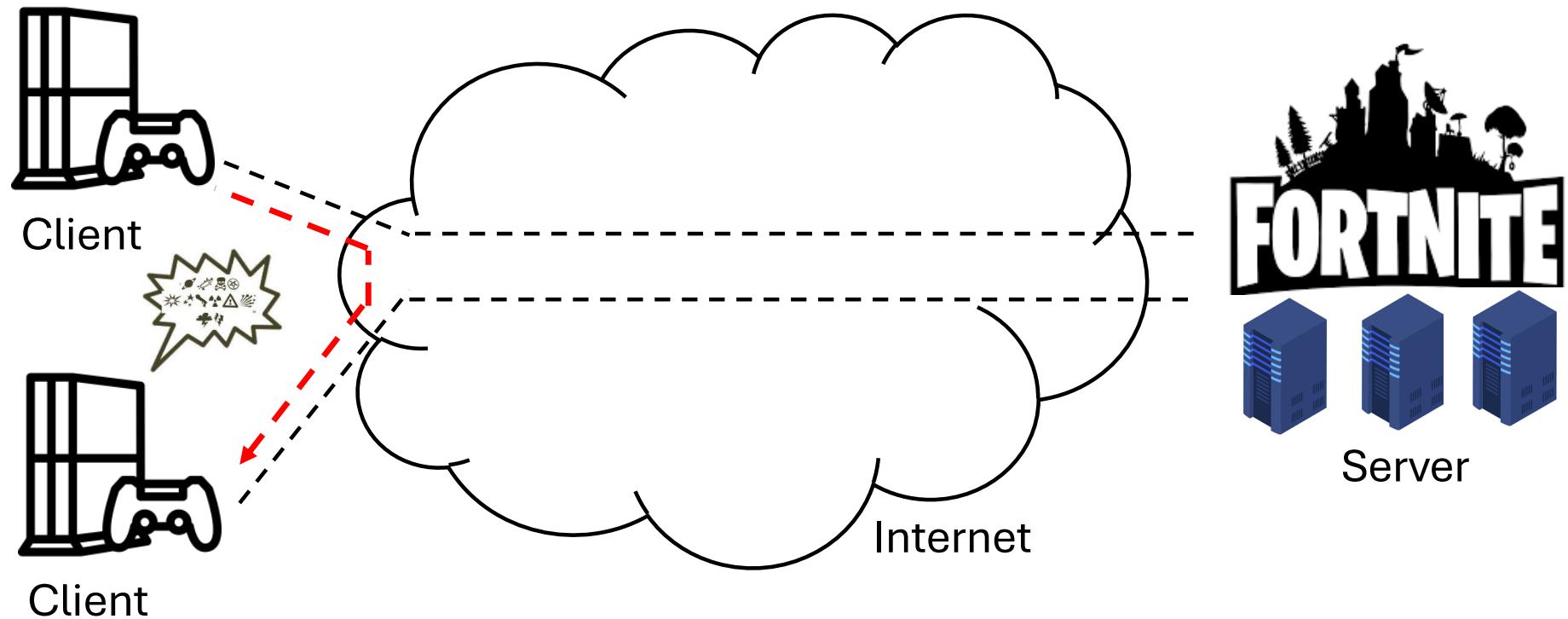


r/ProgrammerHumor

Denial of Service

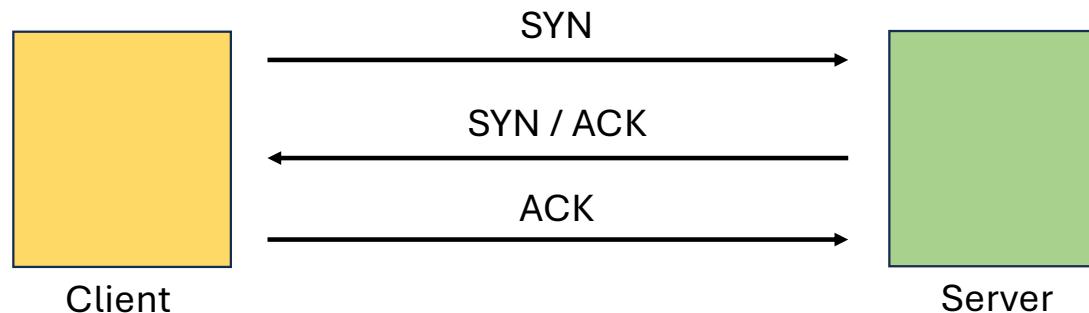


Denial of Service: Getting DoSsed

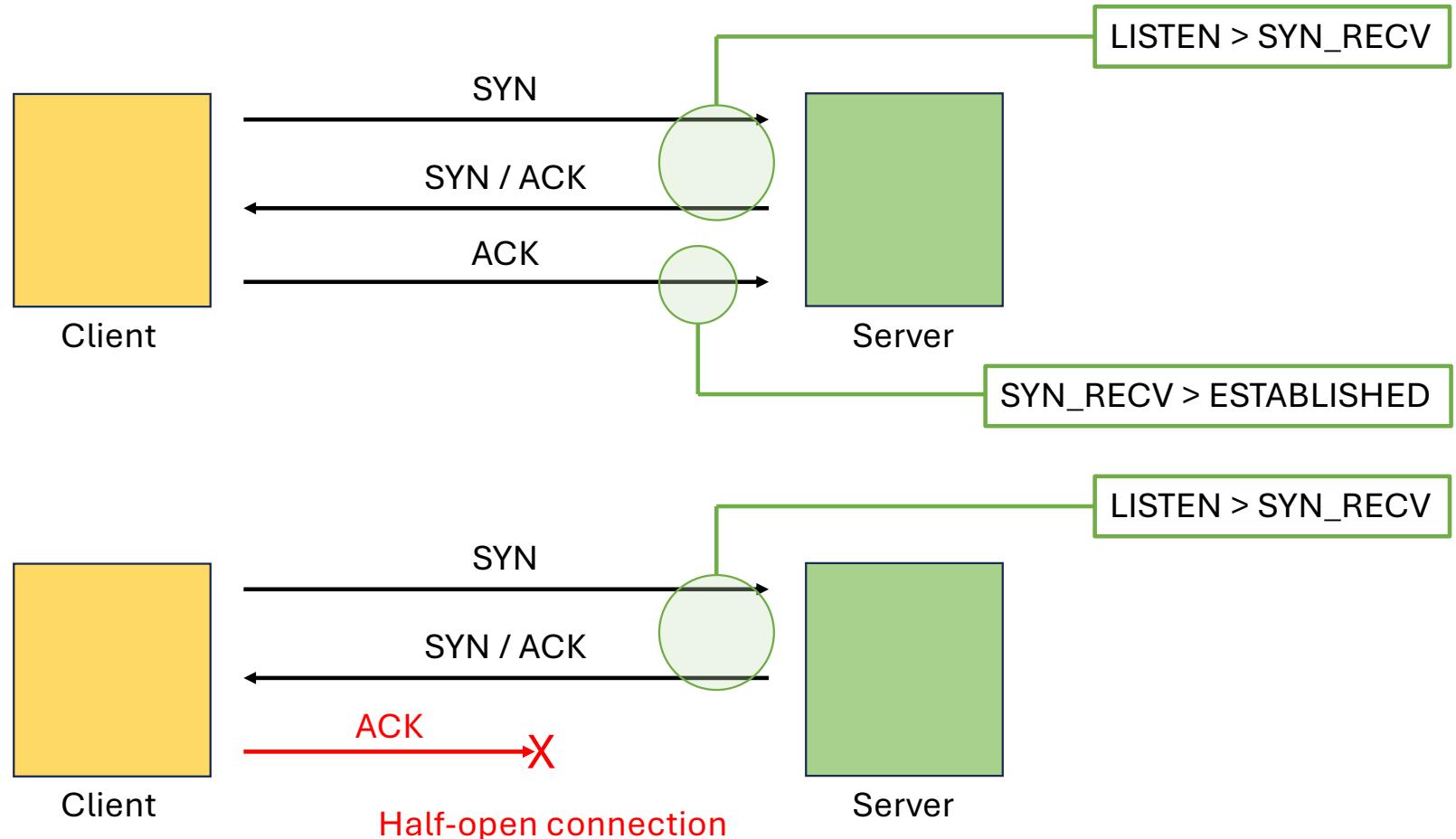


Denial of Service: SYN Flood

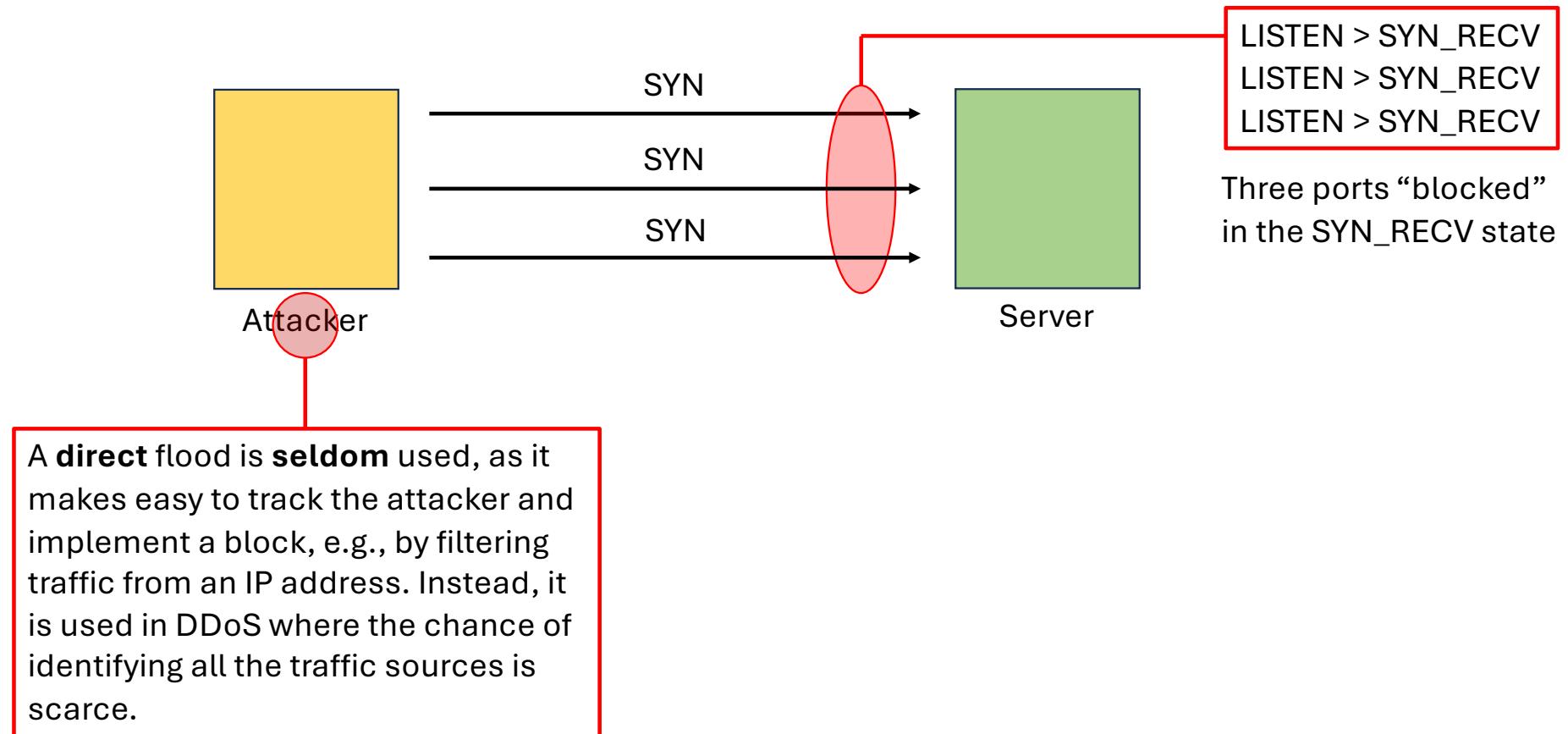
- The attack exploits the **three-way handshake** of TCP:
 - **SYN**: the client initiates a connection with a TCP segment with the SYN flag set to 1 and communicates the Initial Sequence Number (ISN)
 - **SYN/ACK**: the server responds with a TCP segment with both the SYN and ACK flags set to 1 and acknowledge the client ISN and communicates its own ISN
 - **ACK**: the client acknowledge the response by setting the ACK flag to 1 of its TCP segment as well as the server ISN.



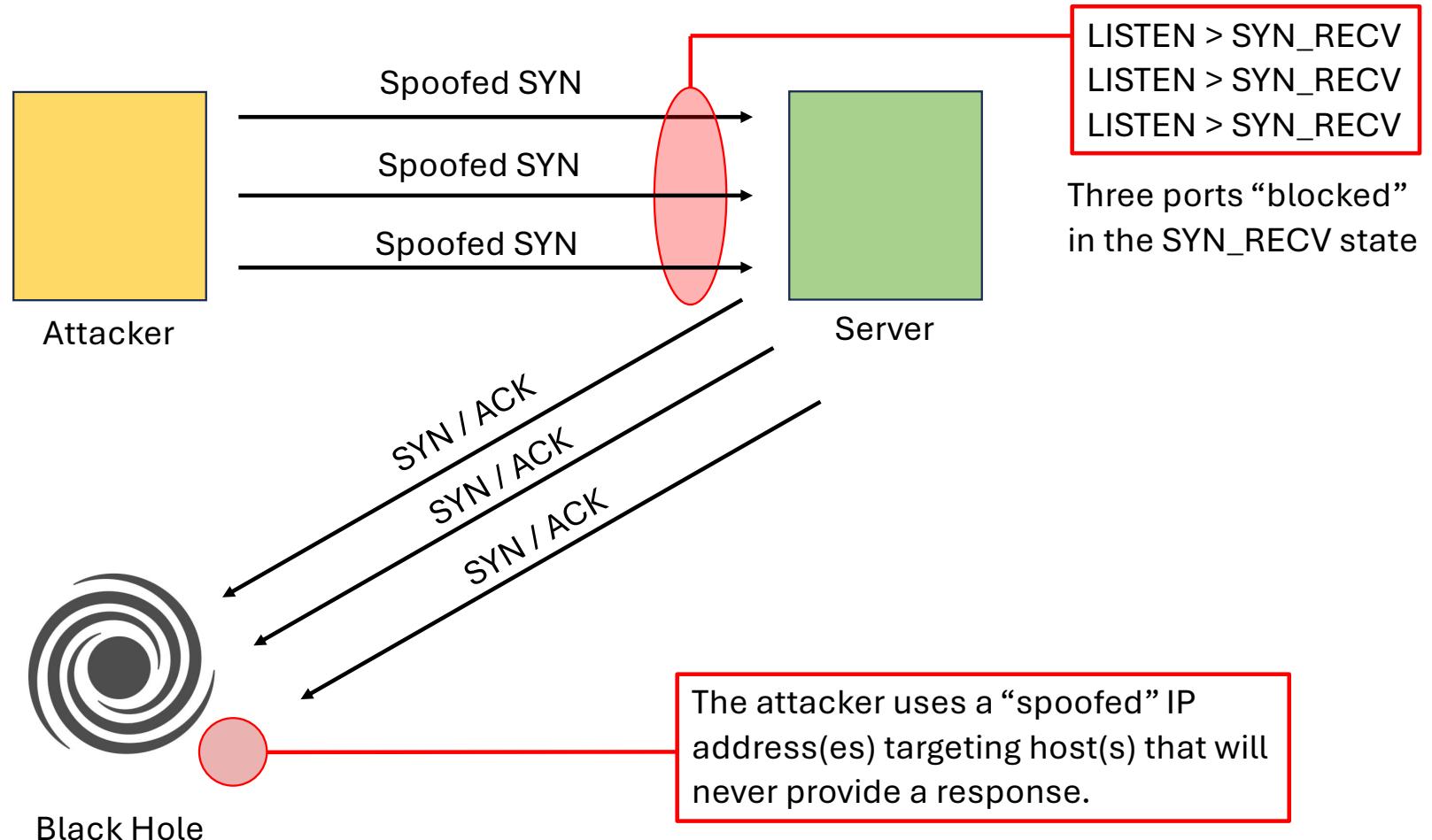
Denial of Service: SYN Flood



Denial of Service: SYN Flood



Denial of Service: SYN Flood

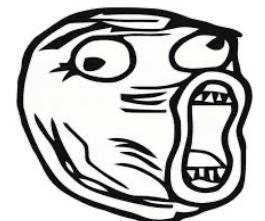


Denial of Service



Meta datacenter in Luleå (Sweden)

DoS



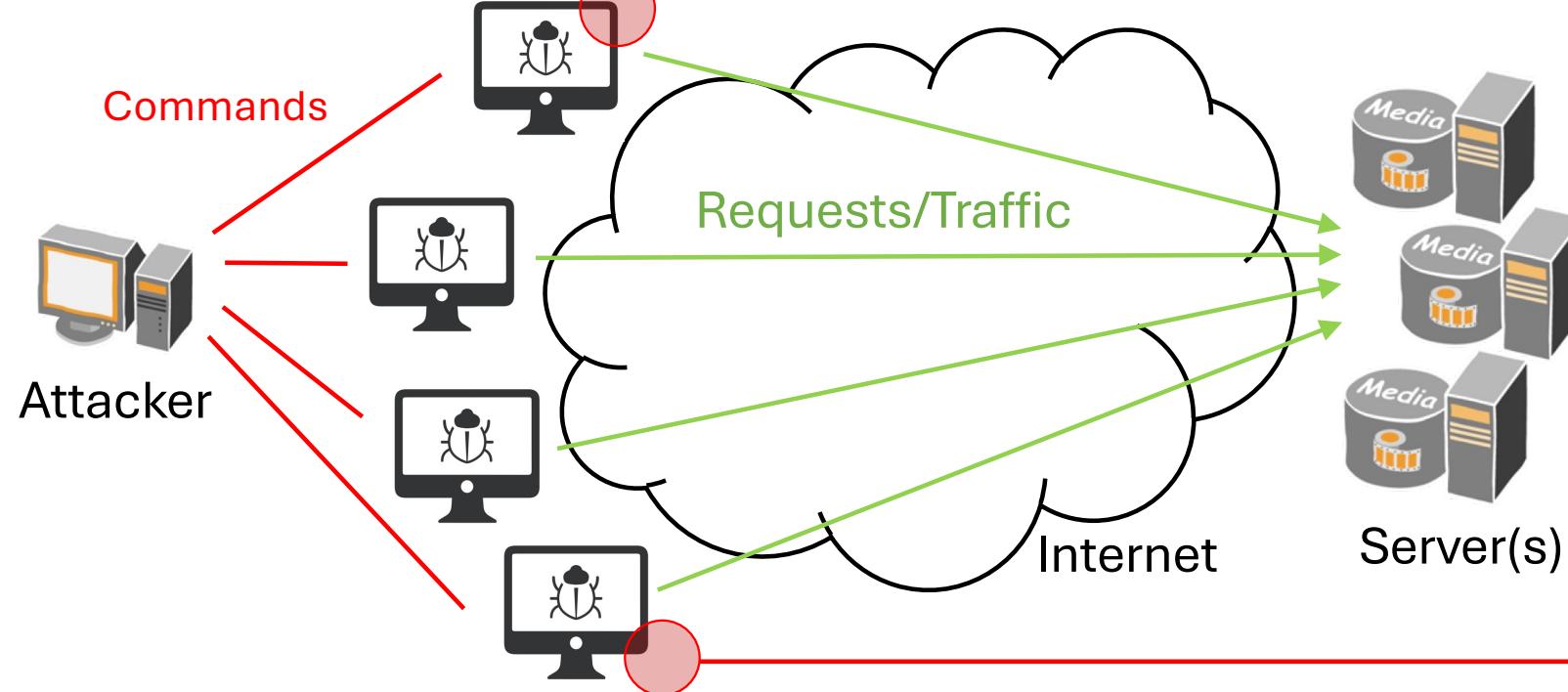
Distributed Denial of Service

- A DoS **usually** requires a **non-negligible amount of resources**.
- Distributed Denial of Service (DDoS):
 - a large population of nodes acts as a **coordinated** group
 - the attack is launched from **multiple** sources
 - **resources** are borrowed from a network of **compromised** devices
 - compromised hosts are defined as **zombies**
 - the attacker is often a sort of “**orchestrator**”.
- DDoS attacks:
 - are effective and hard to mitigate
 - the attacker can be difficult to identify
 - **make each device of interest due to its resources**.



Distributed Denial of Service

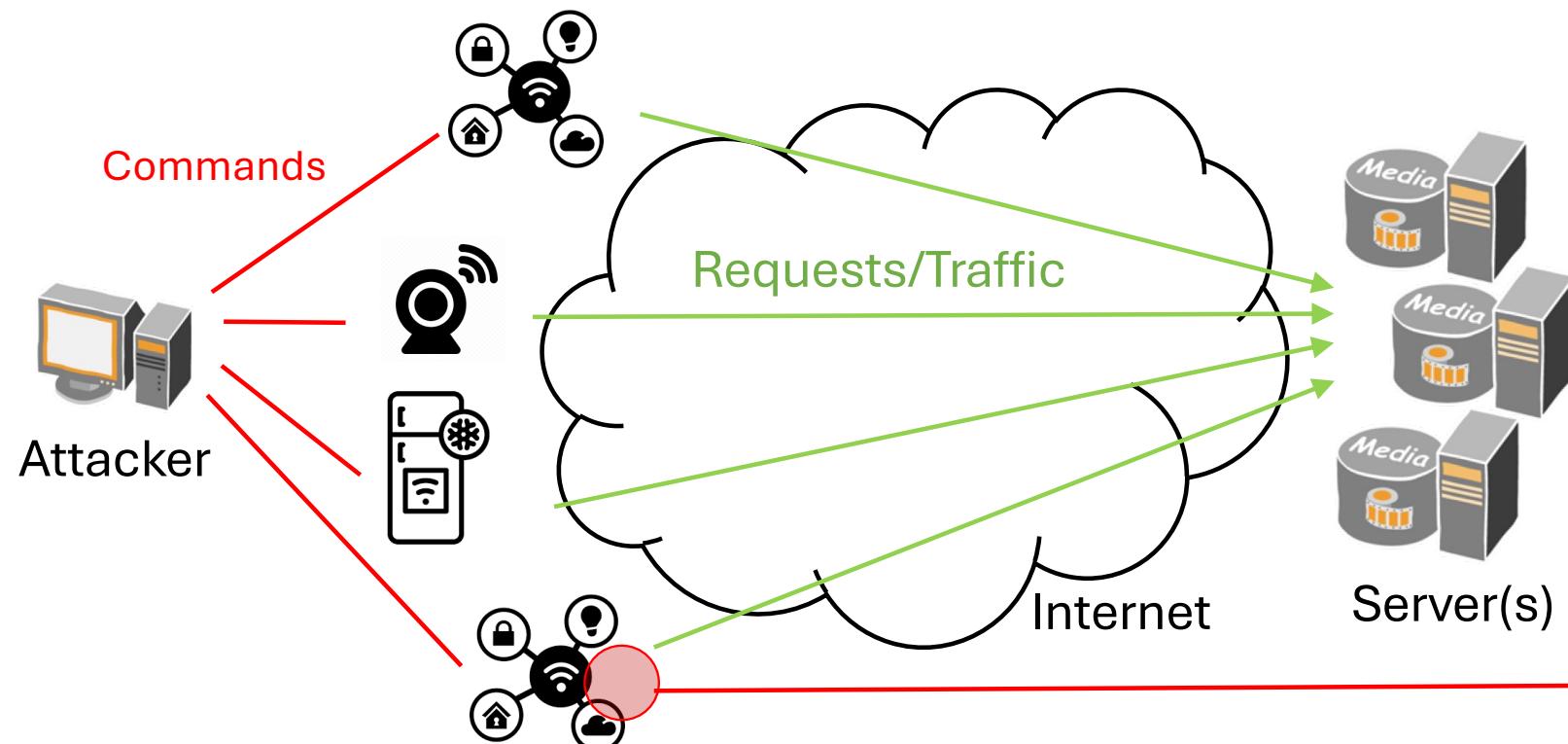
The network of compromised nodes used by the attacker is often called a **botnet**.



A botnet can also be available through a payment: this is a type of **Crime-as-a-Service** model.

Every component of the botnet is important to increase the number of resources that the attacker can use against the victim. **This makes almost any device and host exposed on the Internet attractive.**

Distributed Denial of Service: Mirai (2016)



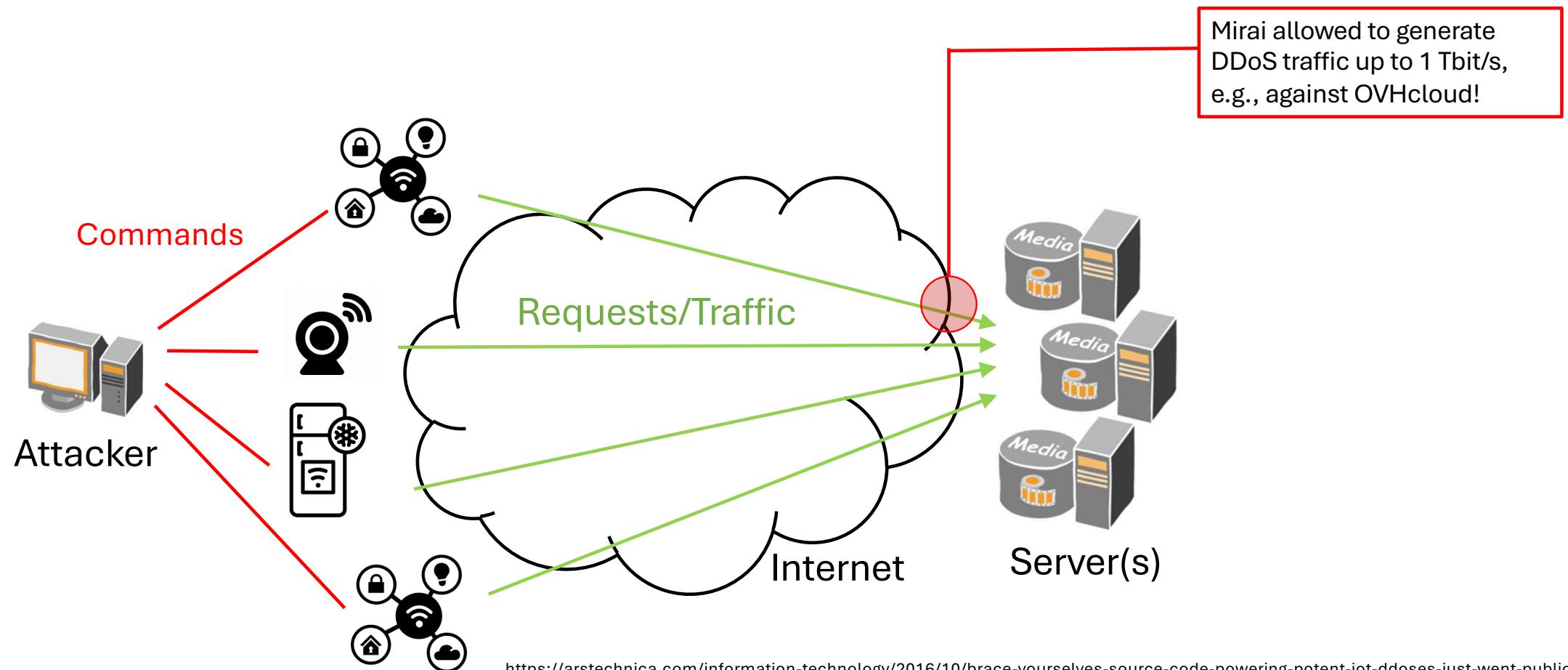
Mirai (未来) is a malware targeting devices based on ARC/Linux, e.g., IP cameras and routers.

Devices are infected due to weak credentials, i.e., more than 60 default user + password entries. Infection lasts until reboot.

Infection spreads via nodes scanning neighbors and do not contact specific IP ranges, e.g., DoD.

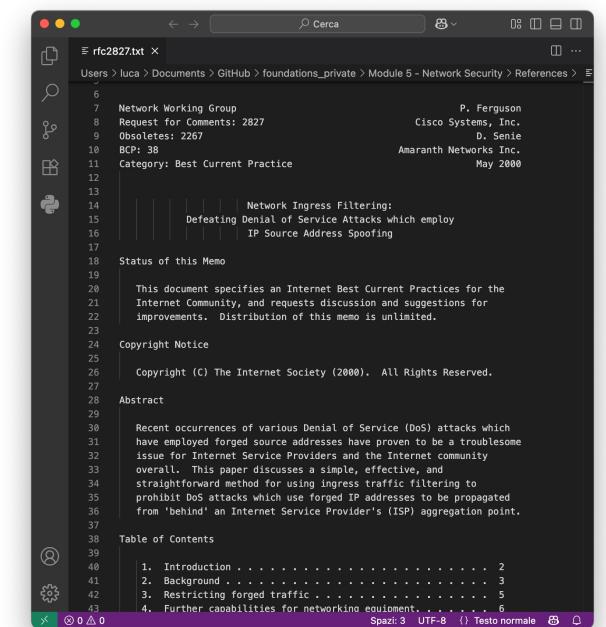
At the peak of popularity, about 1 million of infected devices.

Distributed Denial of Service: Mirai (2016)



(Basic) Defense Mechanisms Against DoS

- Preventing a DoS/DDoS attack is **not an easy task**:
 - sudden or large peaks of traffic could be legitimate.
- **Four possible main line of defense against DoS.**
- **Prevention:**
 - act before the attack to not deny the service to customers
 - **examples:** provide backup resources, act on configurations.
- **Detection and filtering:**
 - identify an ongoing attack and perform mitigation
 - **examples:** search for suspicious patterns and “drop” malicious traffic.
- **Attack source traceback and identification:**
 - find sources during the attack to prevent future, imminent waves
 - **example:** traffic analysis
- **Attack reaction:**
 - after the attack, improve the security posture and report
 - **example:** improve configuration and deploy anti-spoofing filters.



Example: RFC 2827

Firewalls

- A firewall is security tool able to control the network traffic:
 - inbound and outbound flows
 - can be implemented in hardware or software
 - creates a bridge between an internal network or a single host/device towards an external network.
- How a firewall works (roughly):
 - analyzes traffic
 - allows/blocks traffic according to a set of rules
 - provides alarms
 - logs information.



r/technicallythetruth

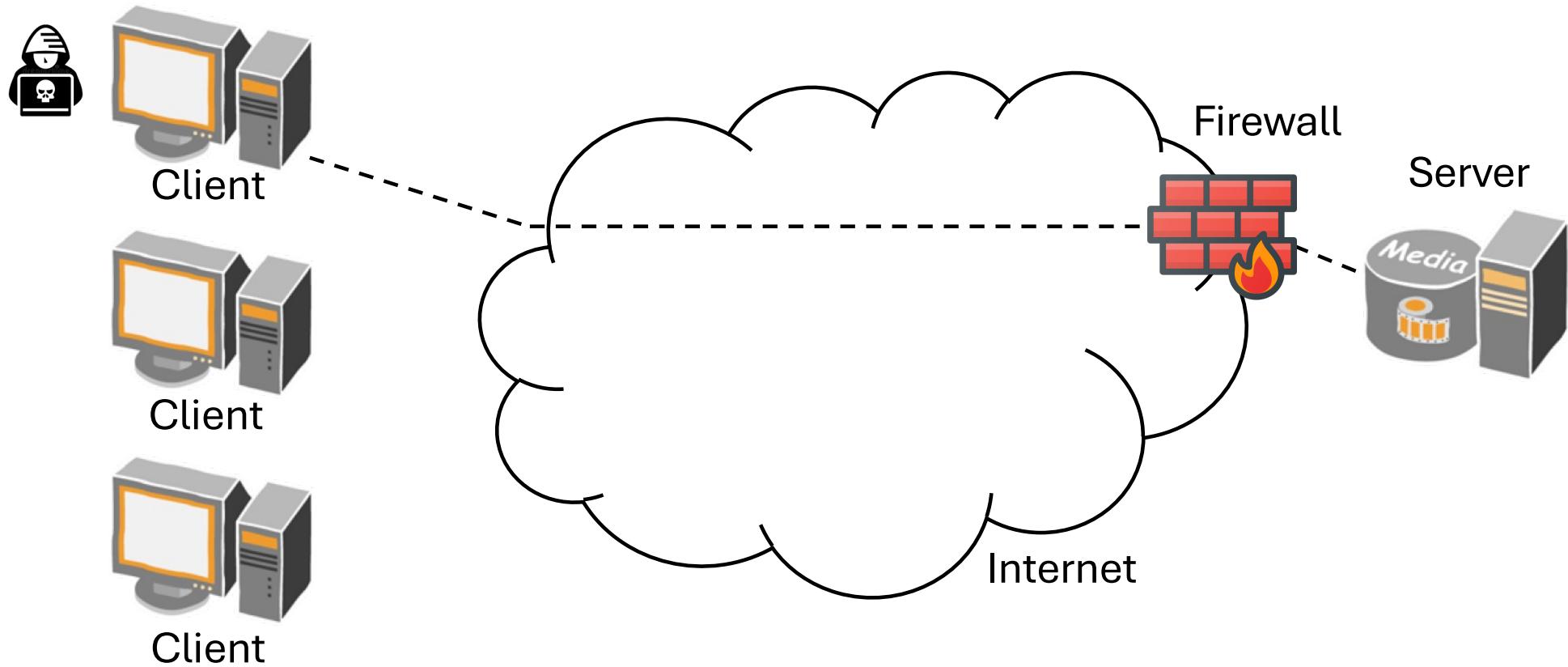
Firewalls

- There are several types of firewalls.
- A possible classification is based on the filtering method:
 - **packet filtering**: the simplest type, packets are filtering according to rules applied to the headers
 - **application layer**: traffic is inspected at the application layer, thus allowing more in-depth inspection
 - **proxy - Web application**: act over specific services, for instance Web traffic.
- Another possible classification is based on how firewalls are placed:
 - **network/perimeter** firewalls: physical or software devices placed at the edge of the network to be protected or before an external network, e.g., the Internet
 - **internal**: placed to segment different zones of a network and enforce segmentation
 - **personal** firewalls: software processes placed on the host to be protected.

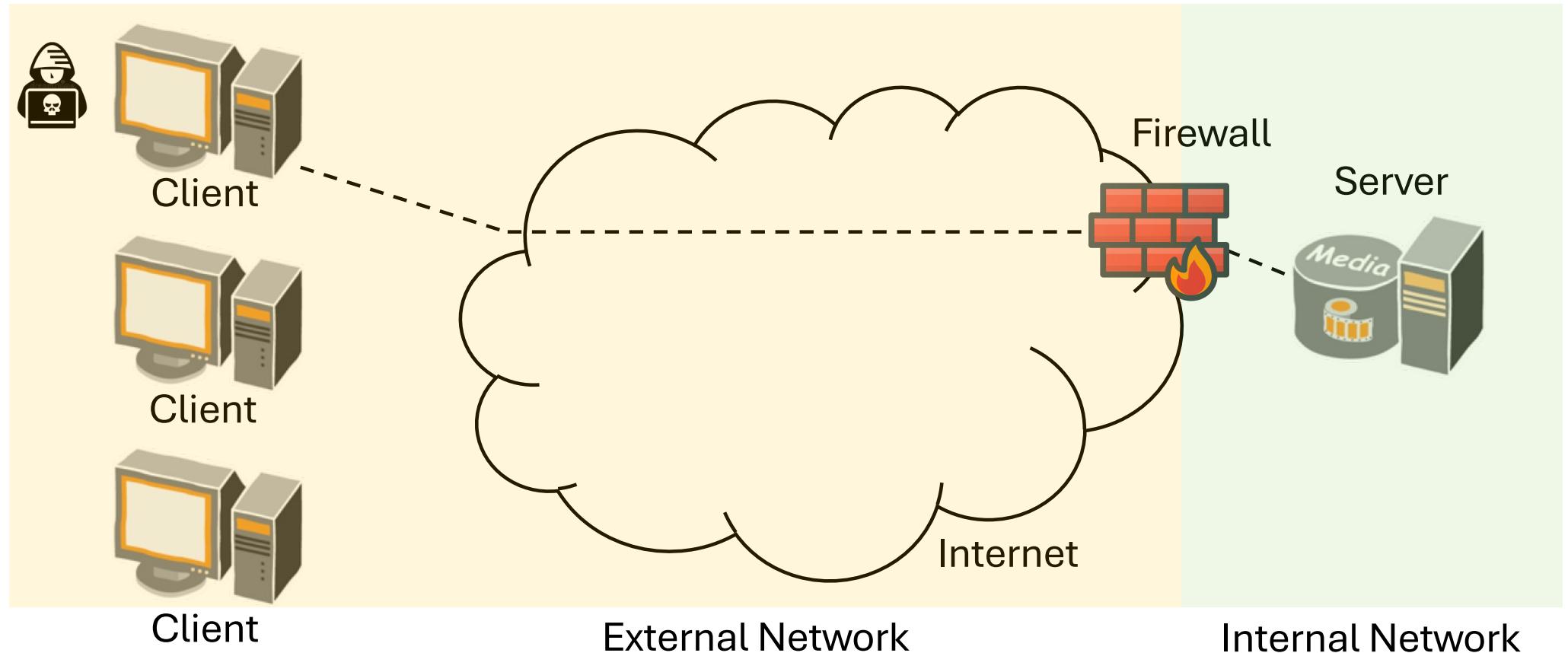
Firewalls: Packet Filters

- Packet filters:
 - also named network layer firewalls
 - operate at ISO/OSI L3 on a packet-by-packet basis
 - apply a set of rules on incoming/outcoming IP packets
 - as the outcome, the packet is forwarded or discarded
- Filtering rules rely on the information available in network packets:
 - source/destination IP addresses
 - transport protocol
 - source/destination ports
 - input/output interfaces

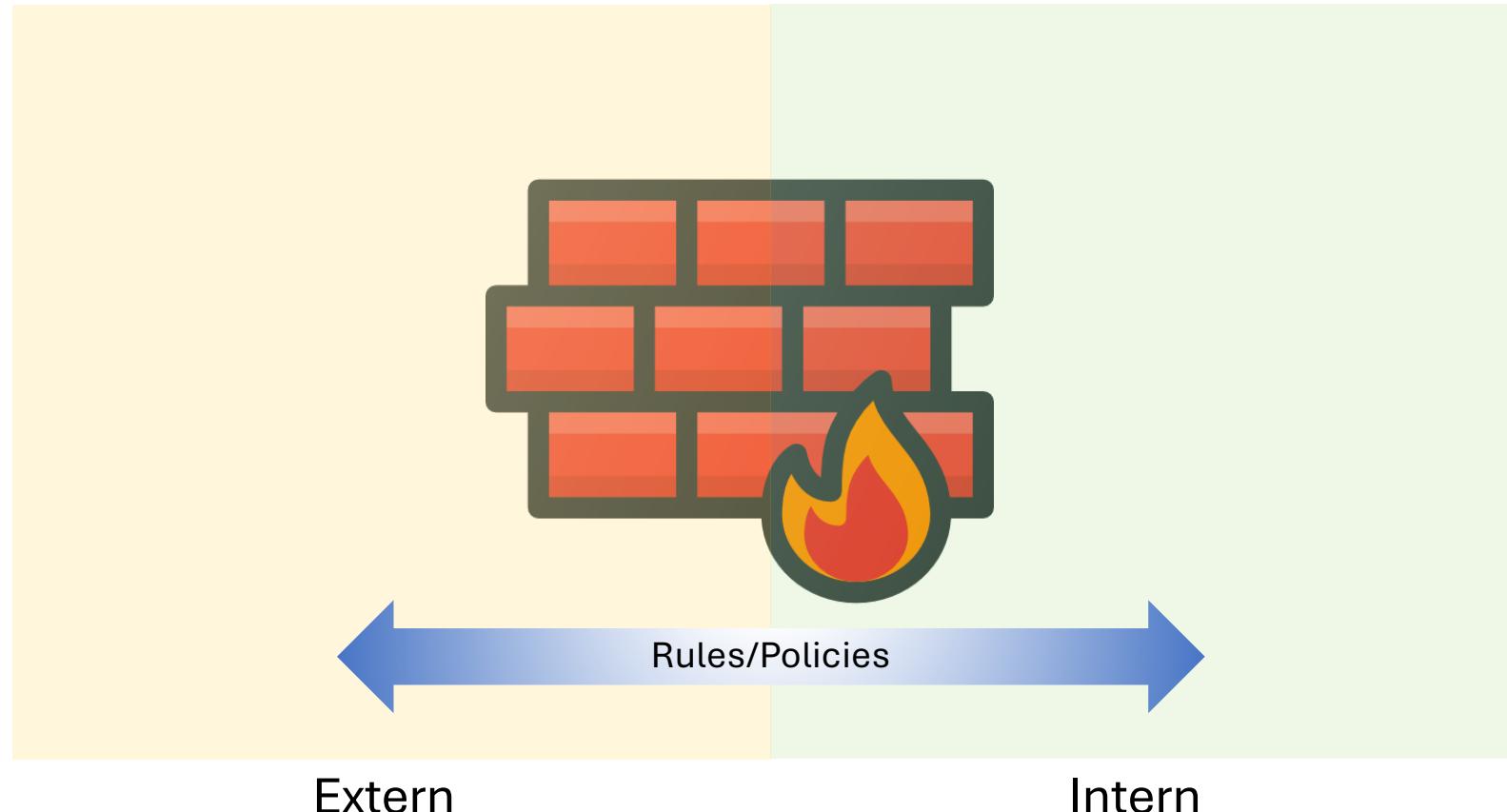
Firewalls: Packet Filters



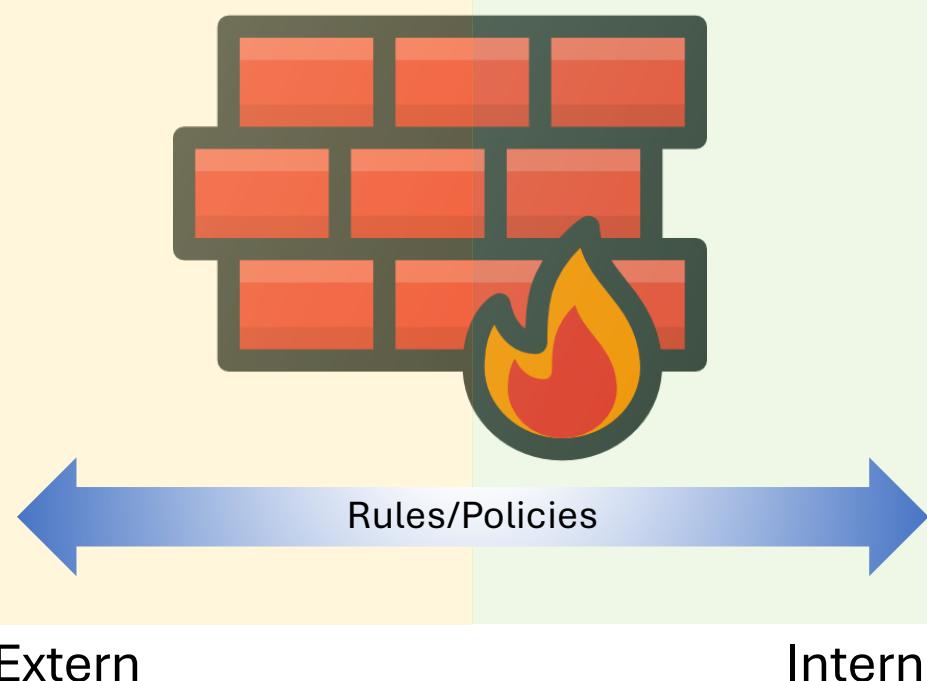
Firewalls: Packet Filters



Firewalls: Packet Filters



Firewalls: Packet Filters



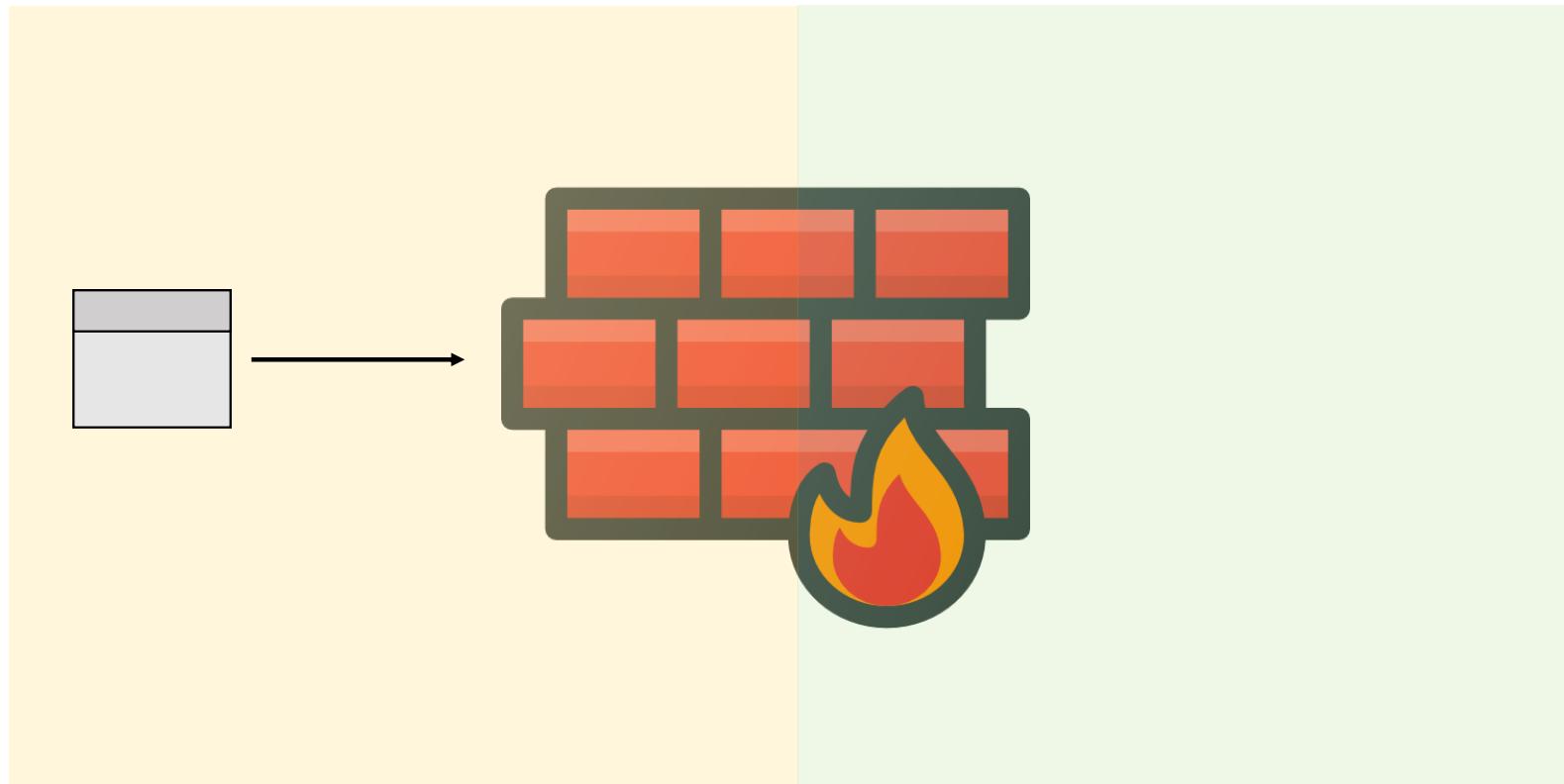
Two possible default policies

Default Allow: what is not expressly prohibited is permitted.

Default Deny: what is not expressly permitted is prohibited.

In general, the “Default Deny” scheme is preferred as it is considered more conservative, and hence more secure.

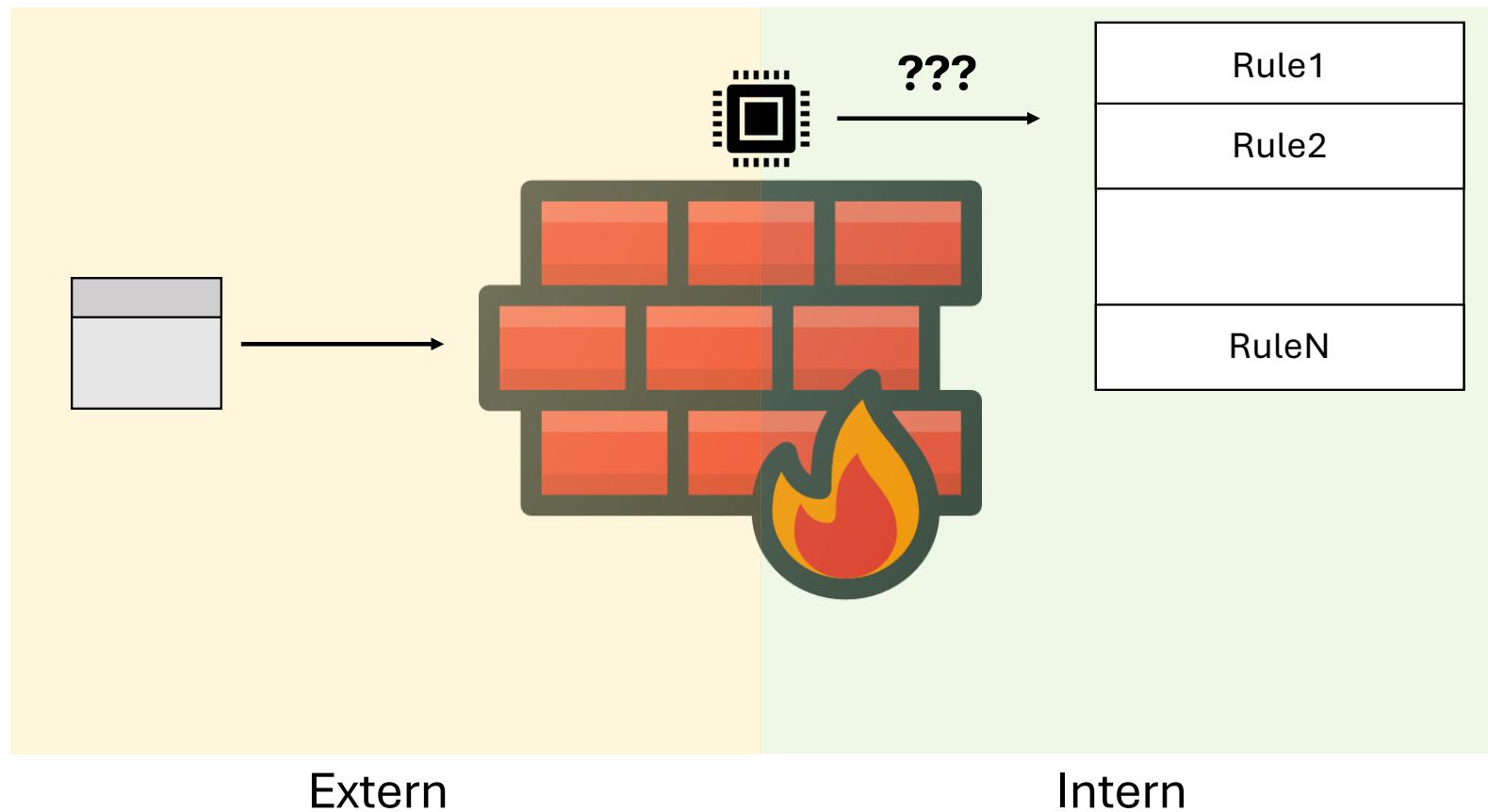
Firewalls: Packet Filters



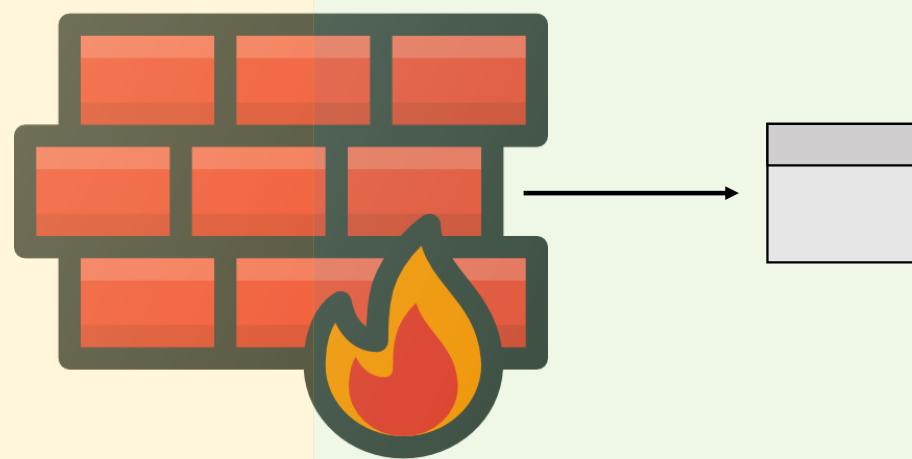
Extern

Intern

Firewalls: Packet Filters



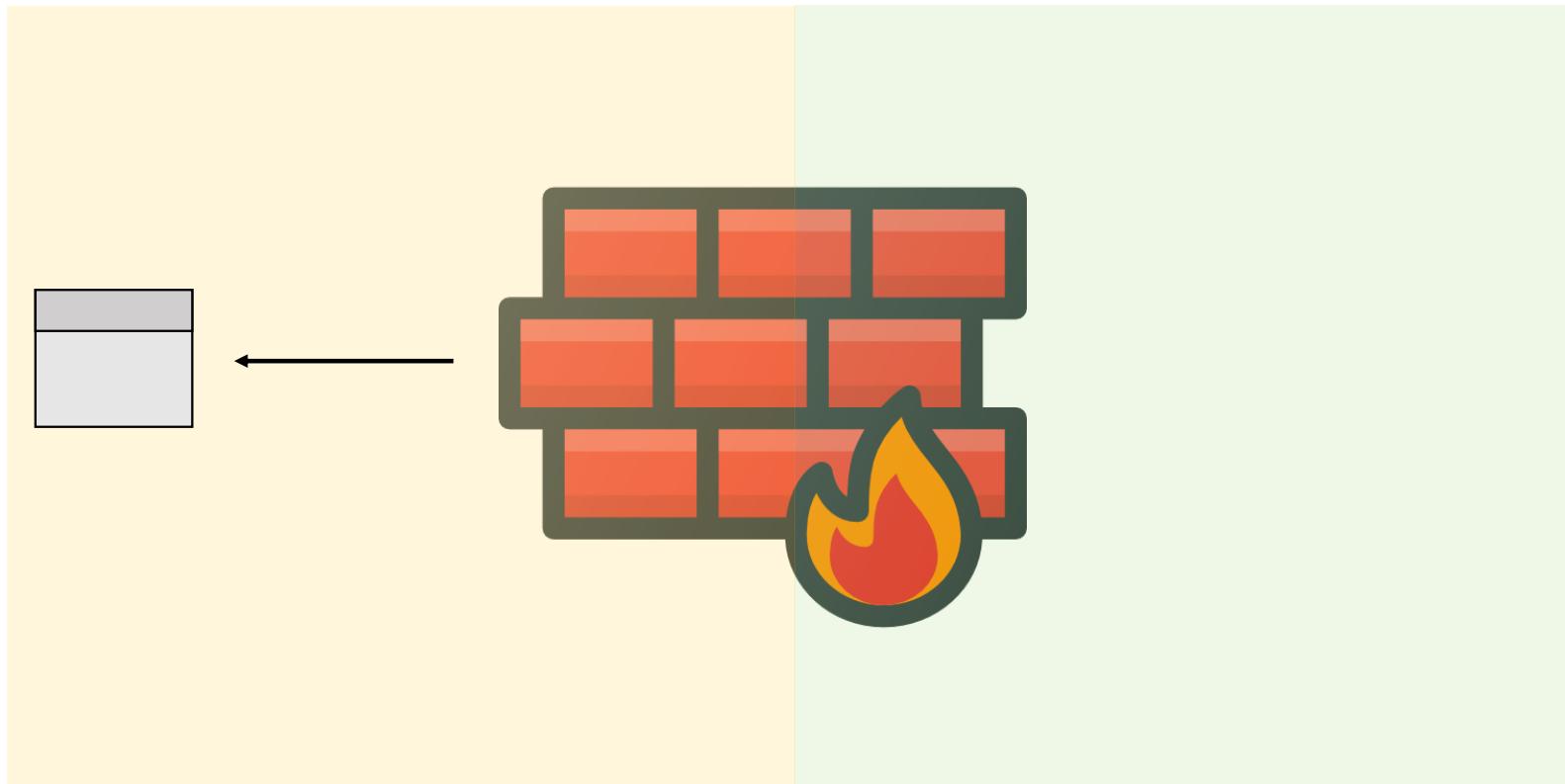
Firewalls: Packet Filters



Three possible actions:

Allow: datagram can pass.

Firewalls: Packet Filters

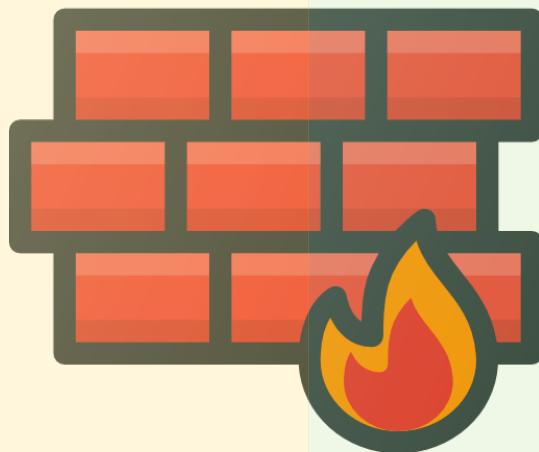


Three possible actions:

Allow: datagram can pass.

Deny: the datagram is blocked or sent back (it happens very rarely).

Firewalls: Packet Filters



Extern

Intern

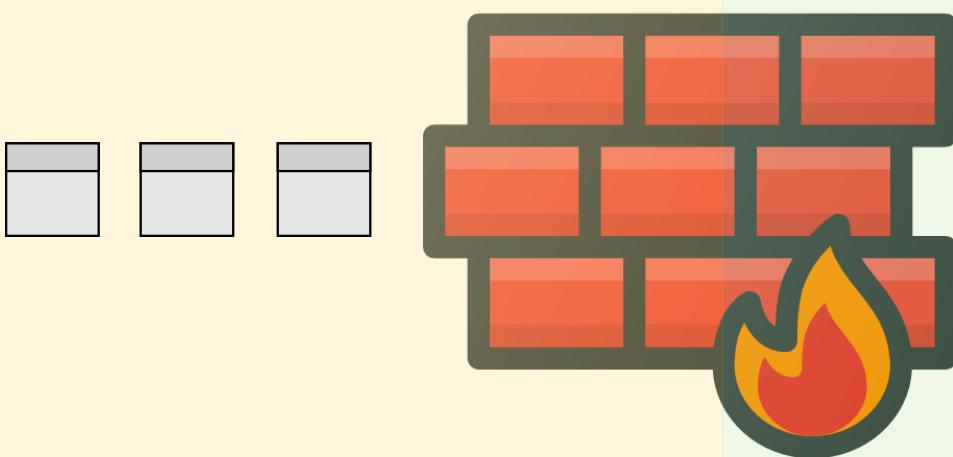
Three possible actions:

Allow: datagram can pass.

Deny: the datagram is blocked or sent back (it happens very rarely).

Drop: the datagram is discarded.

Firewalls: Packet Filters



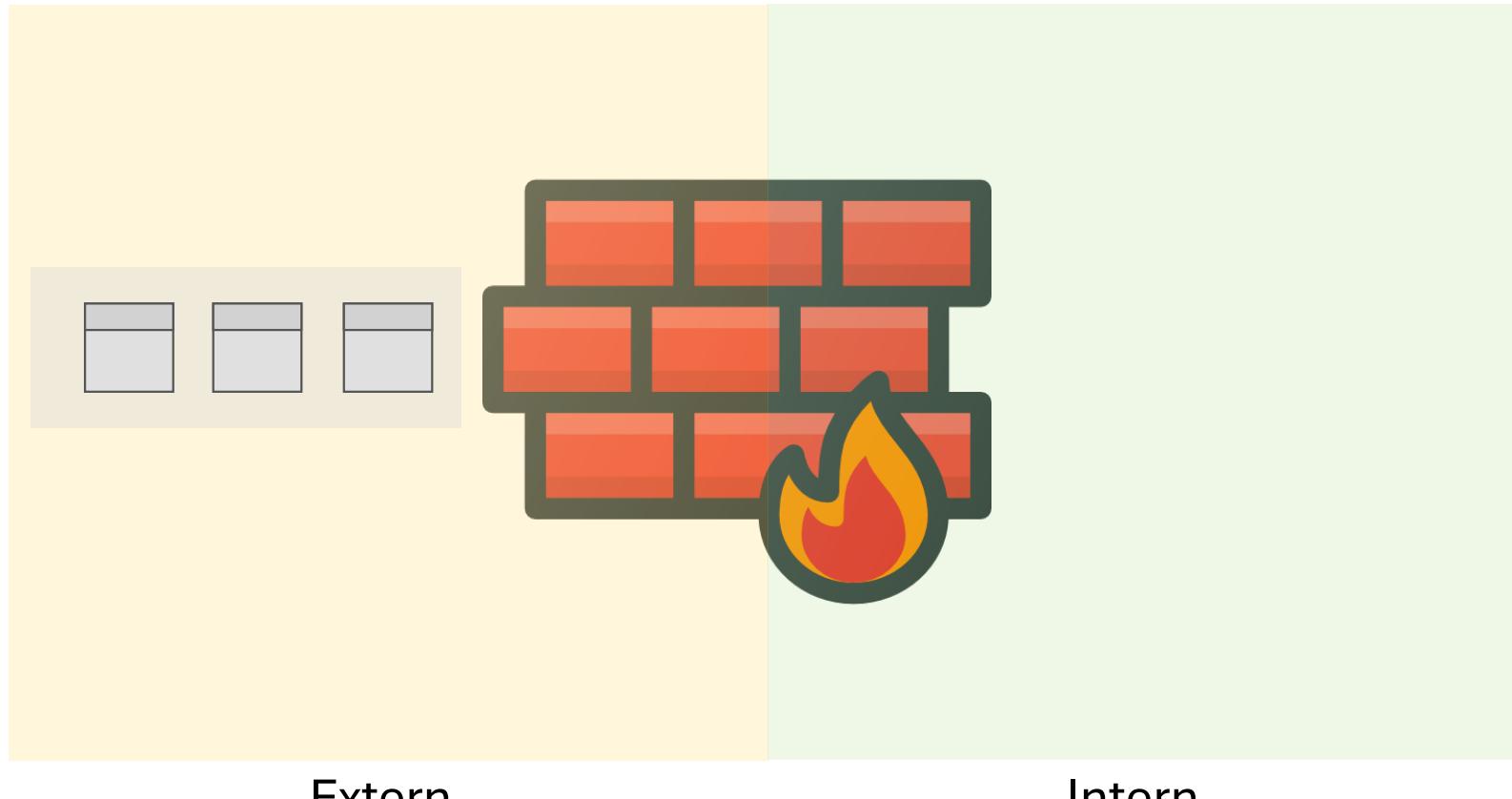
Firewalls can apply rules by considering traffic in two different manners.

Stateless packet filter: each packet is processed in a “standalone” manner.

Decisions are done on a per-packet basis.

This approach is simple, thus it requires less memory and can be fast.

Firewalls: Packet Filters



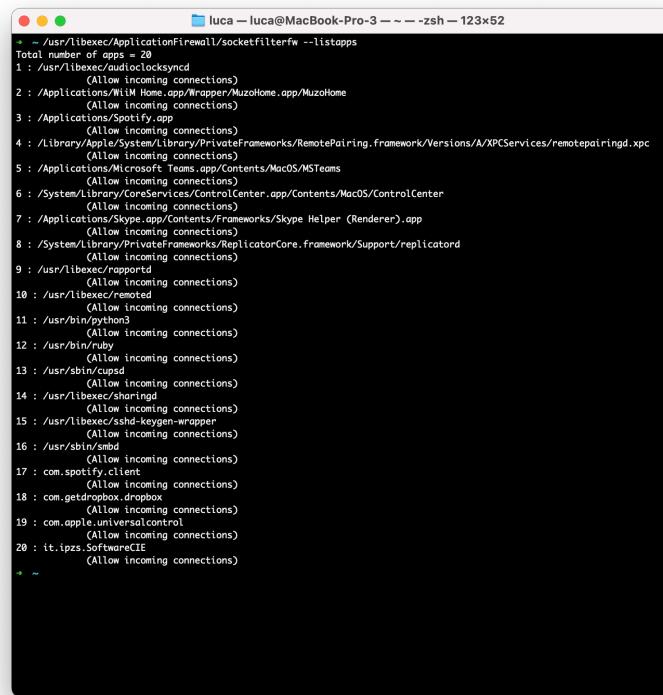
Firewalls can apply rules by considering traffic in two different manners.

Statefull packet filter:
maintains a context about active sessions and exploits “state information” to process packets.

This requires to create a table of conversations (e.g., outbound TCP connections), which needs resources.

Firewalls: Packet Filters

- Examples of filtering rules
 - macOS: */usr/libexec/ApplicationFirewall/socketfilterfw --listapps*



```
luca - luca@MacBook-Pro-3 ~ - zsh - 123x52
~ - /usr/libexec/ApplicationFirewall/socketfilterfw --listapps
Total number of apps = 20
1 : /usr/libexec/audioclocksyncd
    (Allow incoming connections)
2 : /Applications/WiFi Home.app/Wrapper/MuzoHome.app/MuzoHome
    (Allow incoming connectionS)
3 : /Applications/Geoify.app
    (Allow incoming connections)
4 : /Library/ApplicationSupport/PrivateFrameworks/RemotePairing.framework/Versions/A/XPCServices/remotepairingd.xpc
5 : /Applications/Microsoft Teams.app/Contents/MacOS/MSTeams
    (Allow incoming connections)
6 : /System/Library/CoreServices/ControlCenter.app/Contents/MacOS/ControlCenter
    (Allow incoming connections)
7 : /Applications/Skype.app/Contents/Frameworks/Skype Helper (Renderer).app
    (Allow incoming connections)
8 : /System/Library/PrivateFrameworks/ReplicatorCore.framework/Support/replicatord
    (Allow incoming connectionS)
9 : /usr/libexec/reportd
    (Allow incoming connections)
10 : /usr/libexec/remoted
    (Allow incoming connections)
11 : /usr/bin/python3
    (Allow incoming connections)
12 : /usr/bin/ruby
    (Allow incoming connections)
13 : /usr/sbin/cupsd
    (Allow incoming connectionS)
14 : /usr/libexec/sharingd
    (Allow incoming connections)
15 : /usr/libexec/sshd-keygen-wrapper
    (Allow incoming connections)
16 : /usr/sbin/smbd
    (Allow incoming connections)
17 : com.spotify.client
    (Allow incoming connections)
18 : com.getdropbox.dropbox
    (Allow incoming connections)
19 : com.apple.versuscontrol
    (Allow incoming connections)
20 : it.ipzSoftwareCIE
    (Allow incoming connections)
```

Firewalls: Packet Filters

- Examples of filtering rules
 - Linux: inserting a rule in *iptables*

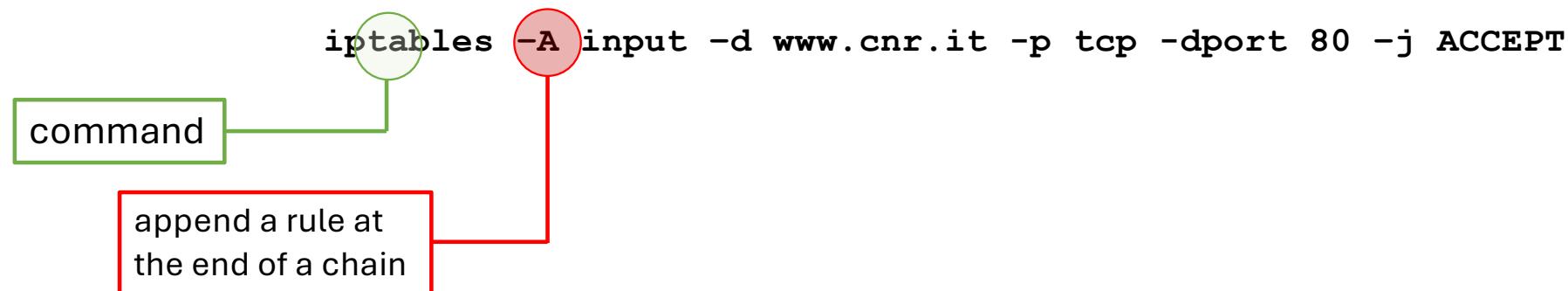
```
iptables -A input -d www.cnr.it -p tcp -dport 80 -j ACCEPT
```

command



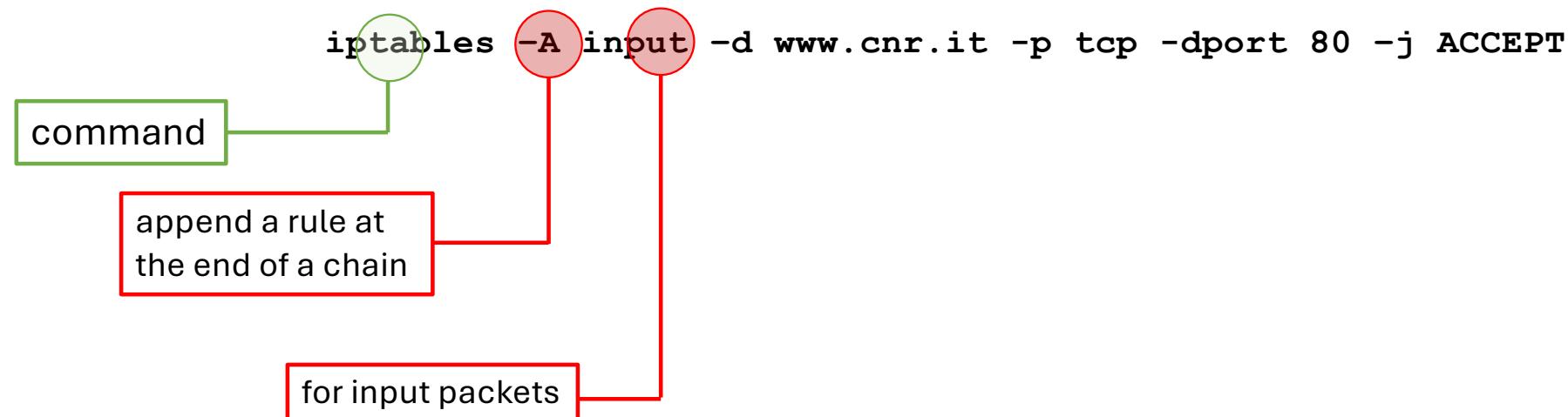
Firewalls: Packet Filters

- Examples of filtering rules
 - Linux: inserting a rule in *iptables*



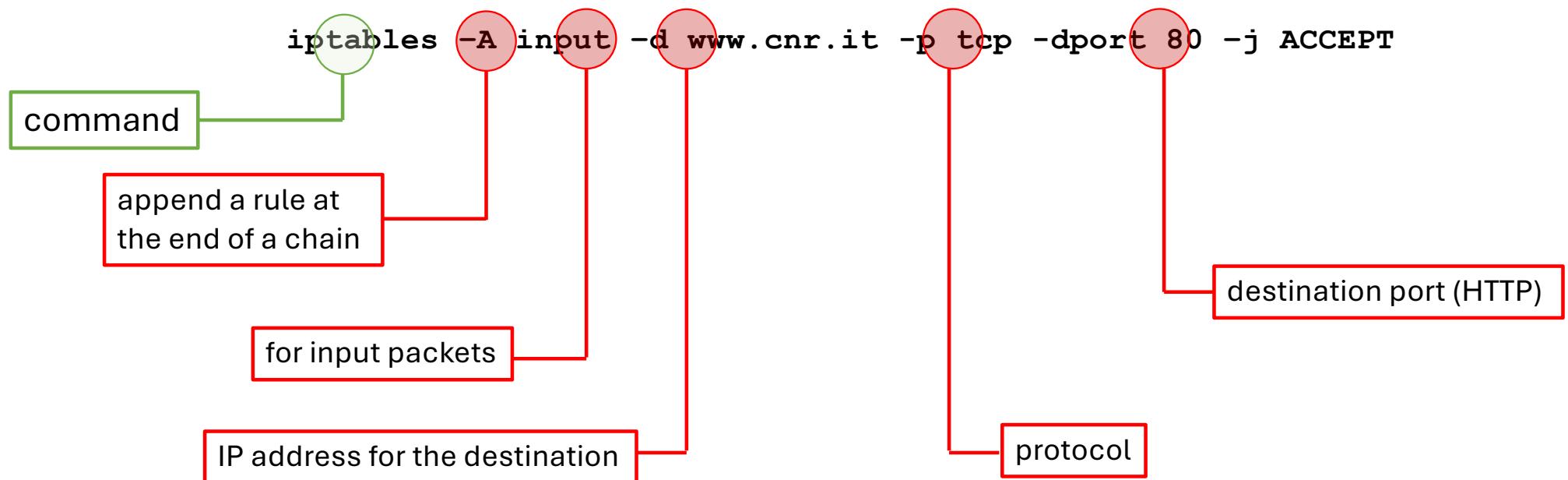
Firewalls: Packet Filters

- Examples of filtering rules
 - Linux: inserting a rule in *iptables*



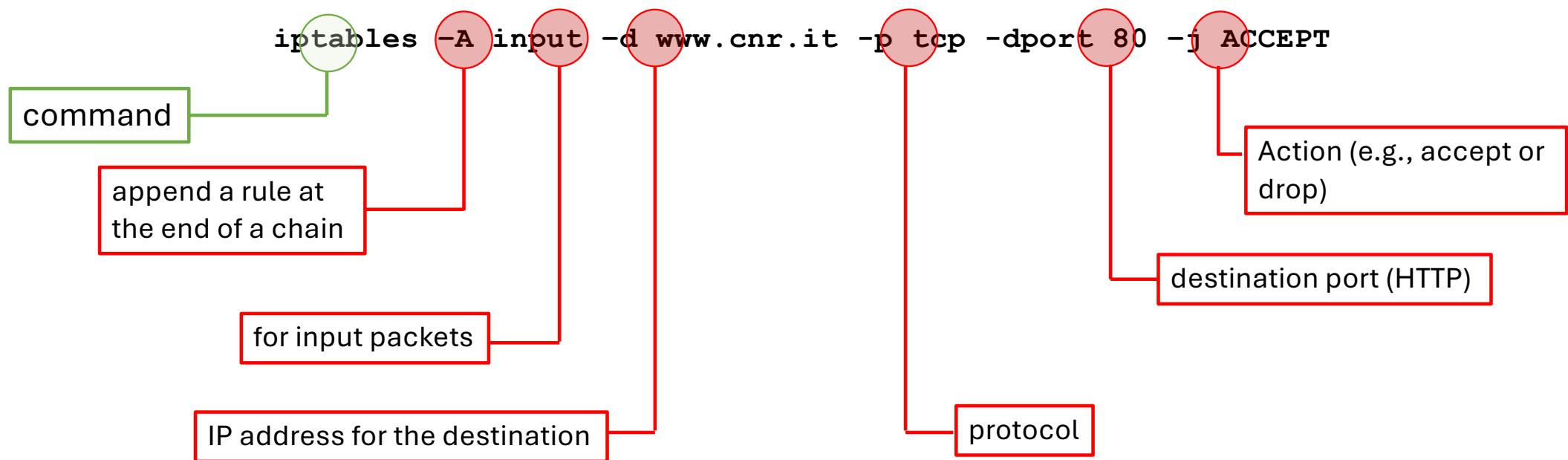
Firewalls: Packet Filters

- Examples of filtering rules
 - Linux: inserting a rule in *iptables*



Firewalls: Packet Filters

- Examples of filtering rules
 - Linux: inserting a rule in *iptables*

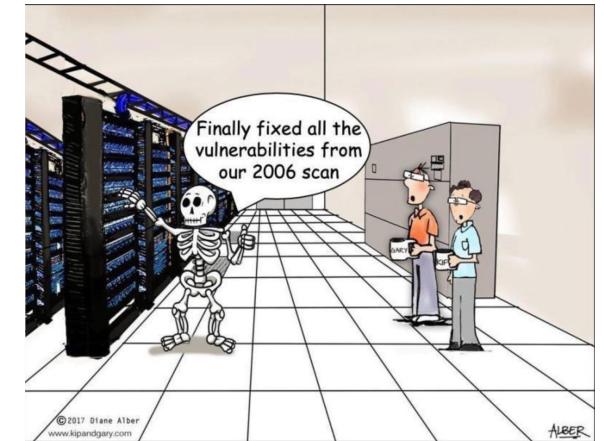


(Network) Vulnerability Scanning

- Vulnerability scanning exploits various tools in an automated manner for finding vulnerabilities in:
 - software
 - systems
 - networks.
- Vulnerability scanning has a two-fold goal:
 - **anticipate** threat actors
 - find possible **entry points** to **launch attacks**.
- The typical lifecycle of a vulnerability scanning process is:
 - **scan**: probe devices, networks and applications to collect information
 - **analysis**: check responses to spot possible weakness or vulnerabilities
 - **report**: provide a report of what has been found.

(Network) Vulnerability Scanning

- Typical information searched via vulnerability scanning are:
 - **misconfigurations:** systems not correctly configured (e.g., default configurations)
 - **outdated software:** hosts/devices running old software version with known unpatched vulnerabilities
 - **missing patches:** software or OSes that without up-to-date patches and fixes (who said IoT?)
 - **weak/default credentials:** entities that can be accessed via unsecure *login+password* entries
 - **open ports:** network ports and services that have been exposed (mostly, in an unnecessary manner).



Source: Diane Alber (<https://x.com/kipandgary>)

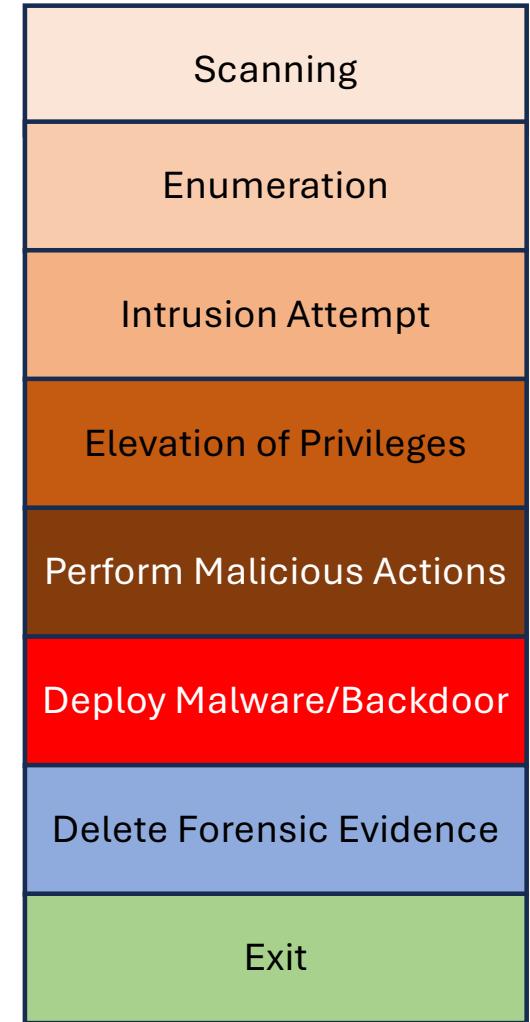
(Network) Vulnerability Scanning

- Several types of vulnerability scanners:
 - there is not a unique taxonomy
 - rise of the cyberscanning concept.
- Most popular scanners:
 - **network and port scanners**: list hosts and network services that are running and reachable, e.g., *nmap*
 - **network vulnerability scanners**: list possible vulnerabilities associated to detected running application, e.g., Nessus, SAINT, OpenVAS
 - **web application security scanners**(^{*}): crawl and test web applications, e.g., Acunetix, Panoptic Scans, Threatspy
 - **database security scanners**: scan databases management systems for possible issues, e.g., Scuba.

(*) OWASP Vulnerability Scanning Tools https://owasp.org/www-community/Vulnerability_Scanning_Tools

Network Vulnerability Scanning

- Cyber scanning or network reconnaissance is becoming popular since attacker can remotely:
 - locate target(s)
 - exploit vulnerable a system
 - design botnets, deploy malware, start phishing and spam campaigns, etc.
- Two main types of scanning approaches:
 - **active scanning:** network services and host/devices are identified by transmitting suitable packets or network traffic and monitoring the response
 - **passive scanning:** network services and host/devices are identified by observing the traffic in a specific collection point.



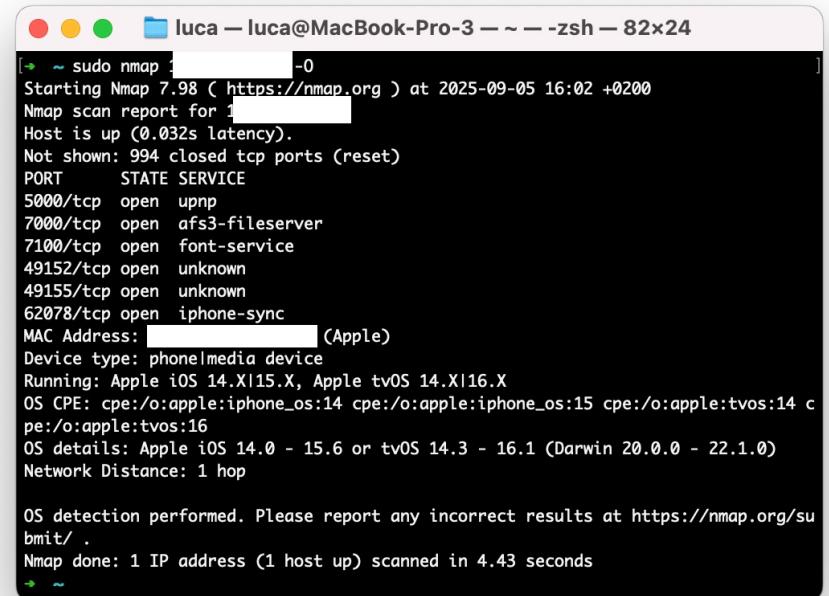
Typical cyber attack leveraging vulnerability scanning

Network Scanning

- Network scanning consists in probing a network to identify:
 - active devices
 - running services
 - open ports
 - vulnerabilities.
- Network scanning tools are typically used by:
 - administrators to check the security of their networks
 - attackers to identify running services on a host for exploitation.
- Most popular scanning tools:
 - standard utilities, e.g., *ping/ping6* and *traceroute/traceroute6*
 - ad-hoc “suites”, e.g., *nmap*

Network Scanning

- In general, a network scanning campaign is devoted to perform:
 - **host discovery**: identifying hosts on a network, e.g., nodes which respond to ping messages
 - **port scanning**: enumerating the open ports of target host(s)
 - **service version**: determining the type of an application and its version
 - **OS fingerprinting**: finding the type/version of the operating system and basic hardware characteristics of the device.

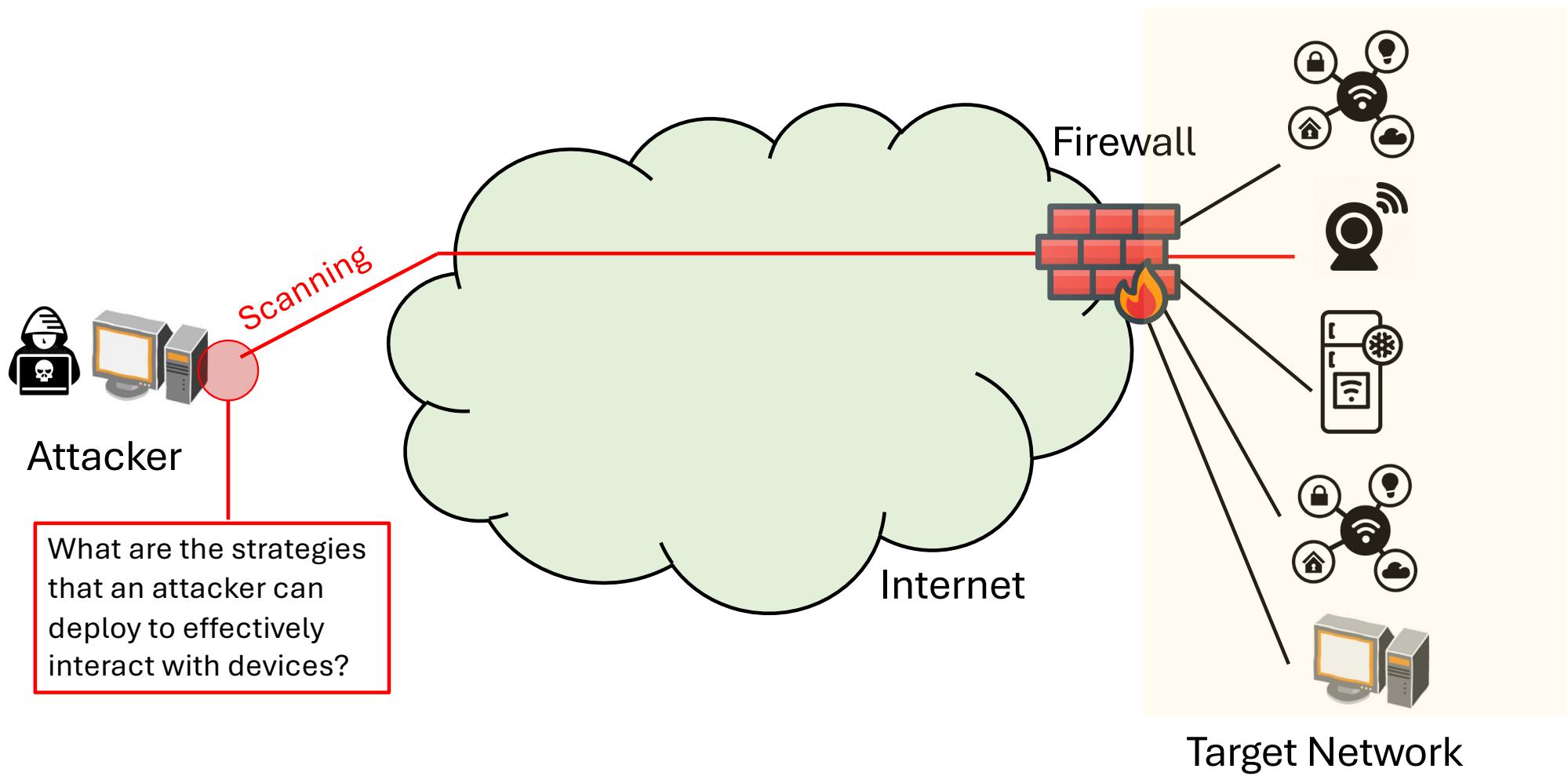


```
[~] ~ sudo nmap [REDACTED] -O
Starting Nmap 7.98 ( https://nmap.org ) at 2025-09-05 16:02 +0200
Nmap scan report for [REDACTED]
Host is up (0.032s latency).
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE
5000/tcp   open  upnp
7000/tcp   open  afs3-filesystem
7100/tcp   open  font-service
49152/tcp  open  unknown
49155/tcp  open  unknown
62078/tcp  open  iphone-sync
MAC Address: [REDACTED] (Apple)
Device type: phone|media device
Running: Apple iOS 14.X|15.X, Apple tvOS 14.X|16.X
OS CPE: cpe:/o:apple:iphone_os:14 cpe:/o:apple:iphone_os:15 cpe:/o:apple:tvos:14 cpe:/o:apple:tvos:16
OS details: Apple iOS 14.0 - 15.6 or tvOS 14.3 - 16.1 (Darwin 20.0.0 - 22.1.0)
Network Distance: 1 hop

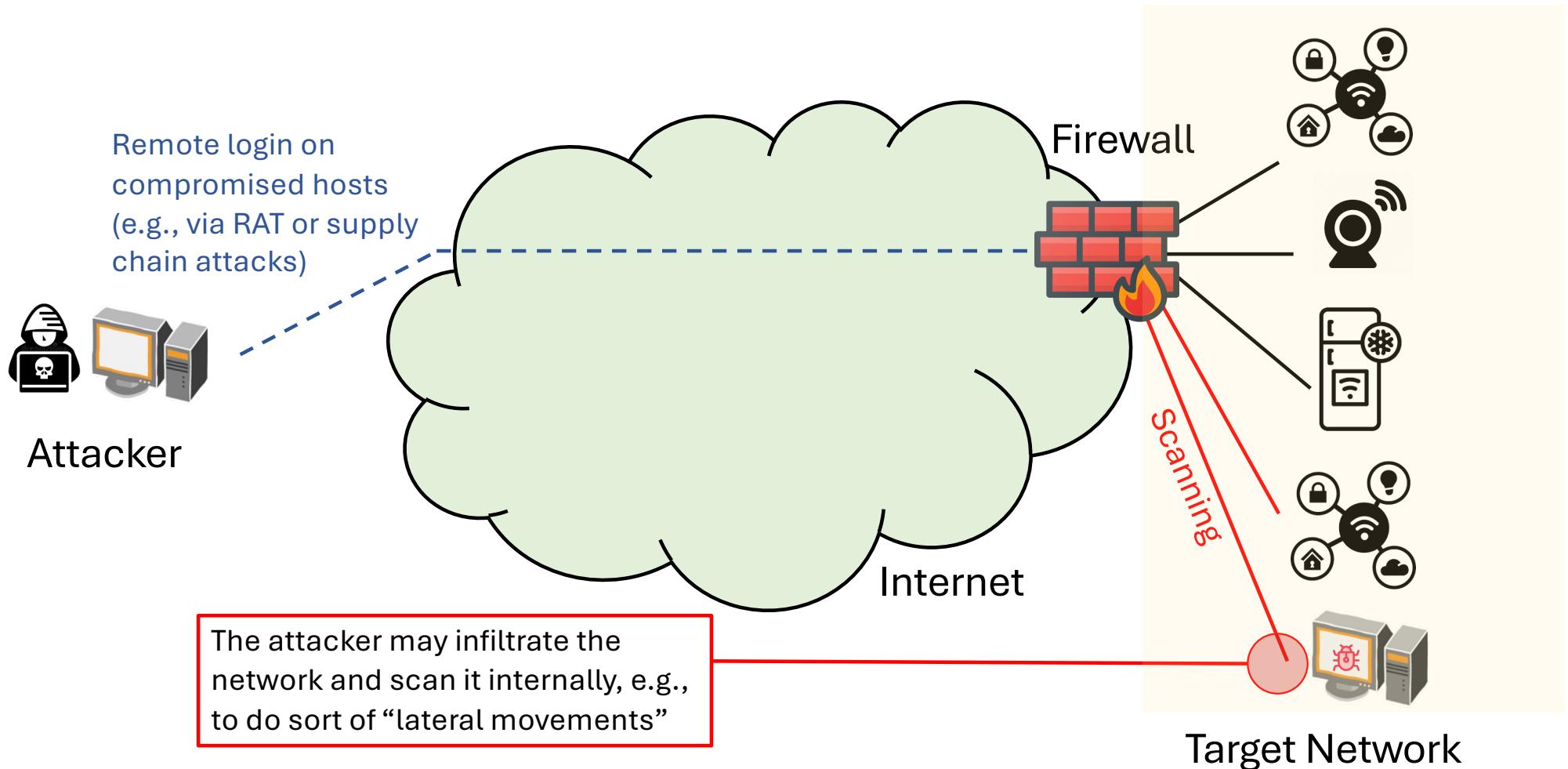
OS detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 4.43 seconds
```

Example of OS fingerprinting

Scanning Scenario: Remote Campaign

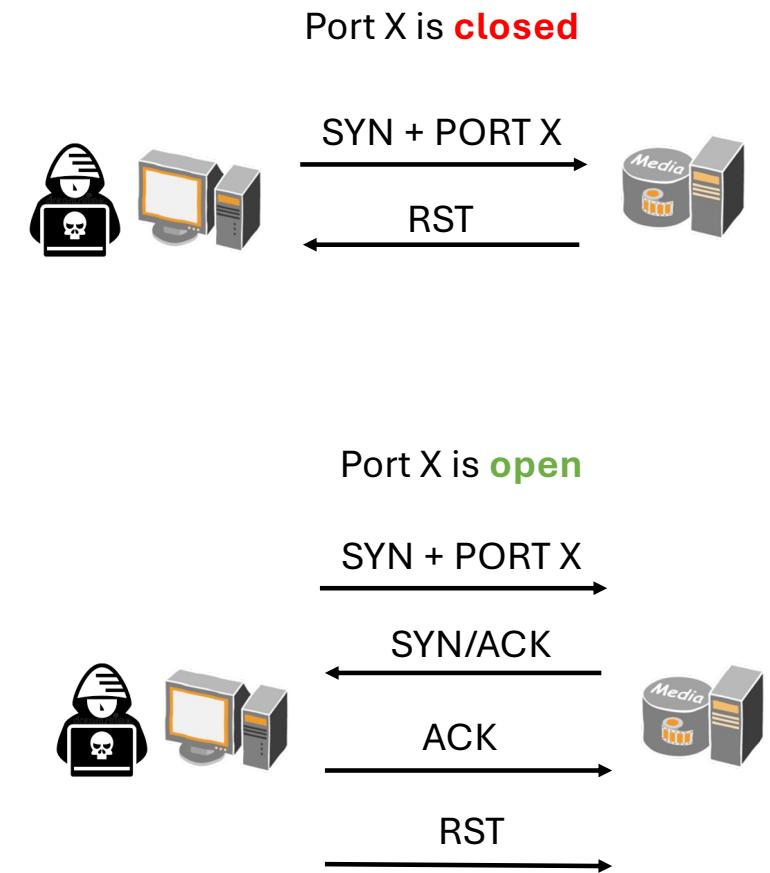


Scanning Scenario: “Local” Campaign



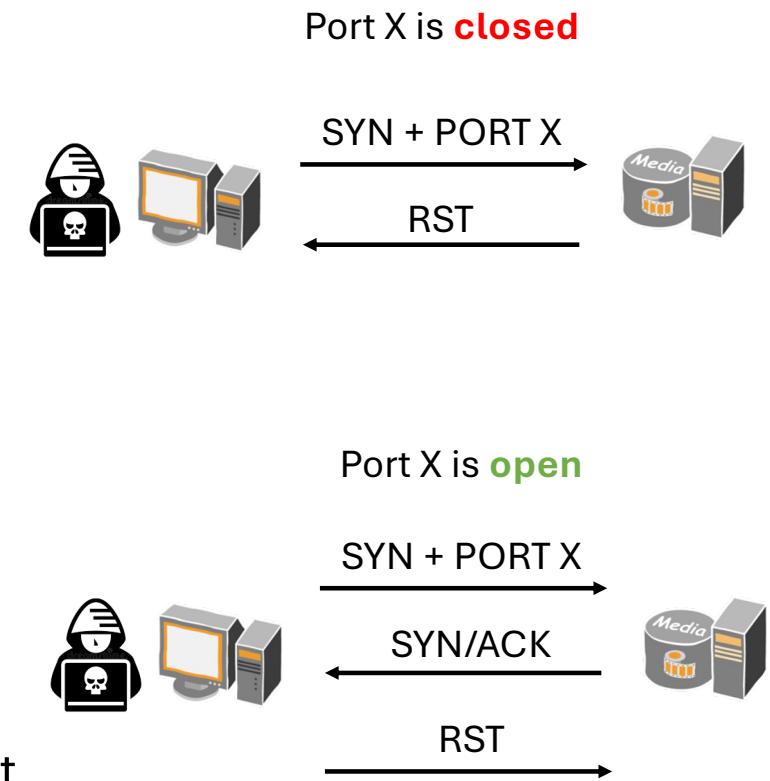
Scanning Strategies

- TCP Scan (also known as open scan or vanilla scan):
 - is the simplest port scanning approach
 - uses the standard `connect()` system call.
- Procedure:
 - if the port of the target is open, the OS completes the TCP three-way handshake and the scanner closes the connection
 - otherwise, an error is returned.
- Pros and cons:
 - the user does not require special privileges
 - prevents a fine-grained control
 - easy to implement, easy to spot (**is life...**)



Scanning Strategies

- SYN Scan (also known as half-open scan):
 - the scanner generates raw IP packets rather than using the network functions of the OS.
- Procedure:
 - a TCP SYN segment is sent and the scanner monitors for a TCP SYN-ACK segment indicating an open port
 - the scanner responds with a TCP RST segment and closes the connection before the three-way handshake is completed
 - a full TCP connection is never opened.
- Pros and cons:
 - since the connection is not established, the target does not log the attempt
 - it requires high privileges, e.g., it cannot be implemented in compromised nodes without escalating as root.



Scanning Strategies

- ACK Scan:

- not aimed at determining whether a port is open but to map firewall rules
- the scanner generates raw IP packets rather than using the network functions of the OS.

- Procedure:

- a TCP ACK segment is sent towards the target port/host

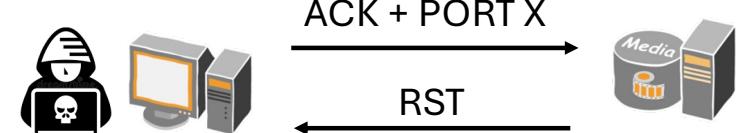
- Pros and cons:

- scanning a single port is almost “invisible”
- storms of unsolicited ACKs would reveal the trick.

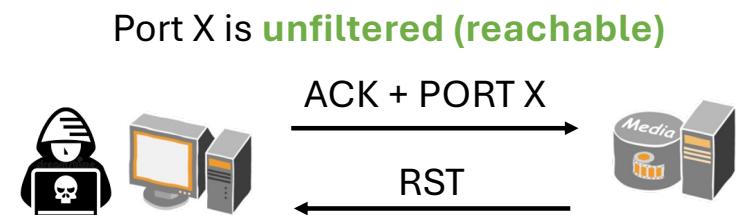
Port X is **filtered (non-reachable)**



Port X is **unfiltered (reachable)**



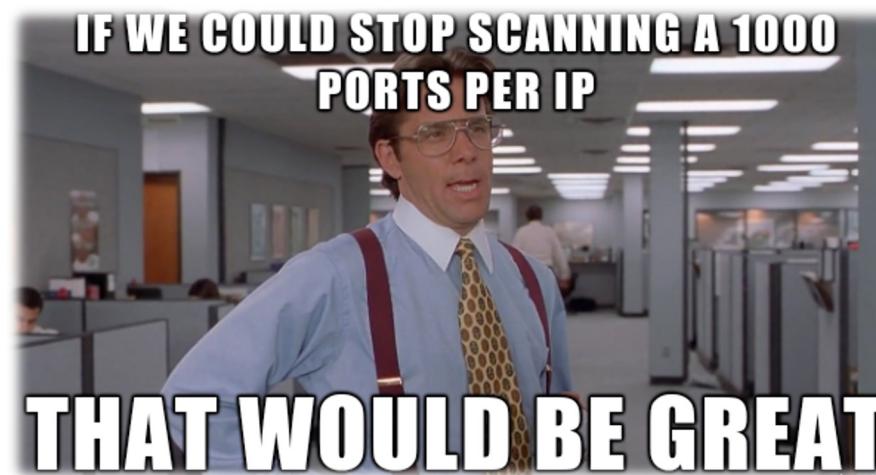
Port X is **closed**



Port X is **open**

Other Scanning Strategies

- UDP scanning:
 - if an UDP packet is sent to a closed port, some OSes respond with an ICMP port unreachable message
 - application-specific UDP packets are sent hoping to generate an application layer response.



r/networkingmemes

Network Scanning: Nmap

- Nmap is a free and network scanner:
 - many functionalities for probing hosts , fingerprint nodes and discover services
 - <https://nmap.org>
 - on macOS: *brew install nmap*
 - **give it a try!**
- The tool:
 - is great for security auditing
 - can monitor host and services
 - cab check various network behaviors
 - is great for reconnaissance



```
luca — luca@dhcpclient112 — ~ -- zsh — 102x23
$ sudo nmap -sA 150.145.████████
[Starting Nmap 7.98 ( https://nmap.org ) at 2025-09-04 18:52 +0200]
Nmap scan report for nameserver.ge.imati.cnr.it (150.145.████████)
[Host is up (0.0011s latency).
Not shown: 989 filtered tcp ports (no-response), 8 filtered tcp ports (admin-prohibited)
PORT      STATE      SERVICE
22/tcp    unfiltered ssh
53/tcp    unfiltered domain
9090/tcp  unfiltered zeus-admin

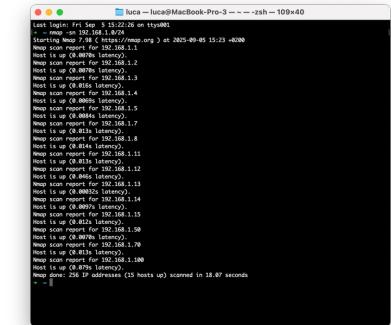
Nmap done: 1 IP address (1 host up) scanned in 5.03 seconds
$ sudo nmap -sS 150.145.████████
[Starting Nmap 7.98 ( https://nmap.org ) at 2025-09-04 18:53 +0200]
Nmap scan report for nameserver.ge.imati.cnr.it (150.145.████████)
Host is up (0.0013s latency).
Not shown: 987 filtered tcp ports (no-response), 10 filtered tcp ports (admin-prohibited)
PORT      STATE      SERVICE
22/tcp    open       ssh
53/tcp    open       domain
9090/tcp  closed    zeus-admin

Nmap done: 1 IP address (1 host up) scanned in 6.39 seconds
```

nmap doing a SYN Scan (-sS) and an ACK Scan (-sA)

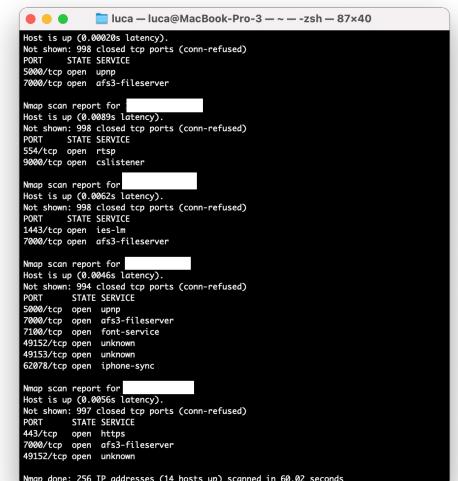
Example: Scanning with Nmap

- Scan for active hosts on a LAN:
 - *nmap -sn target_network/subnet_CIDR_notation*
 - **explanation:** perform a ping scan, i.e., the tool annotates hosts responding to probes. Ports are not scanned.
- Scan for active hosts on a LAN and open ports:
 - *nmap target_network/subnet_CIDR_notation*
 - **explanation:** the tool searches for active hosts and when found ports are scanned.



```
Last login: Fri Sep 5 15:32:26 on ttys001
...
= nmap -sn 192.168.1.0/24
Nmap scan report for 192.168.1.1
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.2
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.3
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.4
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.5
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.6
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.7
Host is up (0.010ms latency).
Nmap scan report for 192.168.1.8
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.9
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.10
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.11
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.12
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.13
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.14
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.15
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.16
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.17
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.18
Host is up (0.000ms latency).
Nmap scan report for 192.168.1.19
Host is up (0.000ms latency).
Nmap done: 256 IP addresses (15 hosts up) scanned in 18.87 seconds
```

Active hosts



```
Host is up (0.00002s latency).
Host shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
5000/tcp  open  upnp
7000/tcp  open  afs3-fileserver

Nmap scan report for [REDACTED]
Host is up (0.00002s latency).
Host shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
554/tcp   open  rtsp
9000/tcp  open  cslistener

Nmap scan report for [REDACTED]
Host is up (0.00002s latency).
Host shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
10000/tcp open  netmtrace
1443/tcp  open  iec-tsc
7000/tcp  open  afs3-fileserver

Nmap scan report for [REDACTED]
Host is up (0.00002s latency).
Host shown: 994 closed tcp ports (conn-refused)
PORT      STATE SERVICE
5000/tcp  open  upnp
7000/tcp  open  afs3-fileserver
7000/tcp  open  file-service
49152/tcp open  unknown
49153/tcp open  unknown
49153/tcp open  unknown
62078/tcp open  iphone-sync

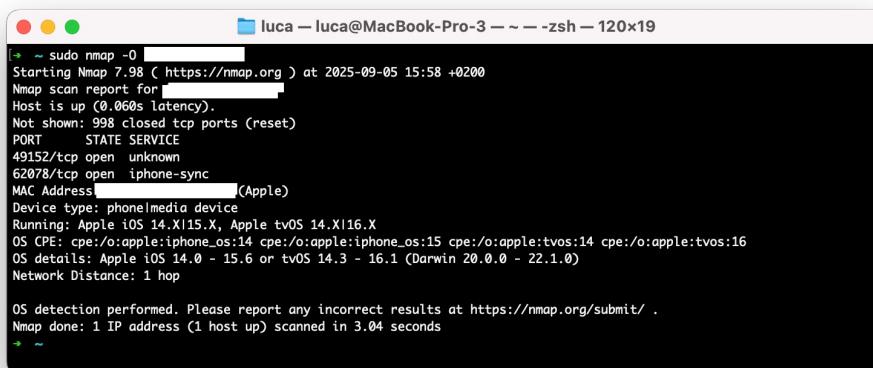
Nmap scan report for [REDACTED]
Host is up (0.00002s latency).
Host shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE
443/tcp   open  https
7000/tcp  open  afs3-fileserver
49152/tcp open  unknown

Nmap done: 256 IP addresses (14 hosts up) scanned in 60.02 seconds
```

Active hosts and opened ports

Example: Scanning with Nmap

- Perform OS fingerprint:
 - *nmap -O host*
 - **requires sudo!**
 - **explanation:** the OS is “guessed” by matching responses from probes against a DB. Typical traits analyzed are: TCP Options, Window Sizes, initial value for the TTL, order of Flags, reaction against specific ICMP stimuli.



The screenshot shows a terminal window titled "luca — luca@MacBook-Pro-3 — ~ — zsh — 120x19". The command entered is "sudo nmap -O [REDACTED]". The output shows the following information:

```
Starting Nmap 7.98 ( https://nmap.org ) at 2025-09-05 15:58 +0200
Nmap scan report for [REDACTED]
Host is up (0.060s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
49152/tcp open  unknown
62978/tcp open  iphone-sync
MAC Address: [REDACTED] (Apple)
Device type: phone|media device
Running: Apple iOS 14.X|15.X, Apple tvOS 14.X|16.X
OS CPE: cpe:/o:apple:iphone_os:14 cpe:/o:apple:iphone_os:15 cpe:/o:apple:tvos:14 cpe:/o:apple:tvos:16
OS details: Apple iOS 14.0 - 15.6 or tvOS 14.3 - 16.1 (Darwin 20.0.0 - 22.1.0)
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.04 seconds
```

Correct guessing of my iPhone

Example: Scanning with Nmap

- Perform OS fingerprint:
 - *nmap -O host*
 - **requires sudo!**
 - **explanation:** the OS is “guessed” by matching against a DB. Typical traits analyzed are: TCP Connect time, value for the TTL, order of Flags, reaction against SYN, etc.

```
luca — luca@MacBook-Pro-3 — ~ — zsh — 120x19
[~] ~$ sudo nmap -O [REDACTED]
Starting Nmap 7.98 ( https://nmap.org ) at 2025-09-05 15:58 +0200
Nmap scan report for [REDACTED]
Host is up (0.060s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
49152/tcp open  unknown
62978/tcp open  iphone-sync
MAC Address: [REDACTED] (Apple)
Device type: phone|media device
Running: Apple iOS 14.X|15.X, Apple tvOS 14.X|16.X
OS CPE: cpe:/o:apple:iphone_os:14 cpe:/o:apple:iphone_os:15 cpe:/o:apple:tvos:14 cpe:/o:apple:tvos:16
OS details: Apple iOS 14.0 - 15.6 or tvOS 14.3 - 16.1 (Darwin 20.0.0 - 22.1.0)
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.04 seconds
[~]
```

Correct guessing of my iPhone

The screenshot shows a web browser window displaying the NIST National Vulnerability Database. The URL in the address bar is `md.nist.gov`. The main content area shows the details for CVE-2022-42801. The title is "CVE-2022-42801 Detail". Below it, under "MODIFIED", it says: "This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes." The "Description" section states: "A logic issue was addressed with improved checks. This issue is fixed in tvOS 16.1, iOS 15.7.1 and iPadOS 15.7.1, macOS Ventura 13, watchOS 9.1, iOS 16.1 and iPadOS 16, macOS Monterey 12.6.1. An app may be able to execute arbitrary code with kernel privileges." The "Metrics" section includes tabs for CVSS Version 4.0 (selected), CVSS Version 3.x, and CVSS Version 2.0. It shows a base score of 7.8 HIGH and a vector of CVSS:3.1/AV:L/AC:L/PR:N/U:R/S:U/C:H/I:H/A:H. The "References to Advisories, Solutions, and Tools" section contains links to external resources like packetstormsecurity.com and support.apple.com, along with source and tag information for Apple Inc. The entire screenshot is enclosed in a red border.



Policy Enforcing and Hardening

- Developing effective **policies** can help to reduce the attack surface of networked scenarios.
- Some simple ideas:
 - suitable algorithms for selecting passwords and for forcing updates (e.g., to avoid credential stuffing)
 - forcing software updates (e.g., to fix exploitable CVEs)
 - isolation of “aged” devices (e.g., impede to join the network nodes no supported anymore)
 - zoning or segmentation of networks (e.g., to separate IoT/home automation from the rest of the network).
 - ...

Policy Enforcing and Hardening

- Developing effective **policies** can help to reduce the attack surface of networked scenarios.
- What about the diffusion of the Bring Your Own Device (BYOD) paradigm?
 - enforce configuration policies or hardening
 - prevent some third-part applications
 - block rooted devices
 - ...

Create your mobile app policy

These policy settings apply to all supported cloud mobile apps for both Android and iOS (unless indicated otherwise). [Learn more about these settings](#)

Apply this policy to

All users with access to your organization's products
 Specific users

App data protection

Disable sharing, saving or backing up content from the mobile app
 Disable screenshots and screen recording of the mobile app
 Disable cutting or copying content from the mobile app

App access requirements

Block compromised devices
 Block third-party keyboards on iOS
 Require data encryption
 Require biometric authentication or a device passcode
 Require a minimum OS version
 Override any IP allowlists to allow access from Jira and Confluence mobile apps

[Cancel](#) [Create policy](#)

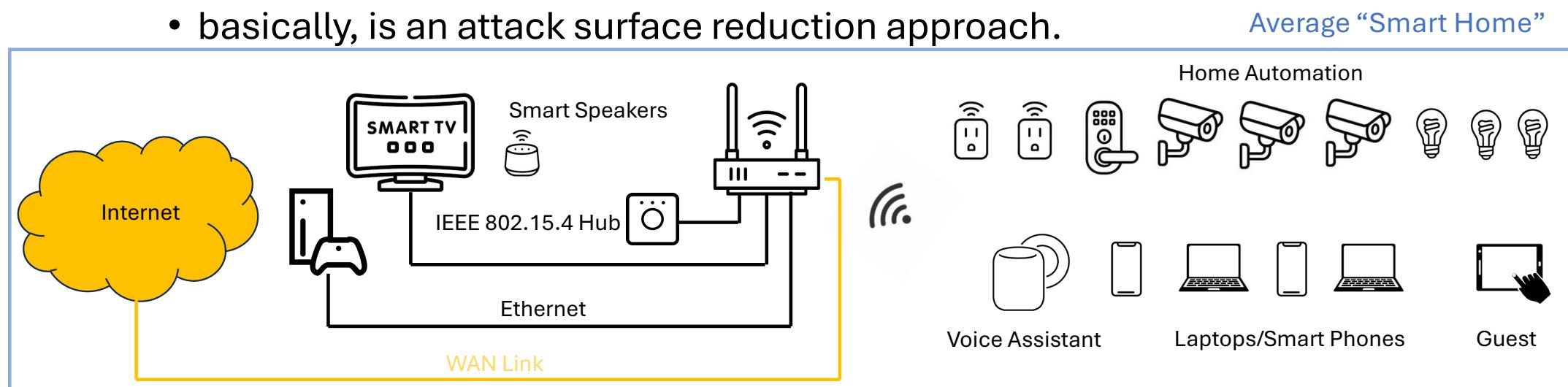
Example: Atlassian: <https://support.atlassian.com/security-and-access-policies/docs/mobile-policy-mam-security-controls-and-supported-apps/>

Network Segmentation

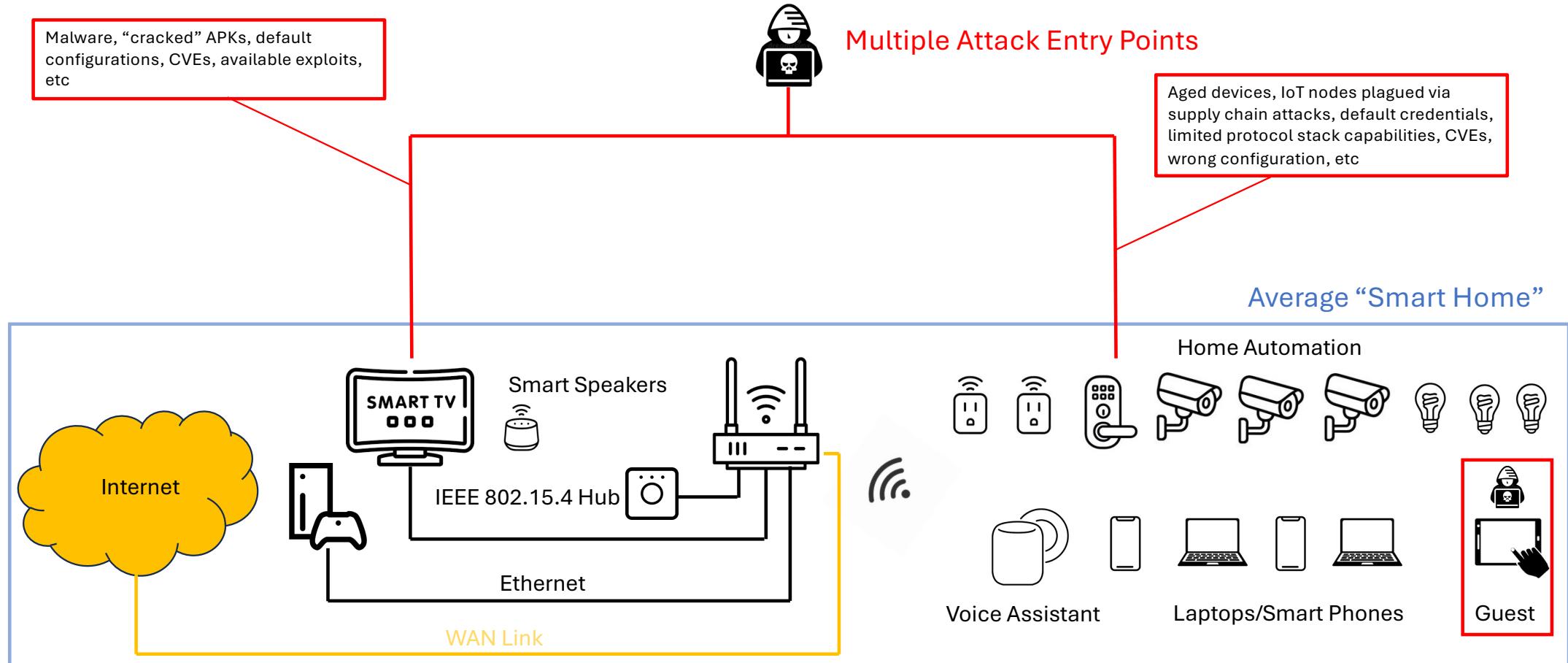
- Network segmentation divides a larger network into smaller and isolated sections:
 - sections are often called zones
 - also known as partitioning
 - isolation improves security and performance
 - basically, is an attack surface reduction approach.

Network Segmentation

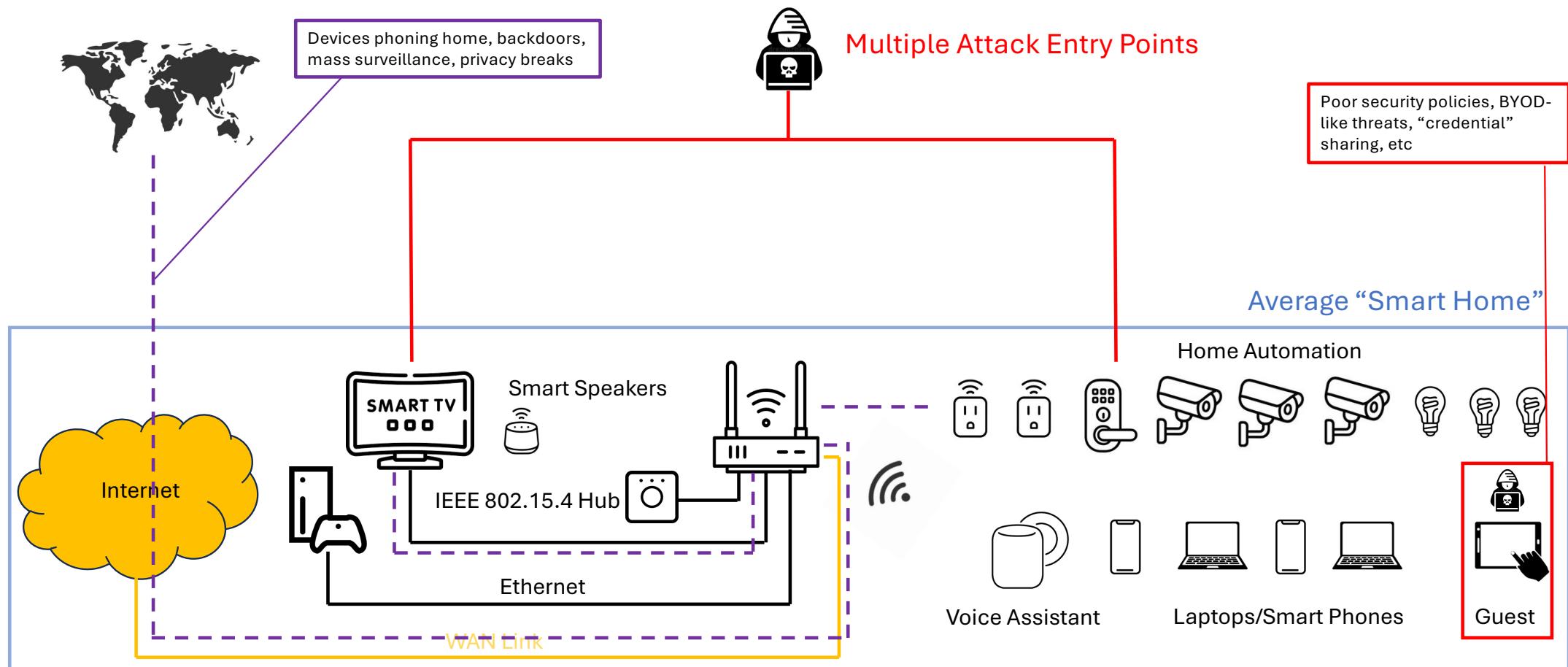
- Network segmentation divides a larger network into smaller and isolated sections:
 - sections are often called zones
 - also known as partitioning
 - isolation improves security and performance
 - basically, is an attack surface reduction approach.



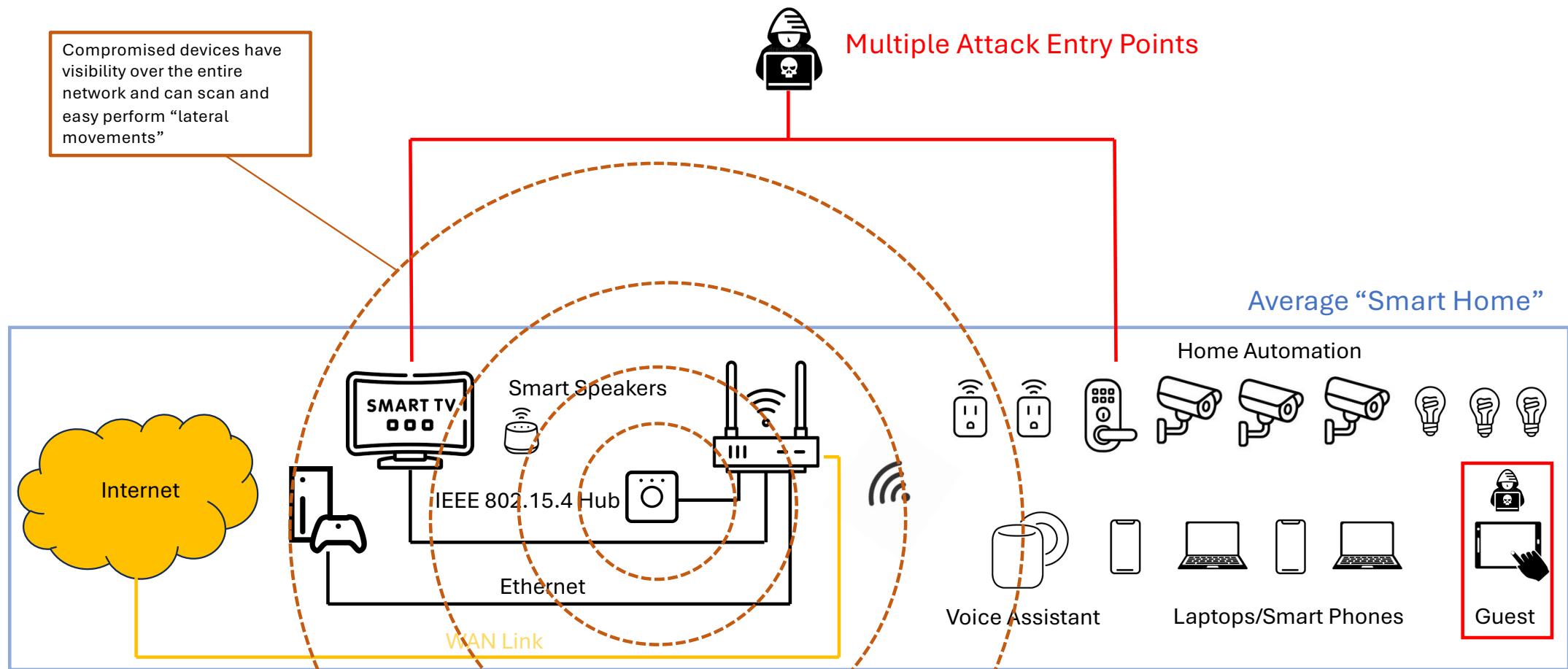
Network Segmentation



Network Segmentation



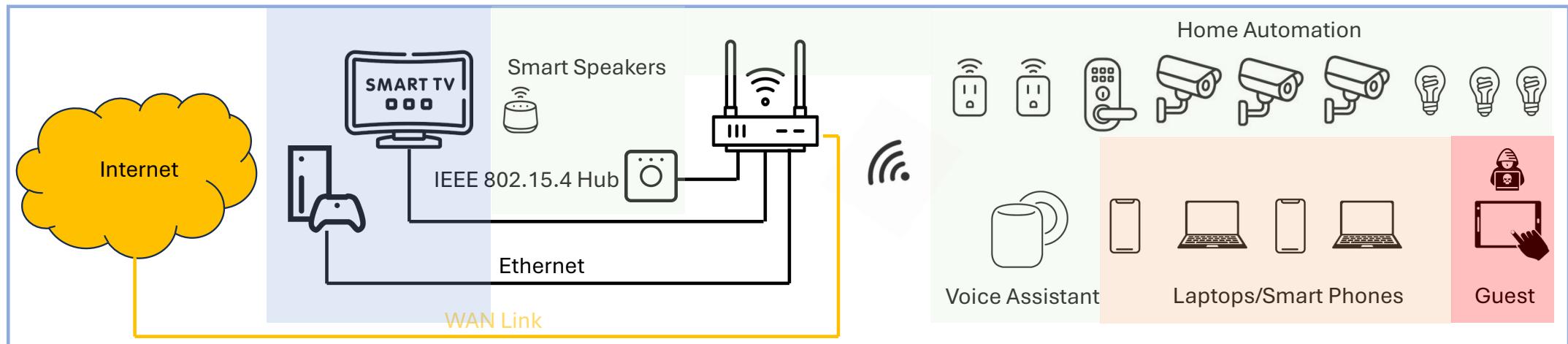
Network Segmentation



Network Segmentation

- VLANs/Subnets
- Firewall Rules
- Ad-hoc SSIDs
- Access Control List
- MAC Filtering

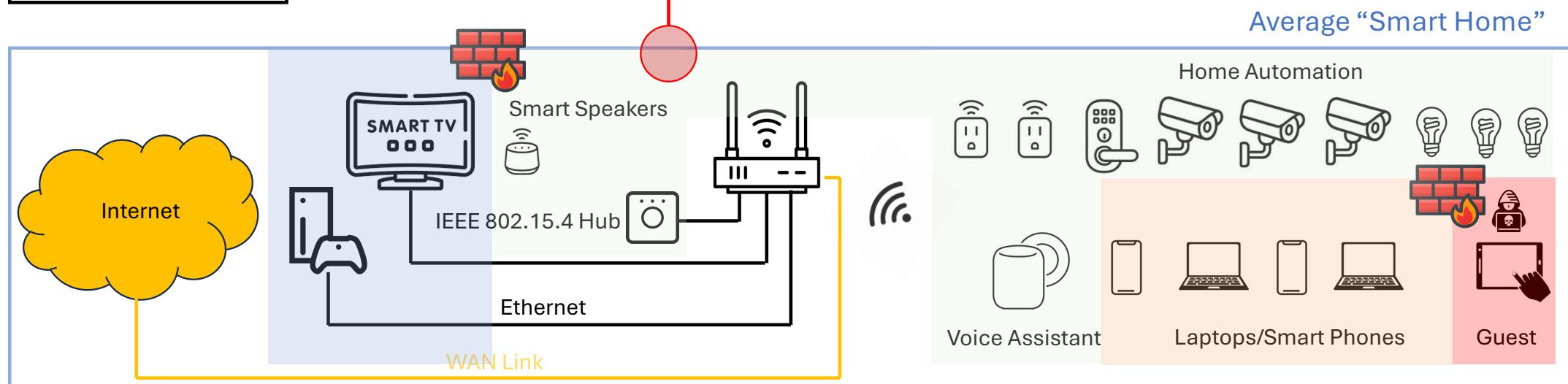
Average “Smart Home”



Network Segmentation

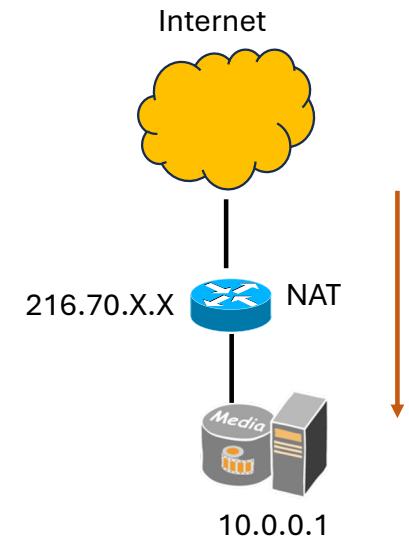
- VLANs/Subnets
- Firewall Rules
- Ad-hoc SSIDs
- Access Control List
- MAC Filtering

Example: separate IoT nodes from other devices and “sandbox” guests or new hardware (e.g., decide if it can be trusted).
Periodically conduct auditing to update configurations



Network Address Translation

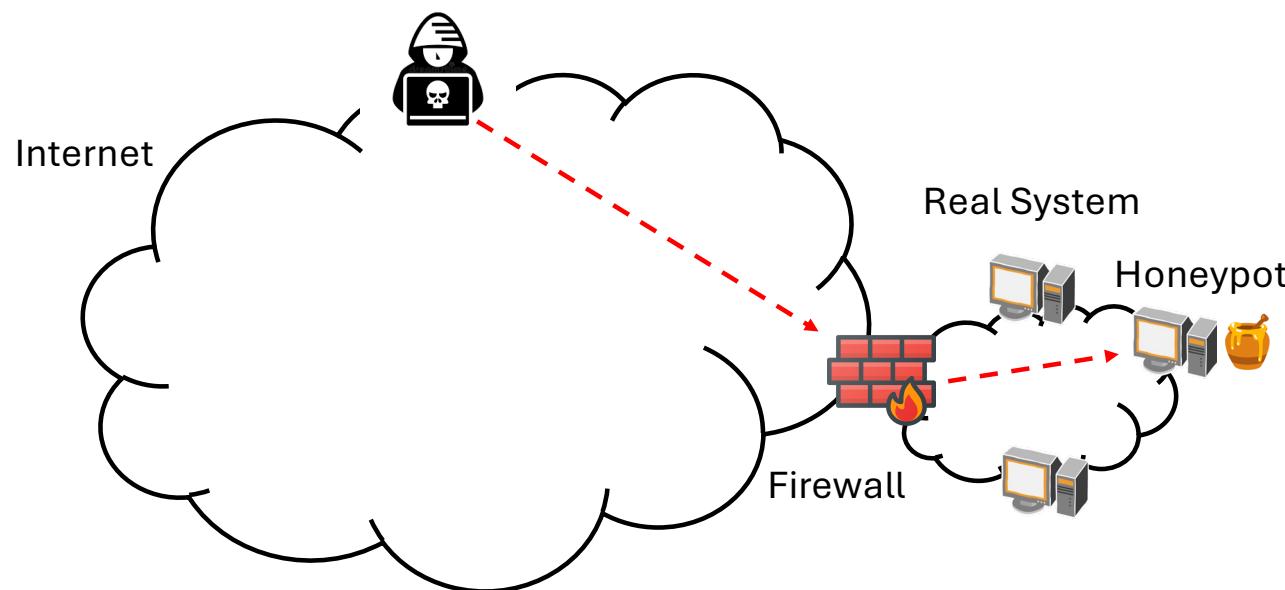
- Network Address Translation (NAT):
 - allows multiple devices on a private network to share the public IP addresses:
 - mainly designed for handling the shortage of IPv4 addresses
 - it can be used to improve security of a network .
- Several types of NATs:
 - out of scope in this course
 - issues in transparency and “**routability**” from the **outside**
 - various traversal techniques.
- Key idea:
 - use a NAT to limit (and control) internal hosts that must be reachable from the Internet.



Contacting 10.0.0.1 requires suitable mappings or in **restricted cone** NATs requires that 10.0.0.1 initiates the conversation.

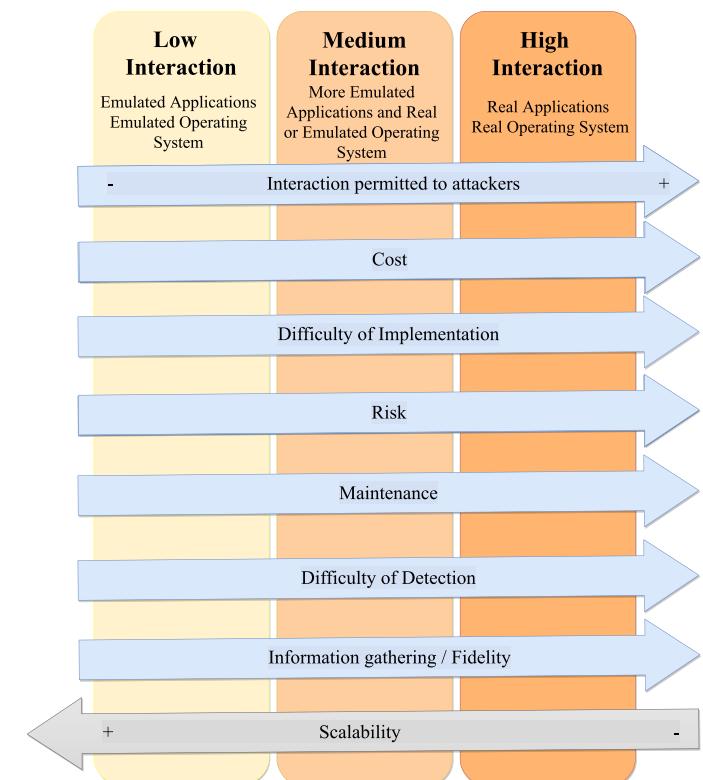
Honeypot

- A honeypot is a tool that serves as a decoy to:
 - attract attackers
 - convince them to think that they have gained access to a real system.



Honeypot

- There are different types of honeypots:
 - **physical honeypots**: real machines or devices
 - **virtual honeypots**: software simulating a host or service.
- A possible, non-exhaustive classification:
 - **high-interaction honeypots**: expose various services with the aim of slowing down the attacker;
 - **low-interaction honeypots**: expose the most requested services
 - **Malware honeypots**: mimic vulnerabilities to lure an attacker and study his/her behavior.



Interaction taxonomy of Franco et al.

Automatic Scanning Tools

- The importance of outlining the network attack surface ignited the creation of various online scanning tools.
- A growing set of services automatically scan the Internet:
 - Shodan, <https://www.shodan.io>
 - CenSys, <https://search.censys.io>
 - ZoomEye, <https://www.zoomeye.ai>
 - PunkSpider, <https://punkspider.org>
- Online services:
 - offer a multitude of features
 - rely upon a multitude of scanning techniques (e.g., banners)
 - various targets, e.g., IoT and the Web.

The image contains two side-by-side screenshots of SSL certificate details for UniPV services, both enclosed in red boxes.

Screenshot 1 (Top): Gestione Credenziali d#039;Ateneo

HTTP/1.1 200 OK
Date: Sun, 24 Aug 2025 02:05:44 GMT
Server: Apache/2.4.28 (Debian)
Set-Cookie: unipv-account-frontend=7druhqa99t7d0llfe030n
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Retry-After: 120
Se...

90.147.108.57
account.unipv.it
apps.unipv.it
UNI-Pavia
Italy, Pavia

Screenshot 2 (Bottom): Progetti COR

HTTP/1.1 200 OK
Date: Sat, 16 Aug 2025 19:14:43 GMT
Server: Apache/2.4.10 (Debian)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post
Pragma: no-cache
Vary: Accept-Encoding
Set-Cookie: unipv-aziende=d03p1qd12dlknmarie2v3kg6n...
193.204.40.161
libra2017.unipv.it
aziendecor.unipv.it
UNI-Pavia
Italy, Pavia

Example: query “UniPV” with Shodan

Wrap Up

- **Networks** are a **pervasive** component for IT/OT deployments and if targeted may lead to **service disruption** or **severe data losses**.
- **Several attacks** are possible and aims at causing **misbehaviors** or (complete) **blockages** of a service.
- Networks also are as a **prime entry point** for attackers and they can be scanned for **services**, **vulnerabilities**, or **misconfigurations**.
- Tools like **port scanners** are important also for performing **assessments**.
- Prime **defensive mechanisms** are **firewalls**, **NATs** and **honeypots**.
- A good strategy for **reducing the attack surface** entails **segmentation**, which aims “isolating” devices or portions of the network.