# 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

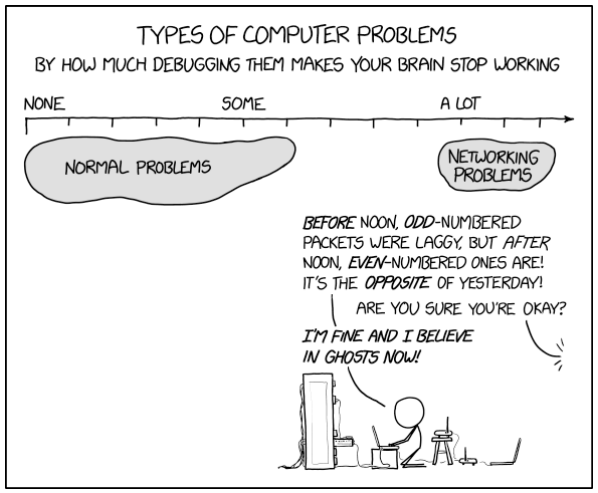
- Why network security in this course?
- Possible Network Attack Types (sniffing and spoofing)
- Denial of Service and Distributed Denial of Service
- Firewalls
- (Network) Vulnerability Scanning
- Policy Enforcing and Hardening
- Network Segmentation
- Network Address Translation - NAT
- Honeypot
-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

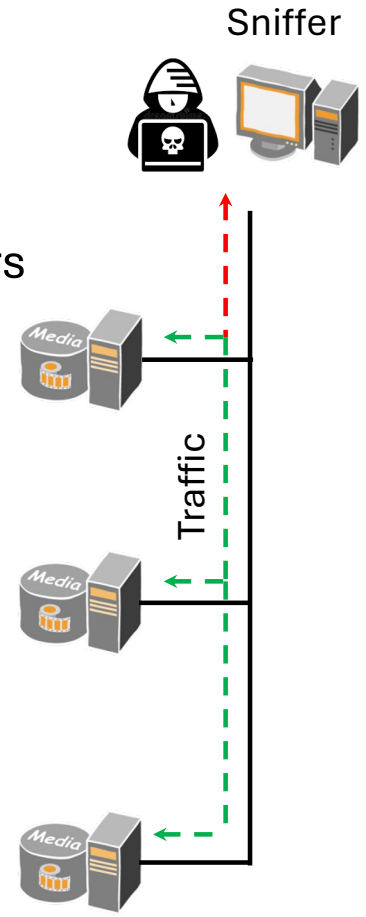
- Networks can be targeted by several attack types:
  - like any other complex systems
  - a **very vast attack surface** to control!
- Main attack types:
  - data interception
  - spoofing
  - redirections and routing attacks
  - violation of authentication/encryption mechanisms
  - DoS/DDoS
  - network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




Source: xkcd

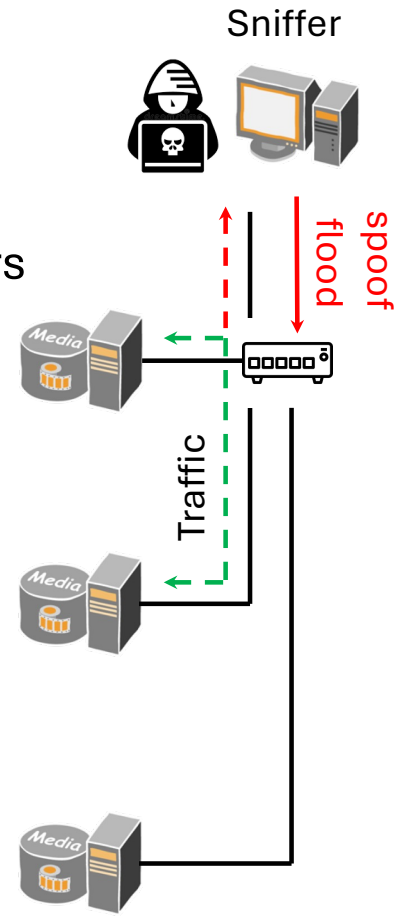
# Sniffing (or eavesdropping)

- 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 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.
- 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)

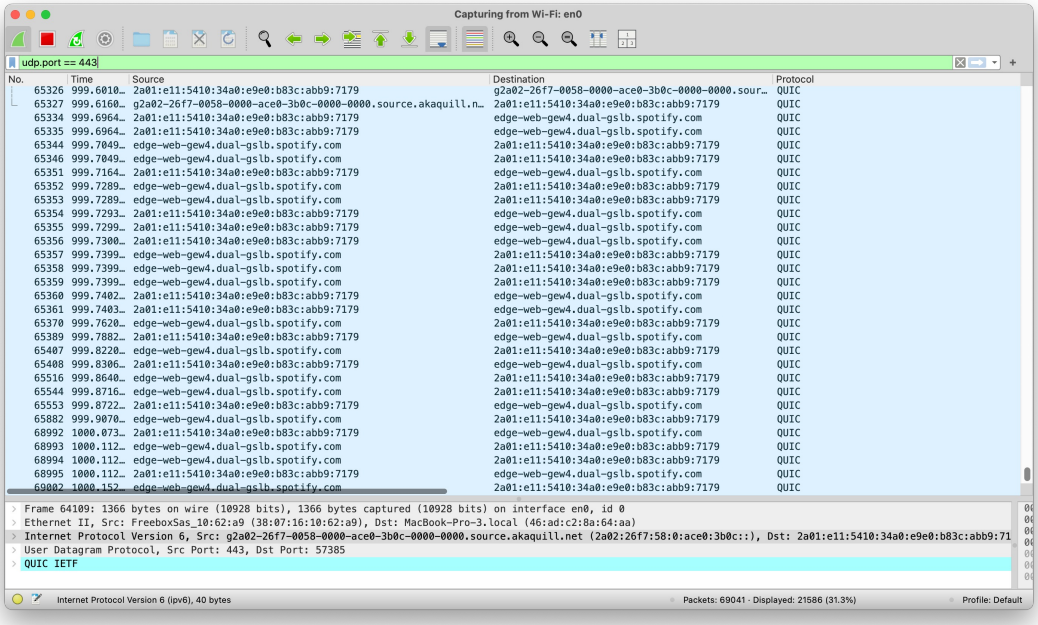
- Most common network sniffers (or protocol analyzer) are:
  - Wireshark (cross platform)
  - tcpdump (Unix-like systems)
- Differences between **Windows** and **Unix** worlds:
  - Windows sniffers are typically based on **winpcap** or **npcap** libraries
  - Unix-like sniffers are typically based on 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

```
[➜  ~ ifconfig| 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!

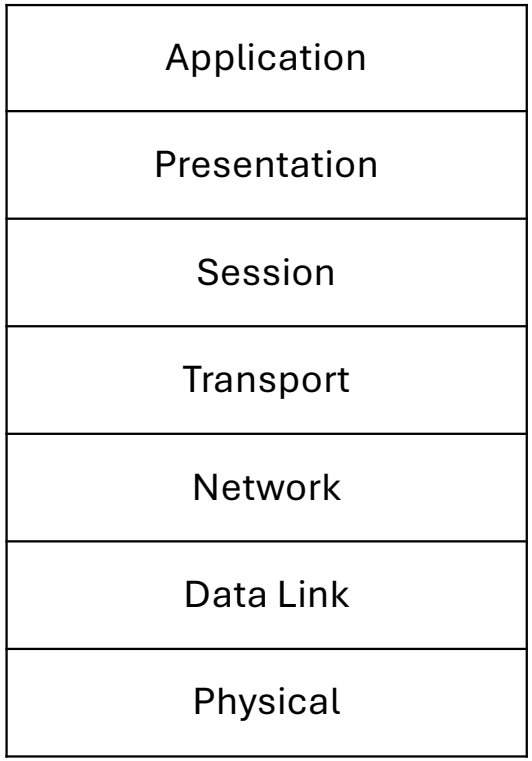
The tool should be considered as “agnostic”, i.e., it can be used for troubleshooting, understanding, as well as to collect traffic for reconnaissance and other malicious purposes (e.g., collecting passwords from weak protocols as in CVE-2021-23846).




…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.

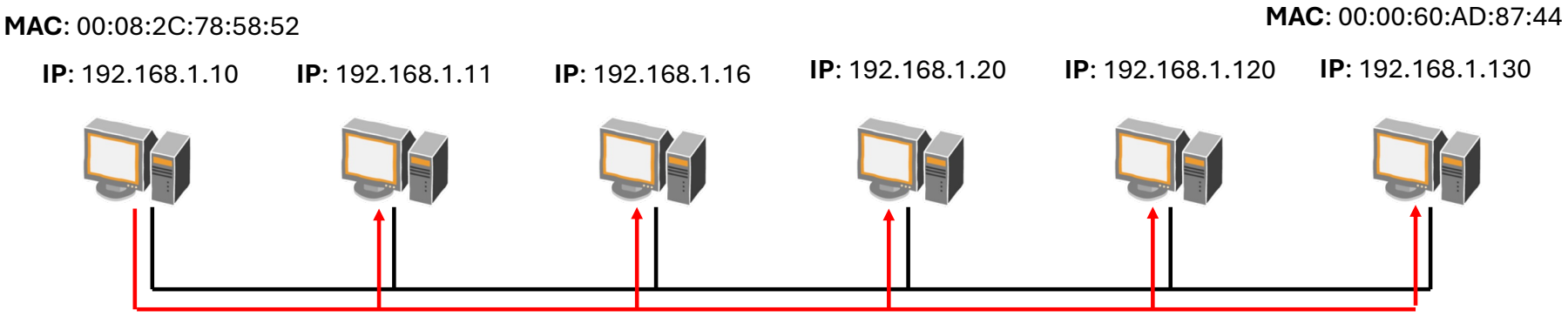



ISO/OSI Reference Model

# 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
  - …

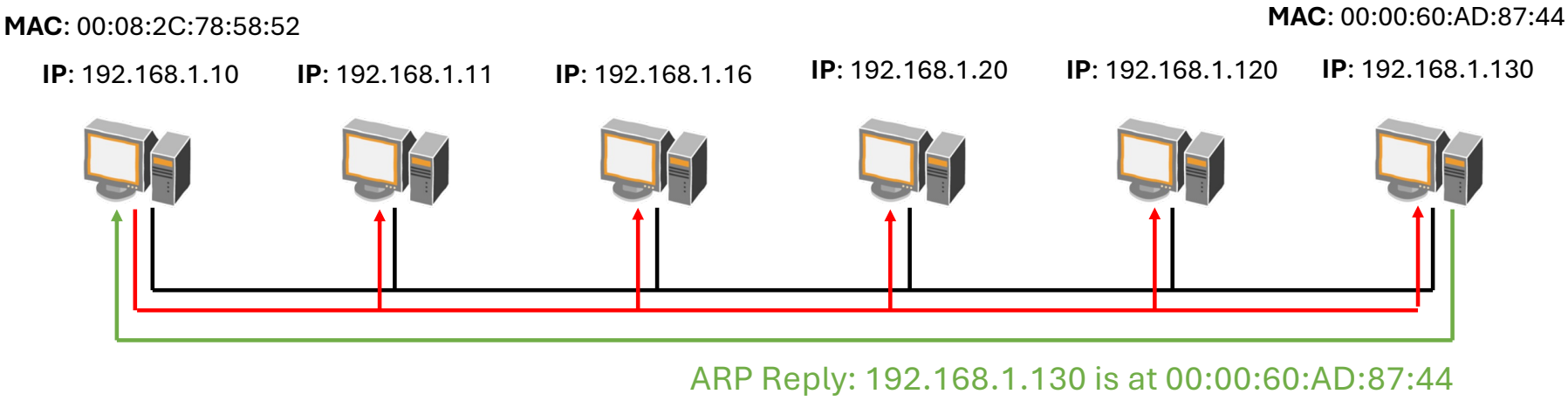
# The Hitchhiker's Guide to ARP




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 |

# The Hitchhiker's Guide to ARP

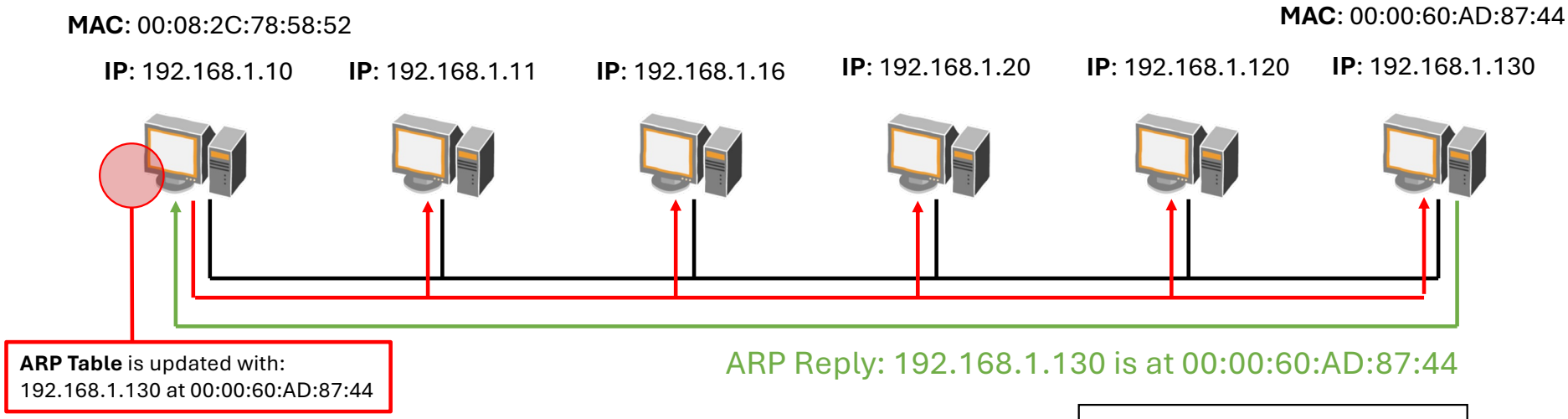



ARP Request: who has 192.168.1.130?

| | |
|---|---|
| **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



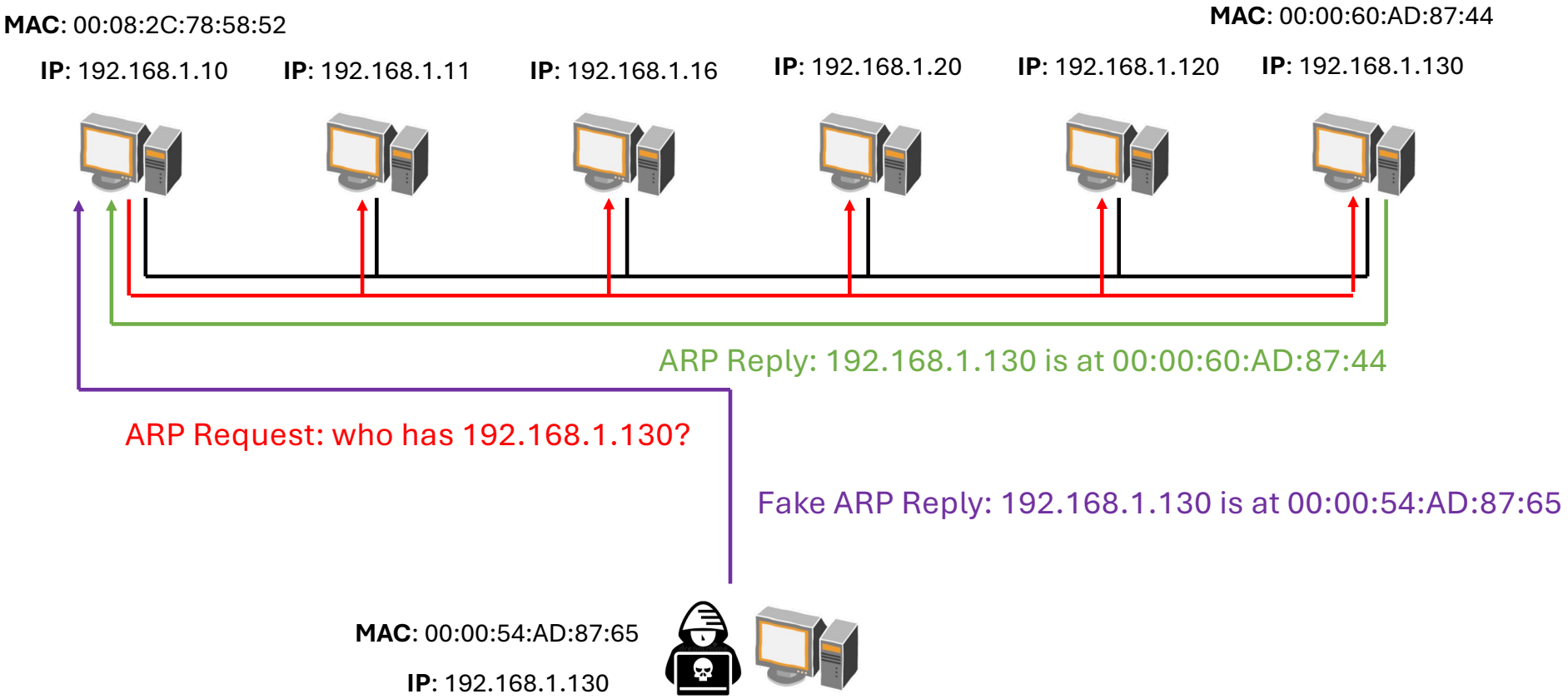

ARP Request: who has 192.168.1.130?

| | |
|---|---|
| **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 |

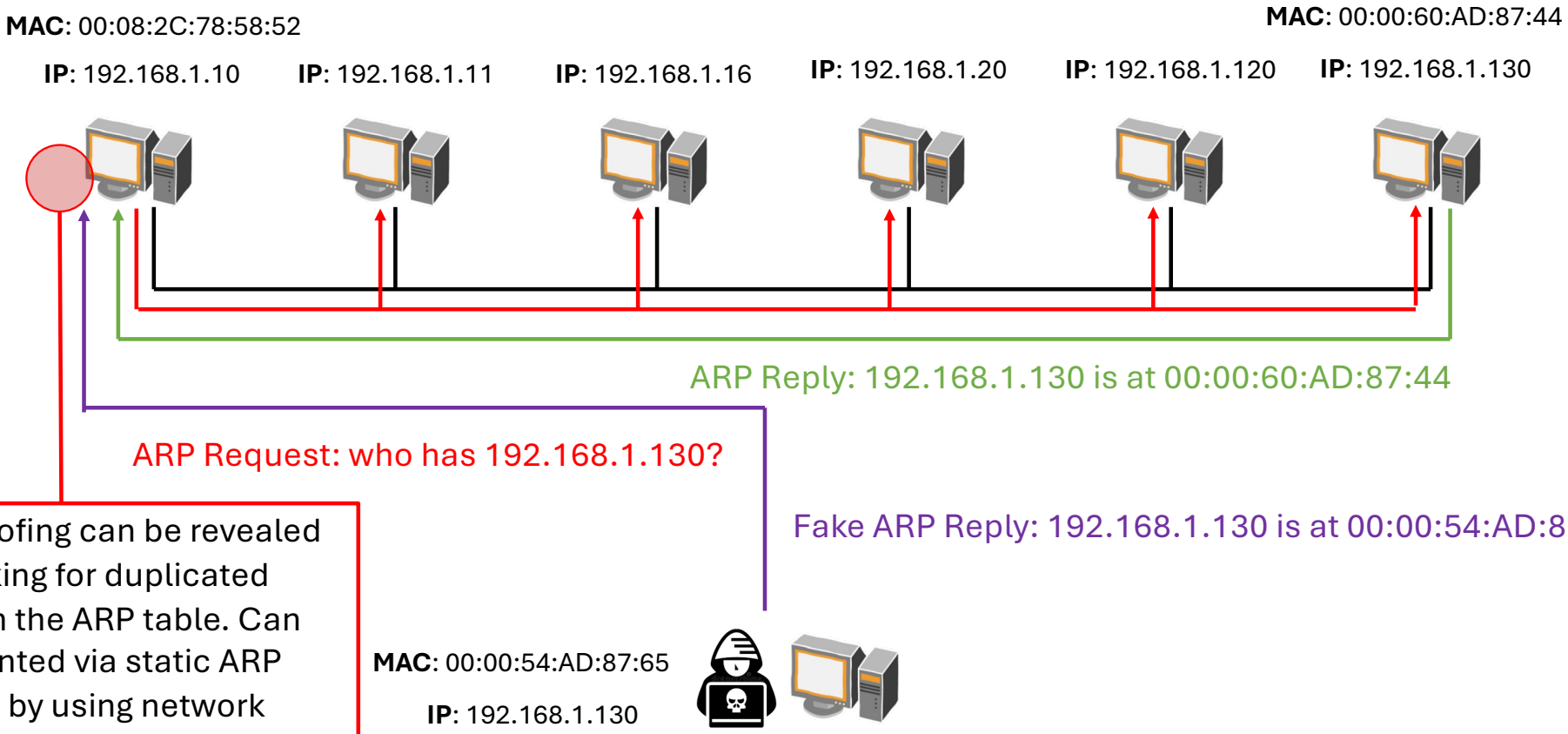
| | |
|---|---|
| **To:** | 00:08:2C:78:58:52 |
| **From:** | 00:00:60:AD:87:44 |
| **MAC-S** | 00:00:60:AD:87:44 |
| **MAC-T** | 00:08:2C:78:58:52 |
| **IP-S** | 192.168.10.130 |
| **IP-T** | 192.168.10.10 |

(Try: arp –a)

# ARP Spoofing



# ARP Spoofing



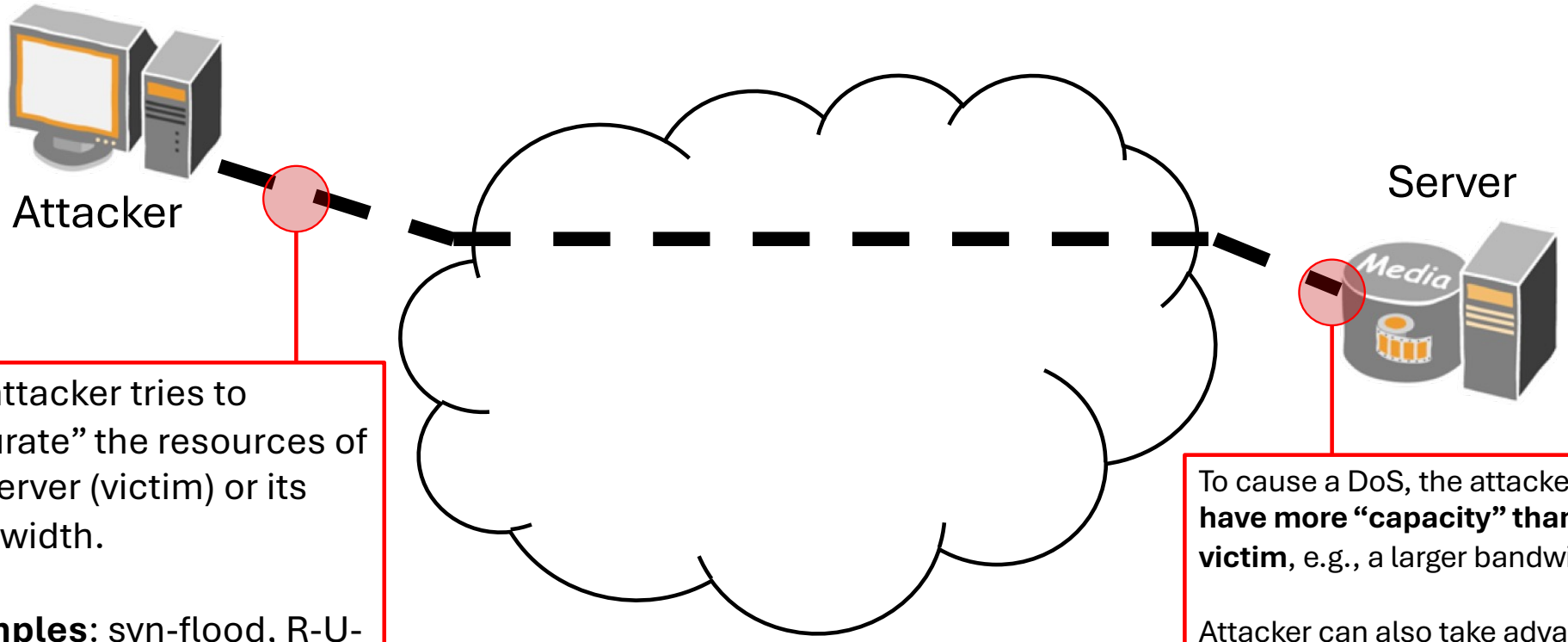

ARP Spoofing can be revealed by checking for duplicated entries in the ARP table. Can be prevented via static ARP tables or by using network isolation (ARP traffic does not travel beyond a sub-network).

# 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



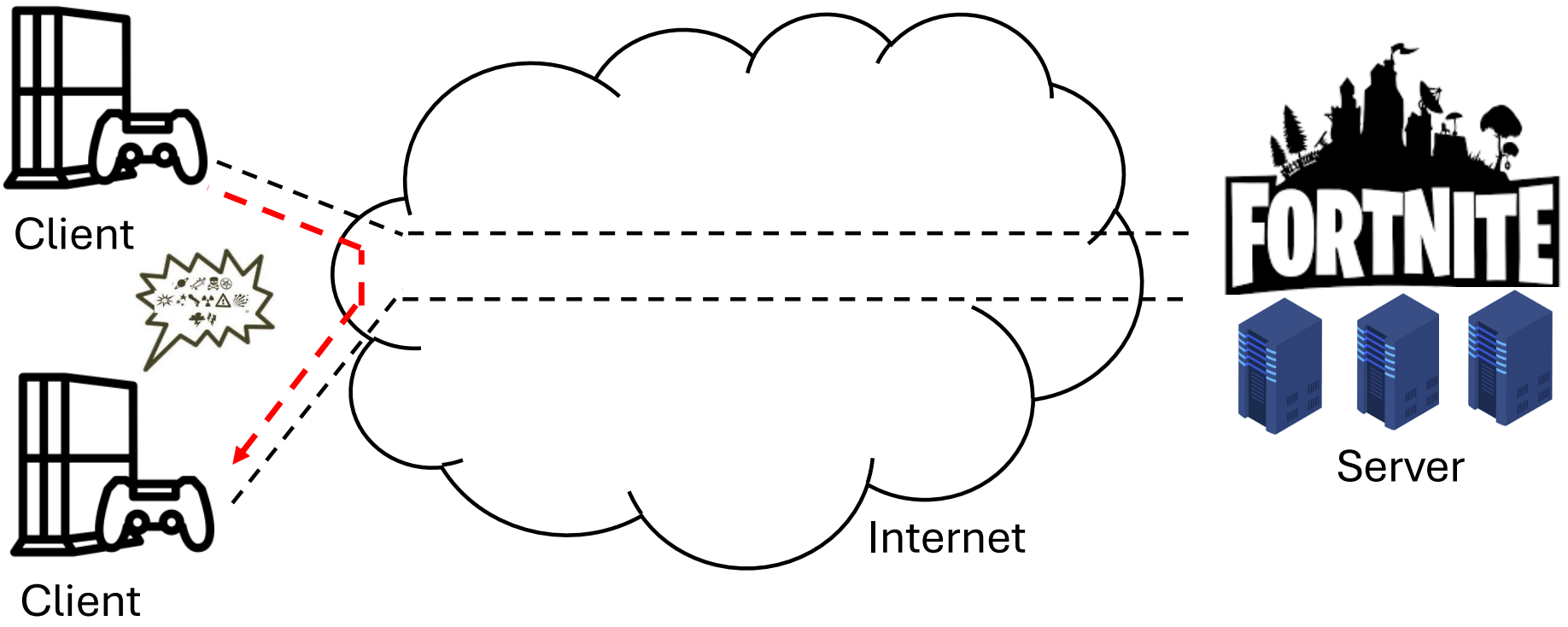

The attacker tries to "saturate" the resources of the server (victim) or its bandwidth.

**Examples**: syn-flood, R-U-Ready Yet?, mailbombing, ping-of-death, etc.

To cause a DoS, the attacker **should have more "capacity" than the victim**, e.g., a larger bandwidth.

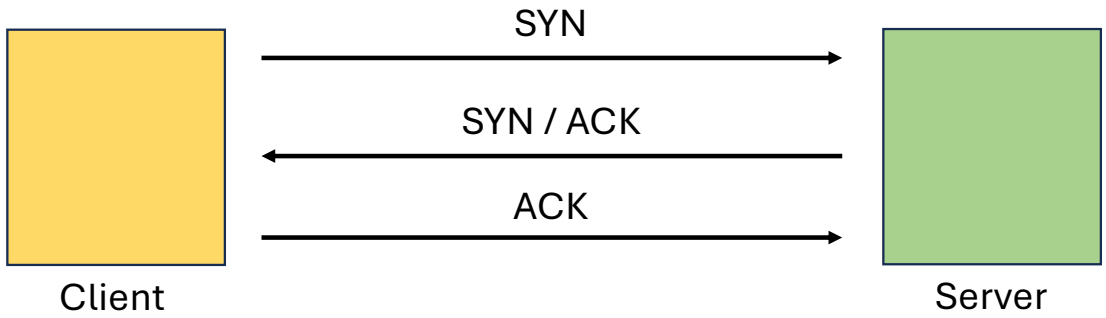
Attacker can also take advantage of bugs, CVEs, and misconfigurations, to deplete resources in a non-brute-force manner.

# Denial of Service: Getting DoSsed

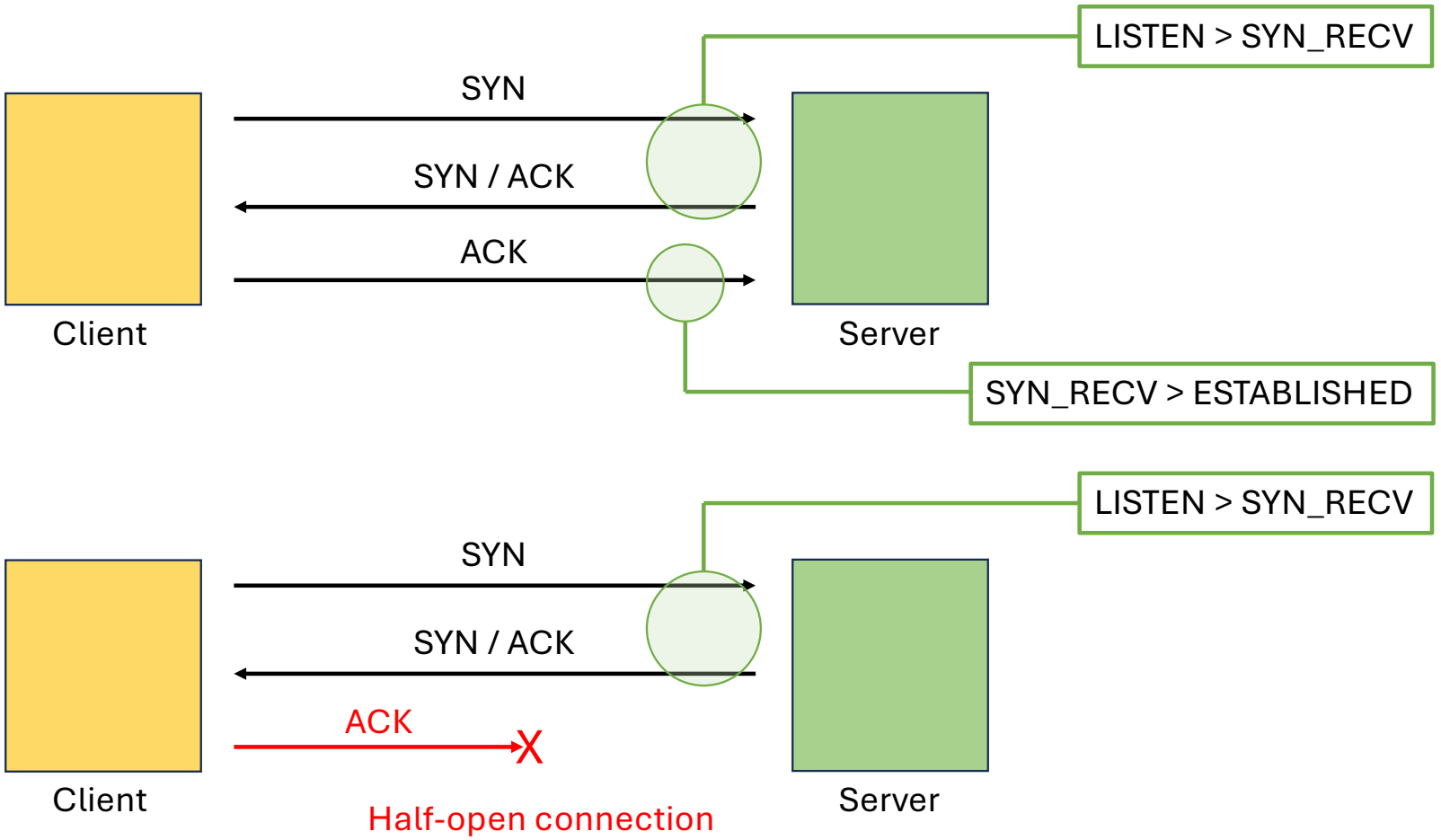


# Denial of Service: SYN Flood

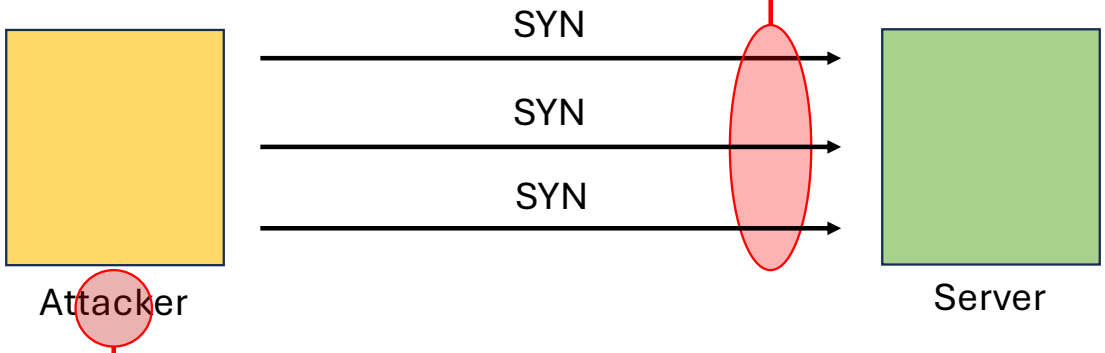
- The attack exploits the **three-way handshake** of TCP:
  - **SYN**: the client initiates a connection with a TCP segment with the SYN flag set to 1 and communicates the Initial Sequence Number (ISN)
  - **SYN/ACK**: the server responds with a TCP segment with both the SYN and ACK flags set to 1 and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

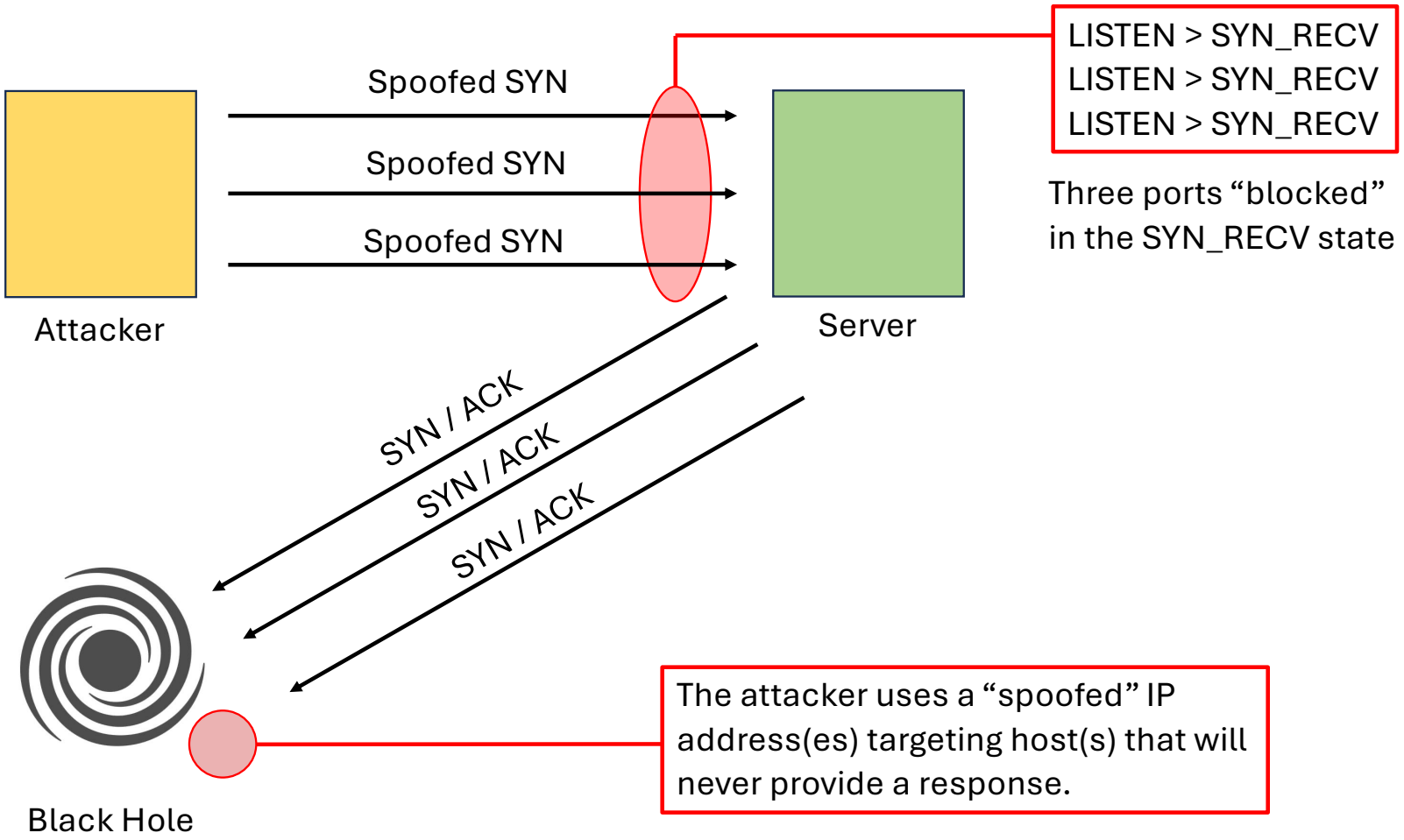



LISTEN > SYN_RECV
LISTEN > SYN_RECV
LISTEN > SYN_RECV

Three ports “blocked” in the SYN_RECV state

A **direct** flood is **seldom** used, as it makes easy to track the attacker and implement a block, e.g., by filtering traffic from an IP address. Instead, it is used in DDoS where the chance of identifying all the traffic sources is scarce.

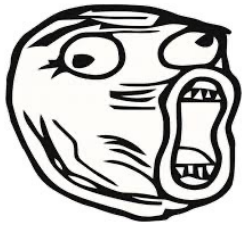
# Denial of Service: SYN Flood



# Denial of Service

Meta datacenter in Luleå (Sweden)

DoS

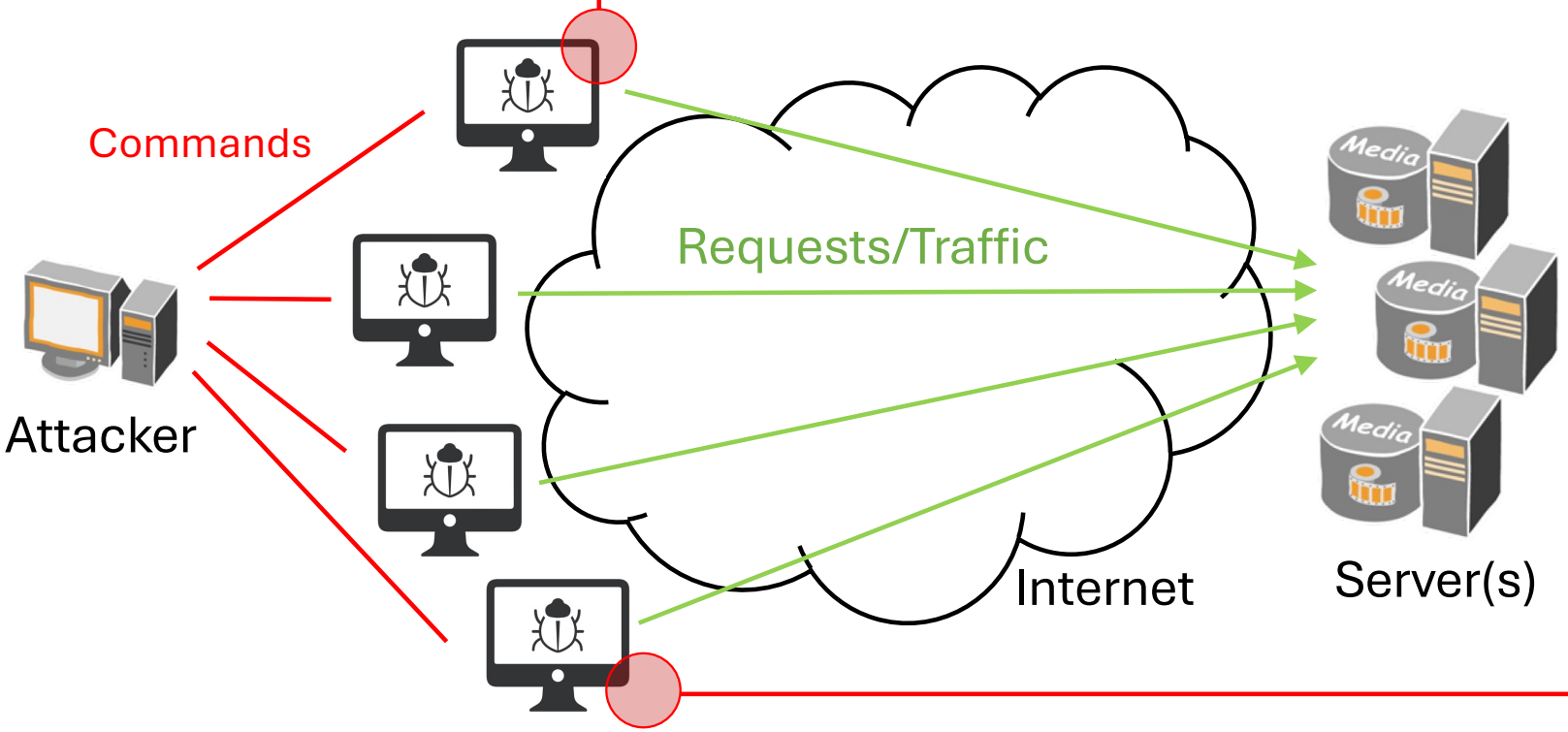


# Distributed Denial of Service

- 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

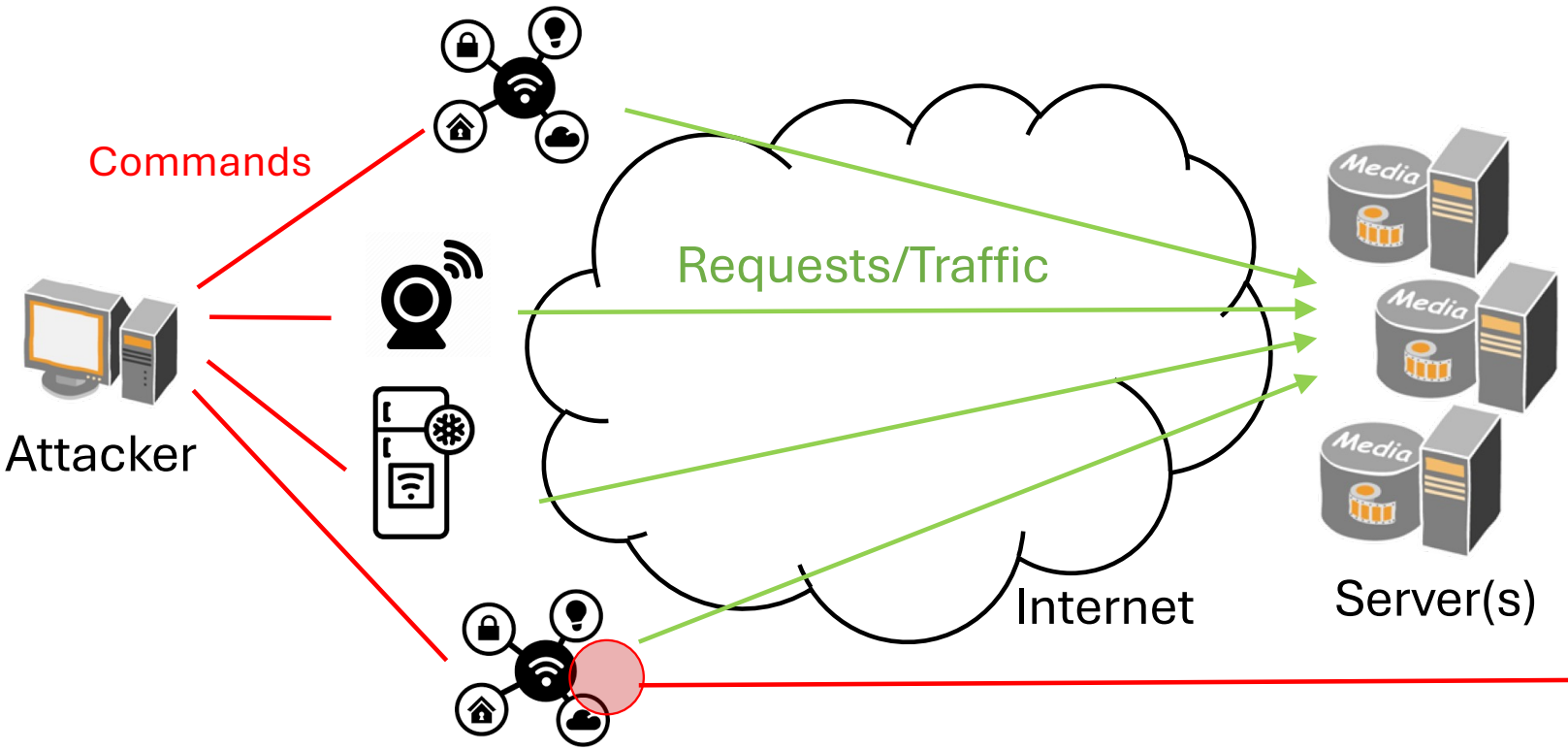
The network of compromised nodes used by the attacker is often called a **botnet**.




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)



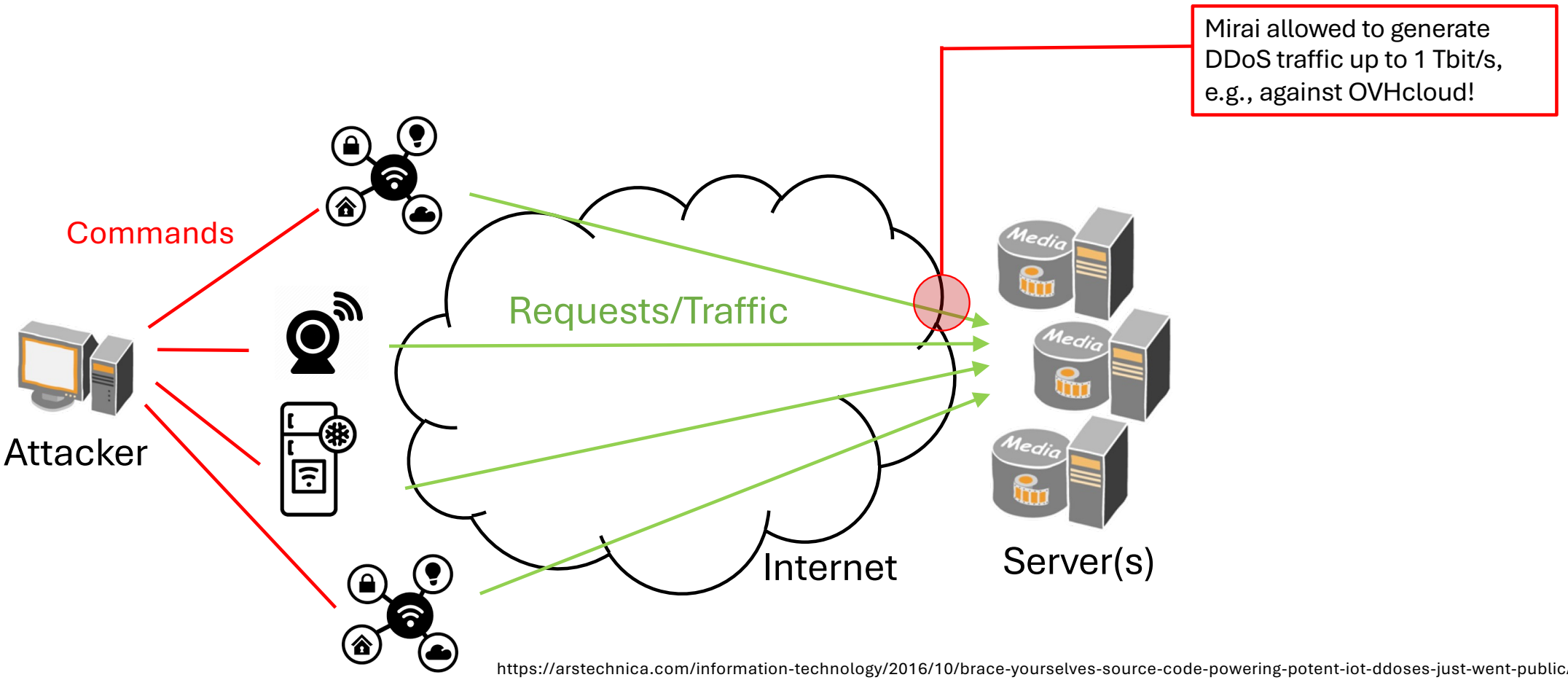

Mirai (未来) is a malware targeting devices based on ARC/Linux, e.g., IP cameras and routers.

**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 contacting specific IP ranges, e.g., DoD.

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

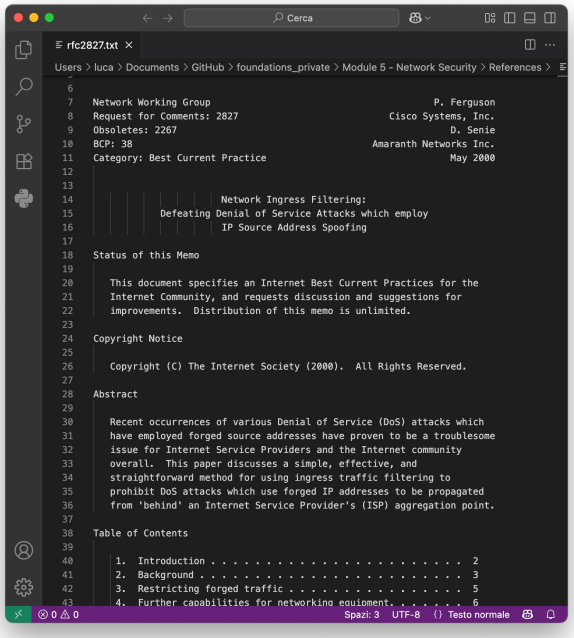
# Distributed Denial of Service: Mirai (2016)




https://arstechnica.com/information-technology/2016/10/brace-yourselves-source-code-powering-potent-iot-ddoses-just-went-public/

# (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

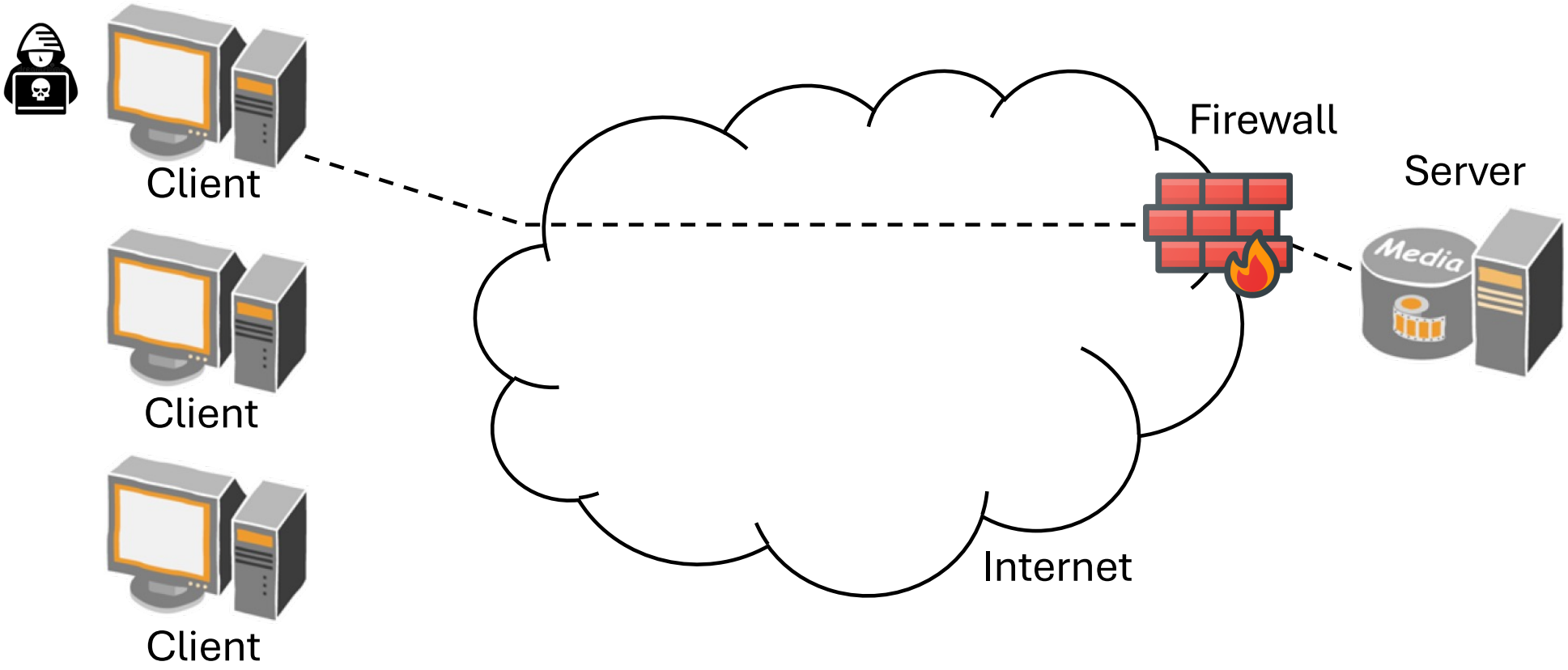
# Firewalls

- 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.

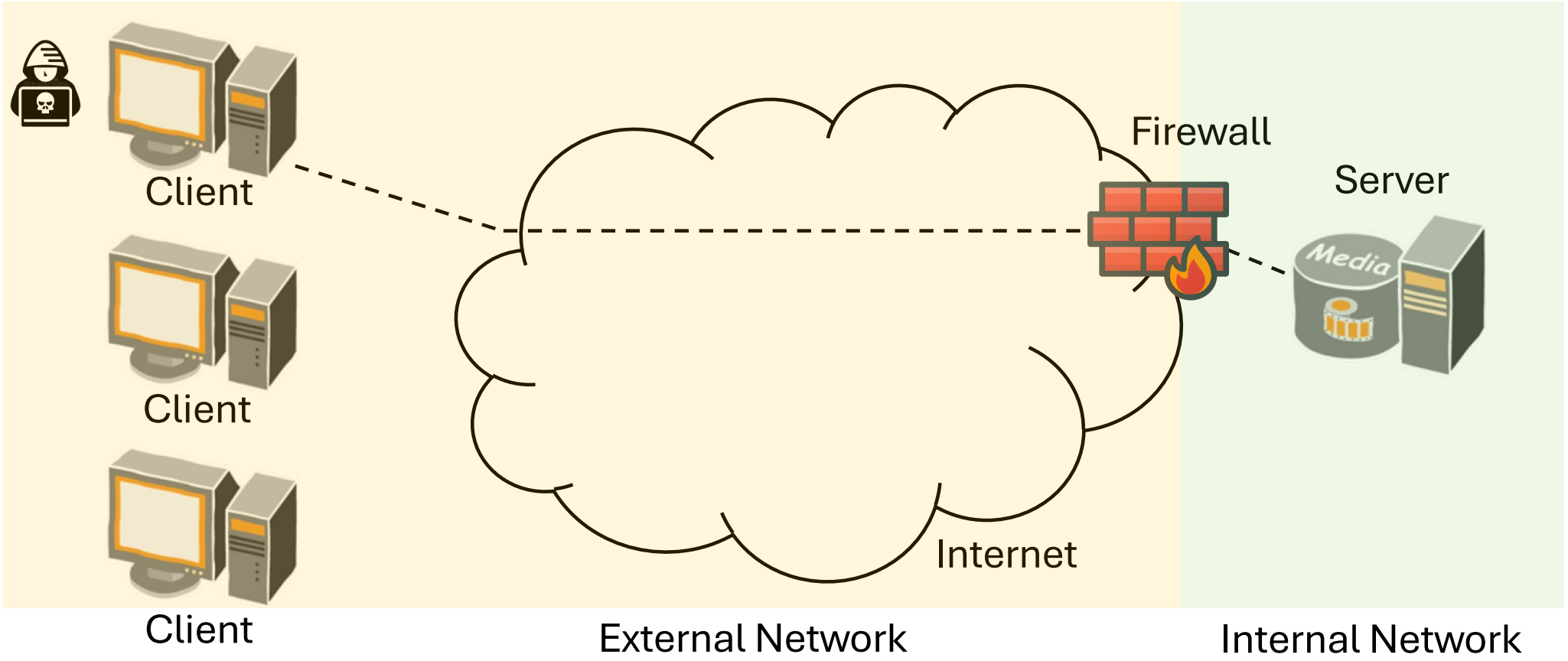
# Firewalls: Packet Filters

- 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

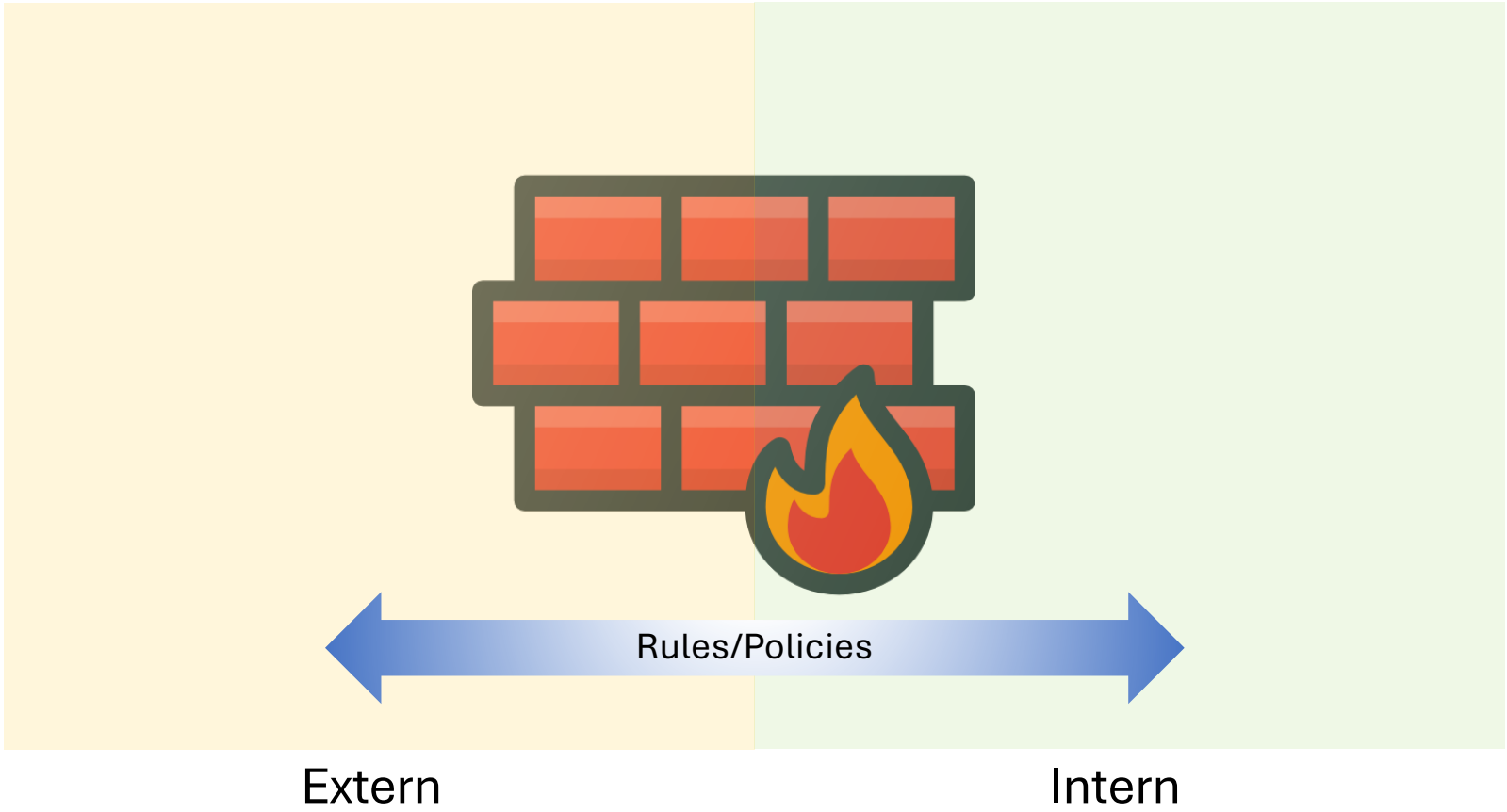
# Firewalls: Packet Filters



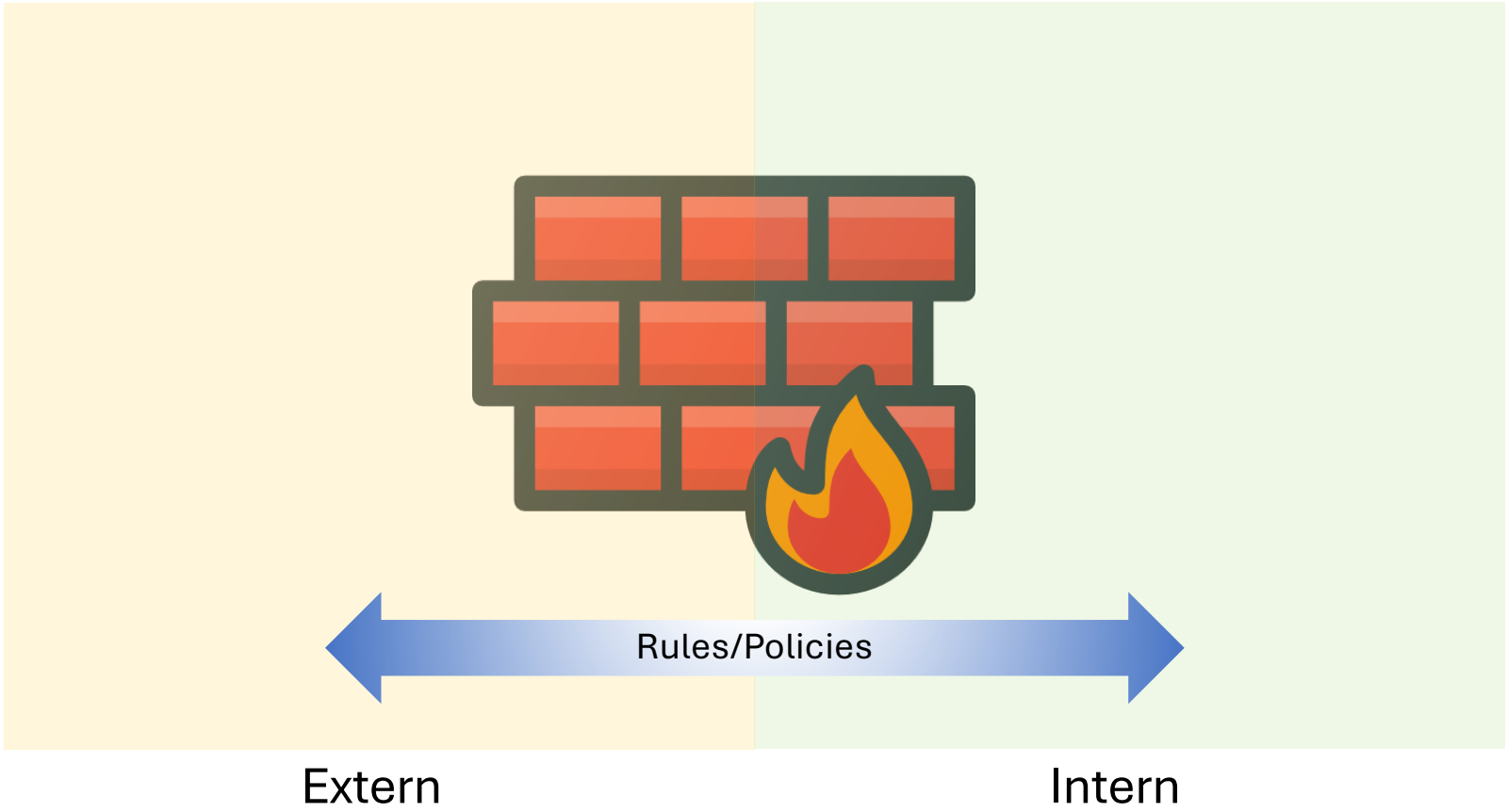
# Firewalls: Packet Filters



# Firewalls: Packet Filters



# Firewalls: Packet Filters



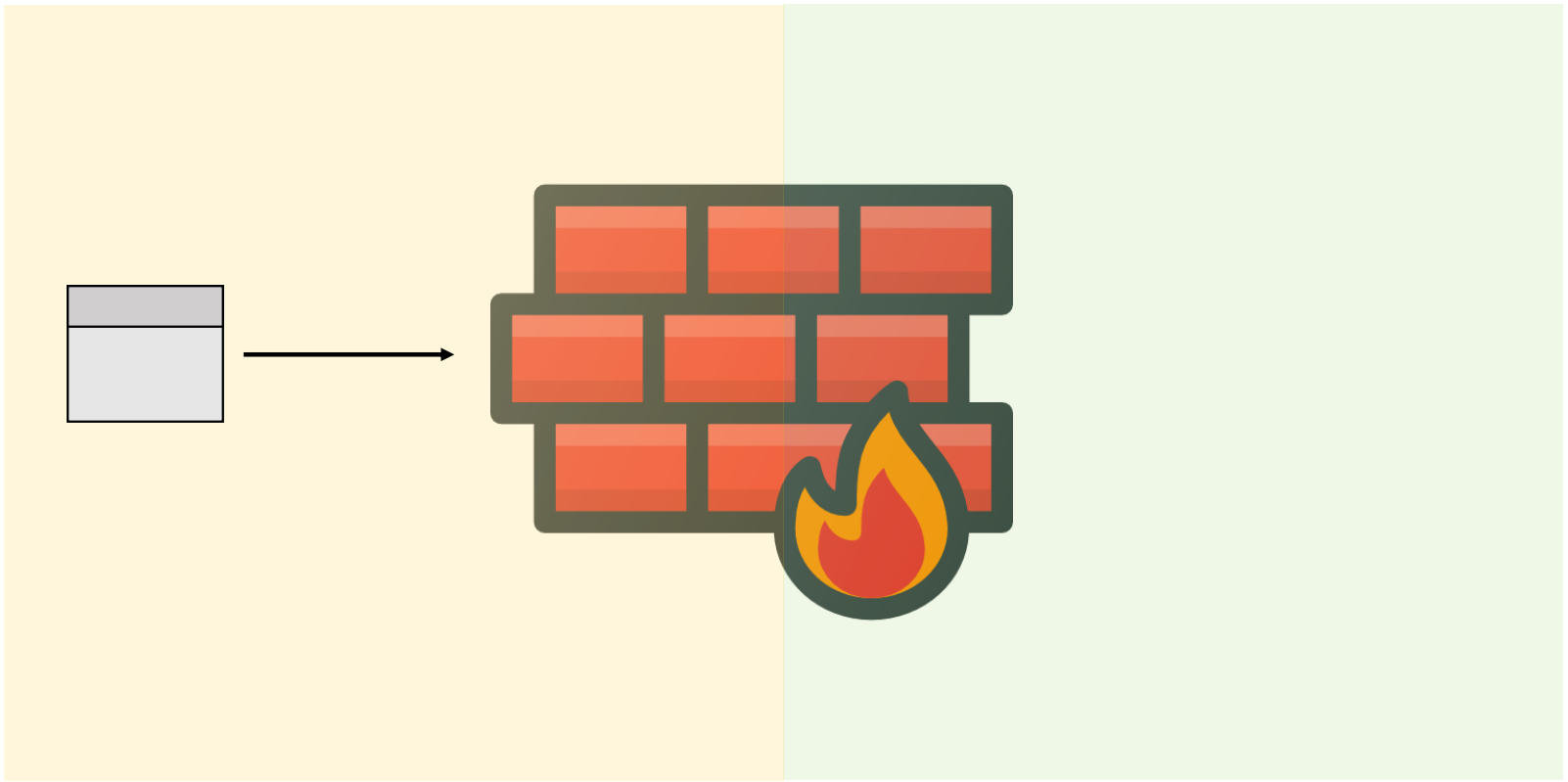

Two possible default policies

**Default Allow**: what is not expressly prohibited is permitted.

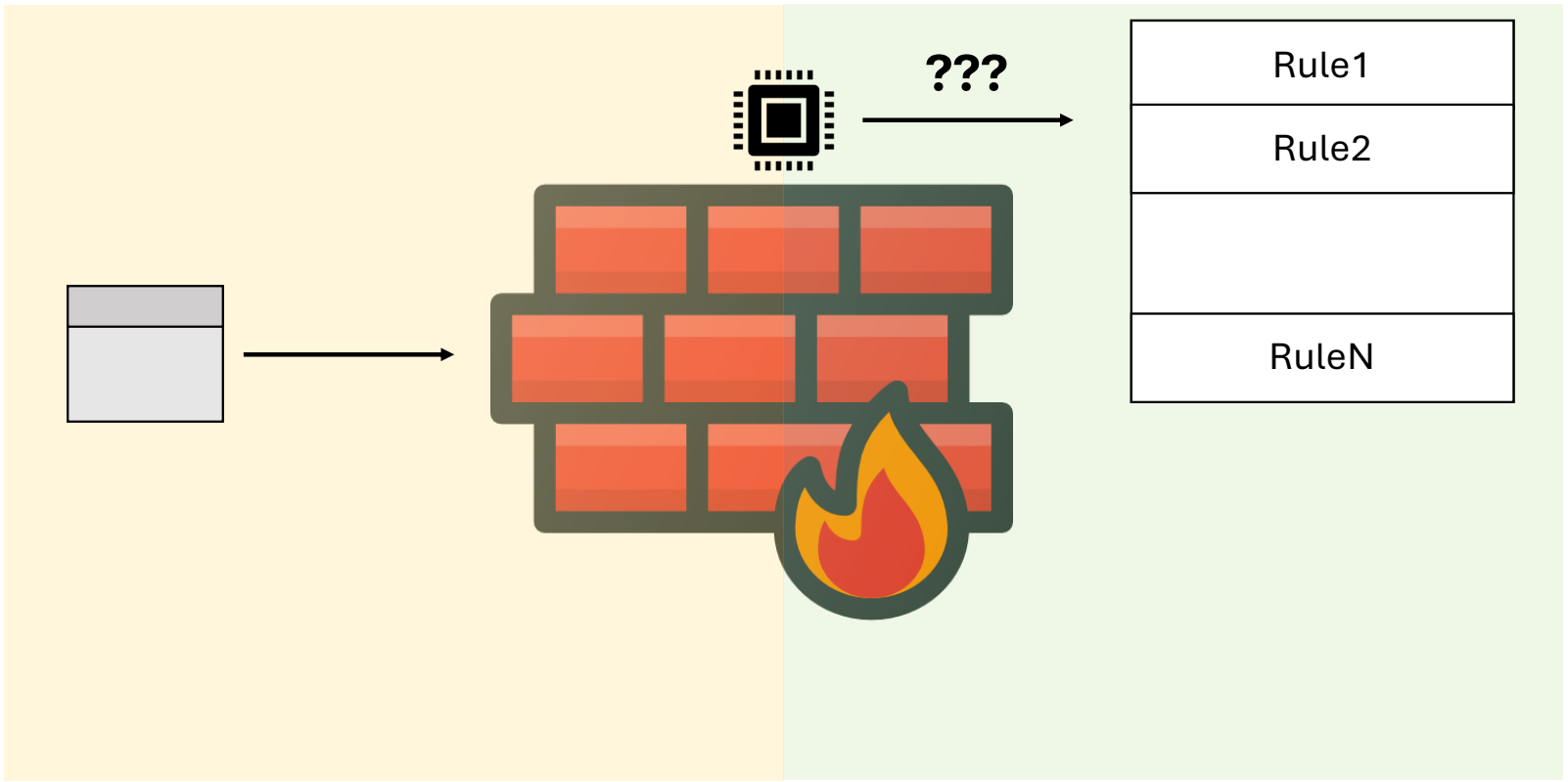
**Default Deny**: what is not expressly permitted is prohibited.

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

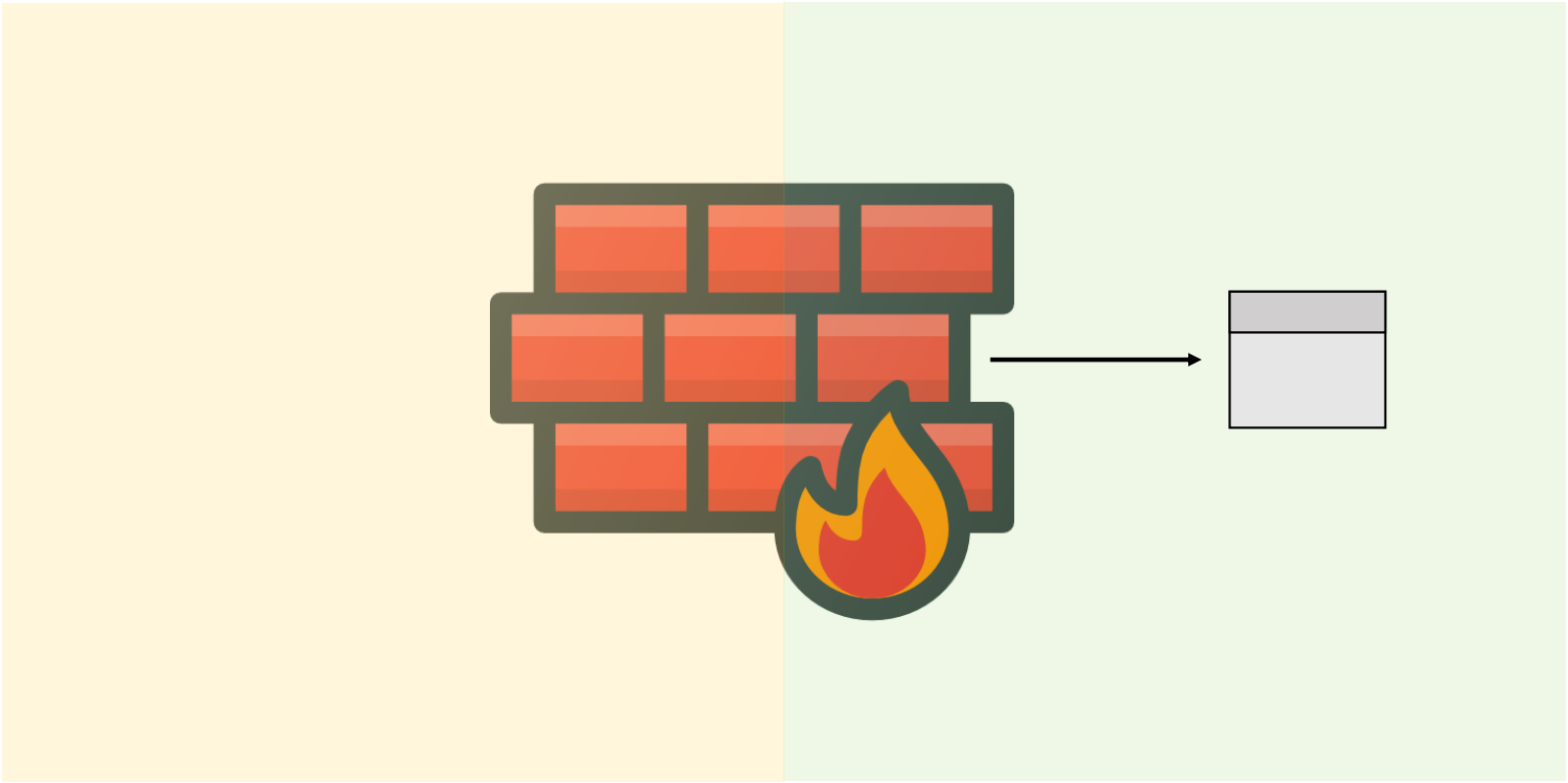
# Firewalls: Packet Filters



# Firewalls: Packet Filters



# Firewalls: Packet Filters



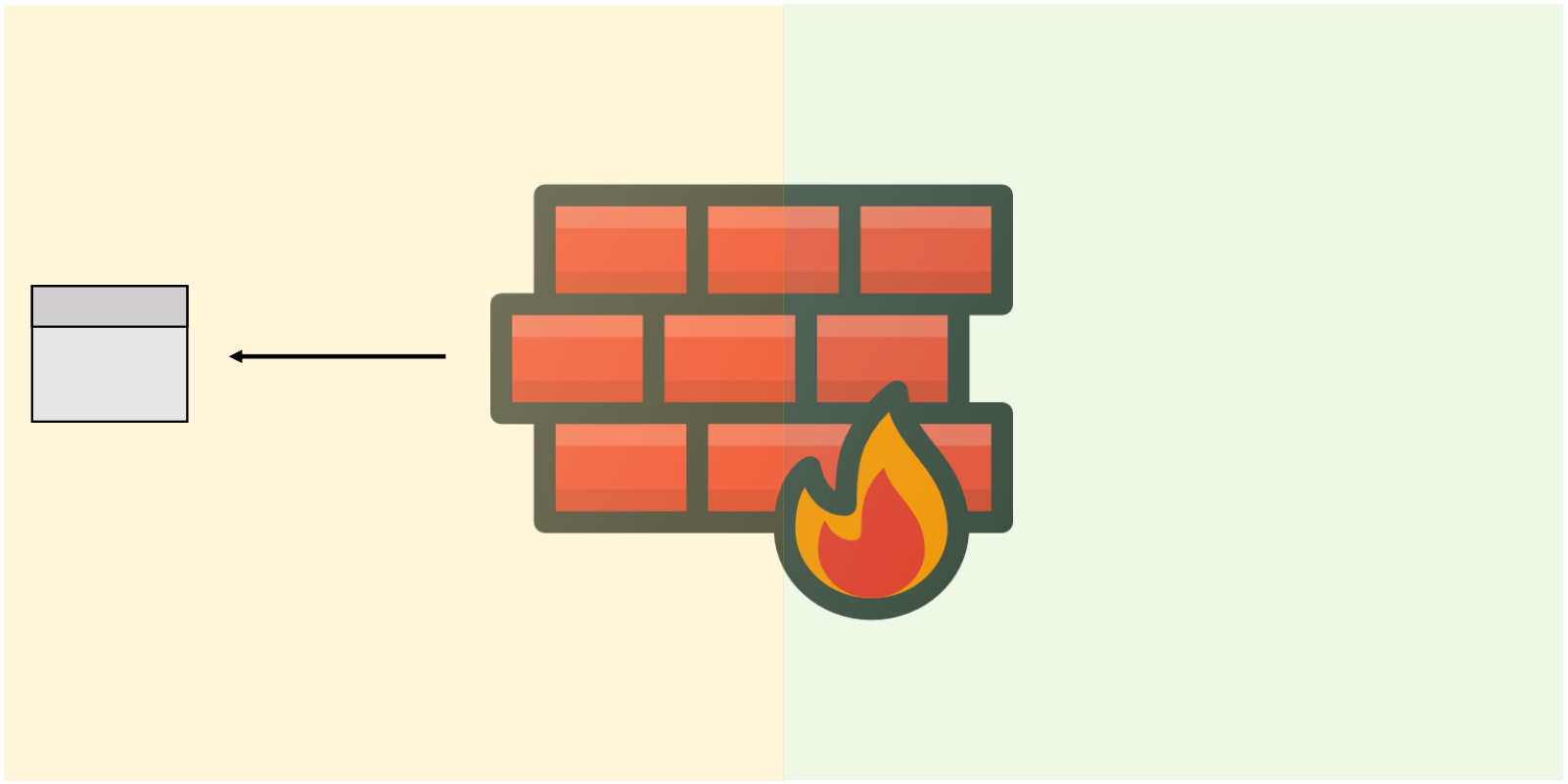
Extern

Intern

Three possible actions:

**Allow**: datagram can pass.

# Firewalls: Packet Filters

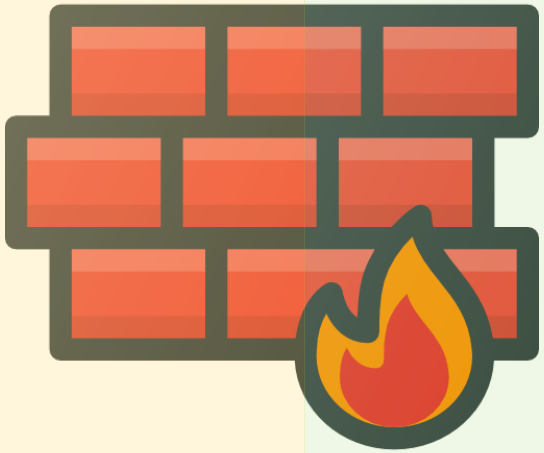



Three possible actions:

**Allow**: datagram can pass.

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

# Firewalls: Packet Filters

Extern

Intern

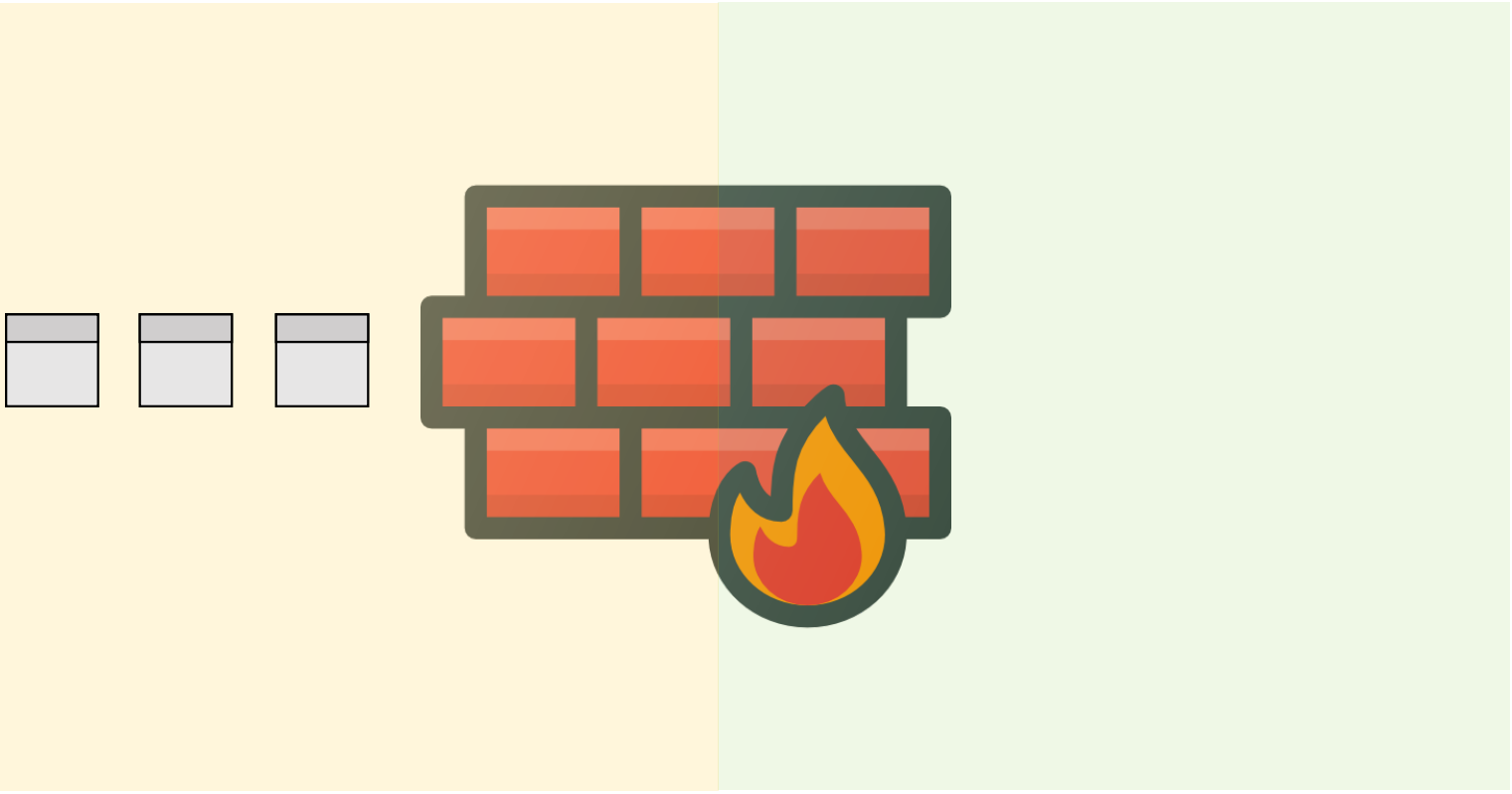
Three possible actions:

**Allow**: datagram can pass.

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

**Drop**: the datagram is discarded.

# Firewalls: Packet Filters



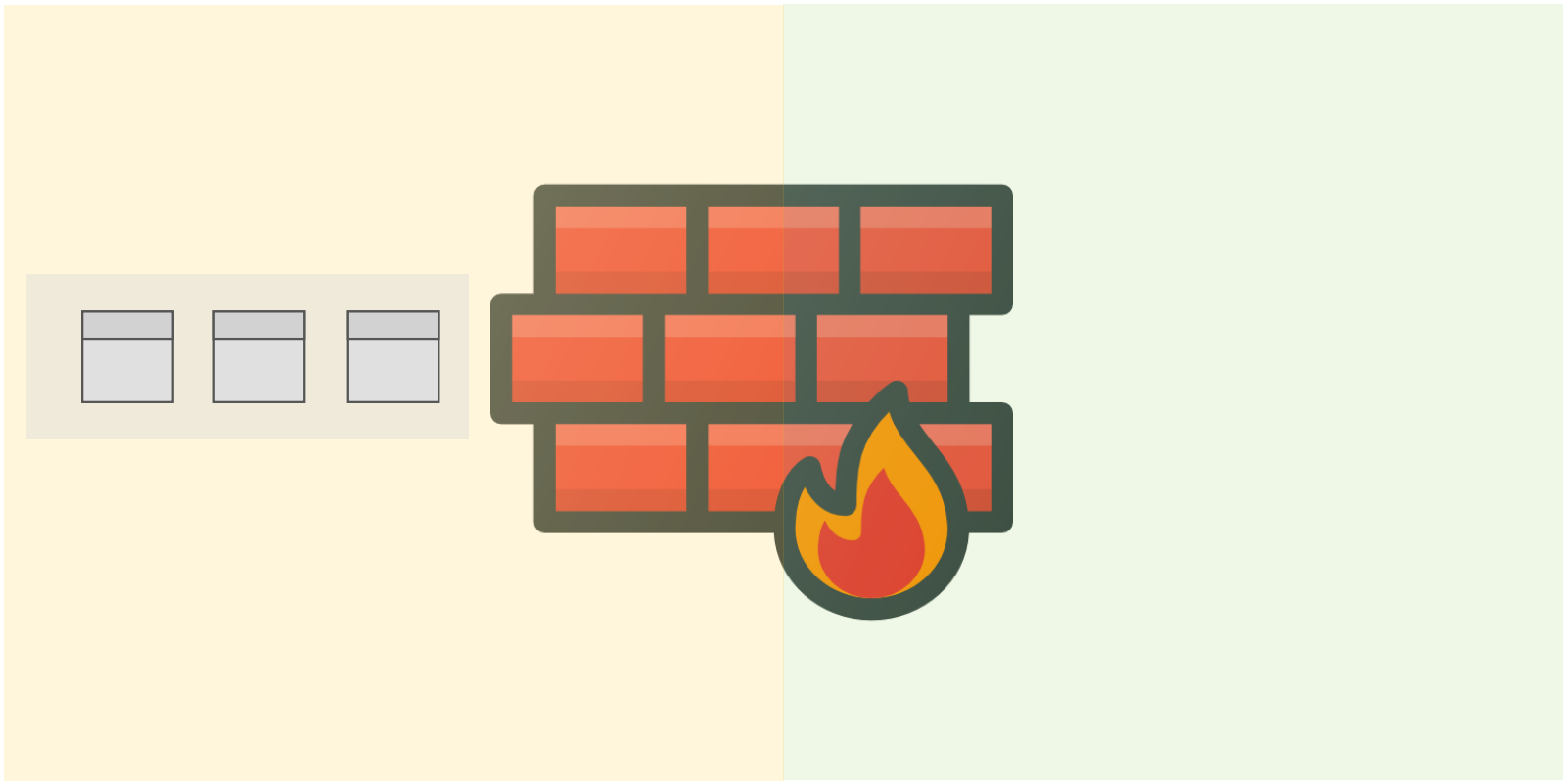

Firewalls can apply rules by considering traffic in two different manners.

**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



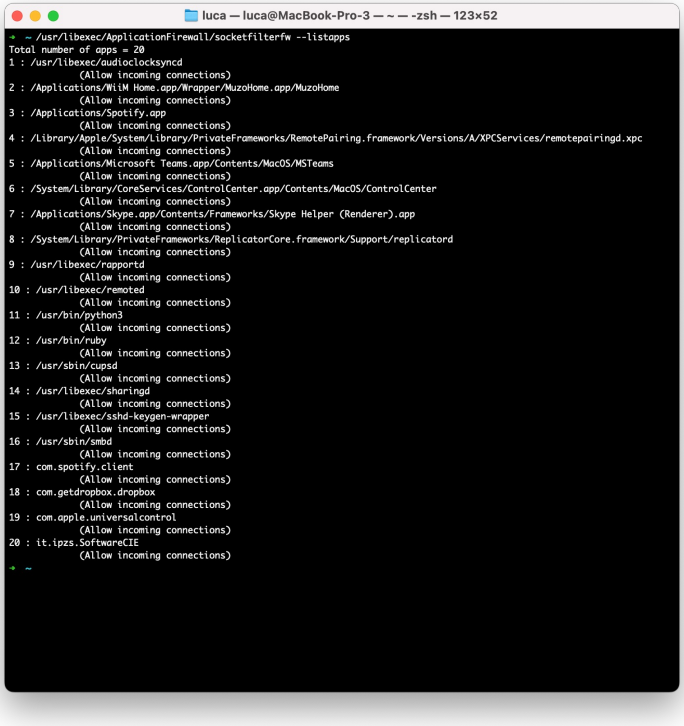

Firewalls can apply rules by considering traffic in two different manners.

**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*



# Firewalls: Packet Filters

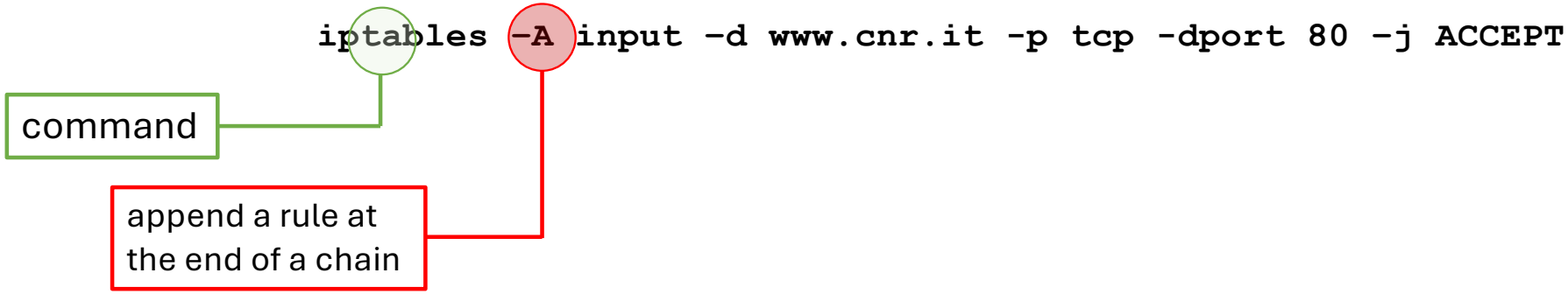
- Examples of filtering rules
  - Linux: inserting a rule in *iptables*

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

command

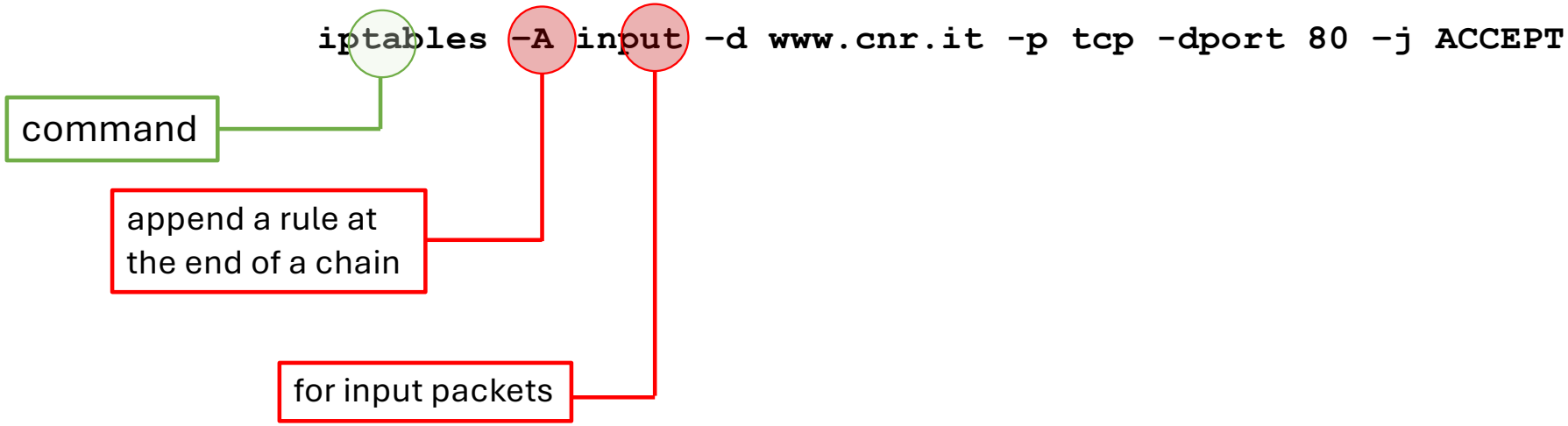
# Firewalls: Packet Filters

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



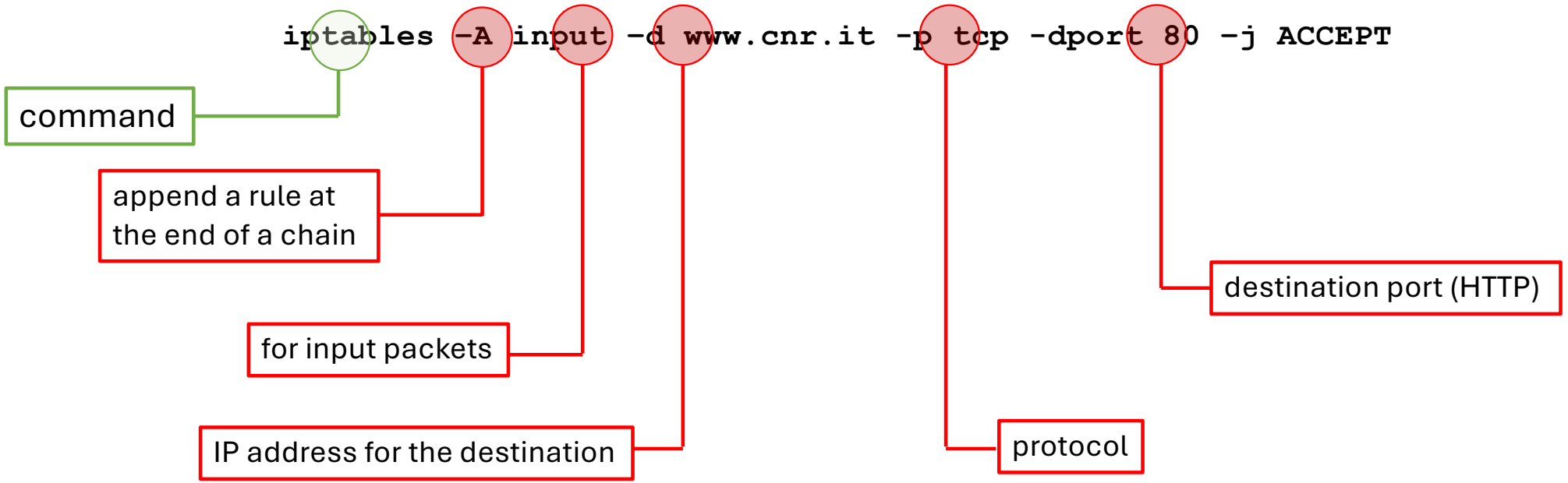
# Firewalls: Packet Filters

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



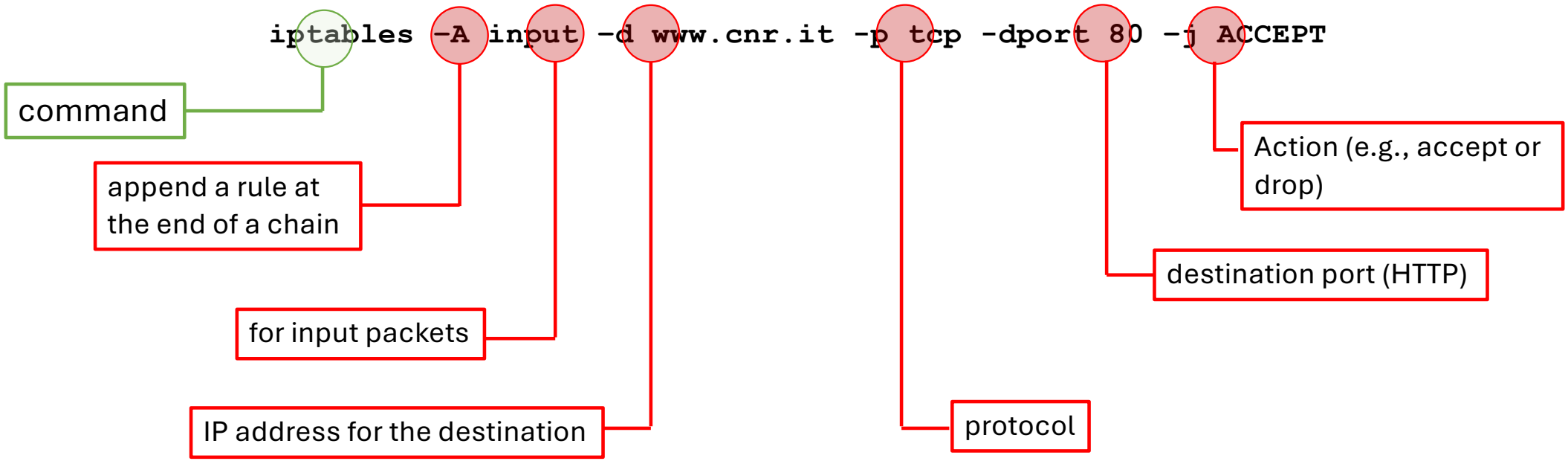
# Firewalls: Packet Filters

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



# Firewalls: Packet Filters

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

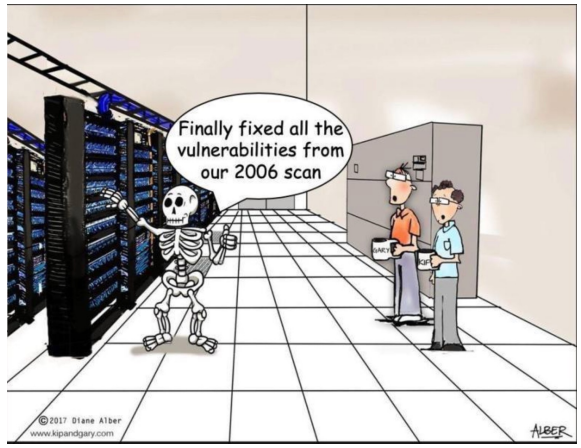


# (Network) Vulnerability Scanning

- 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)

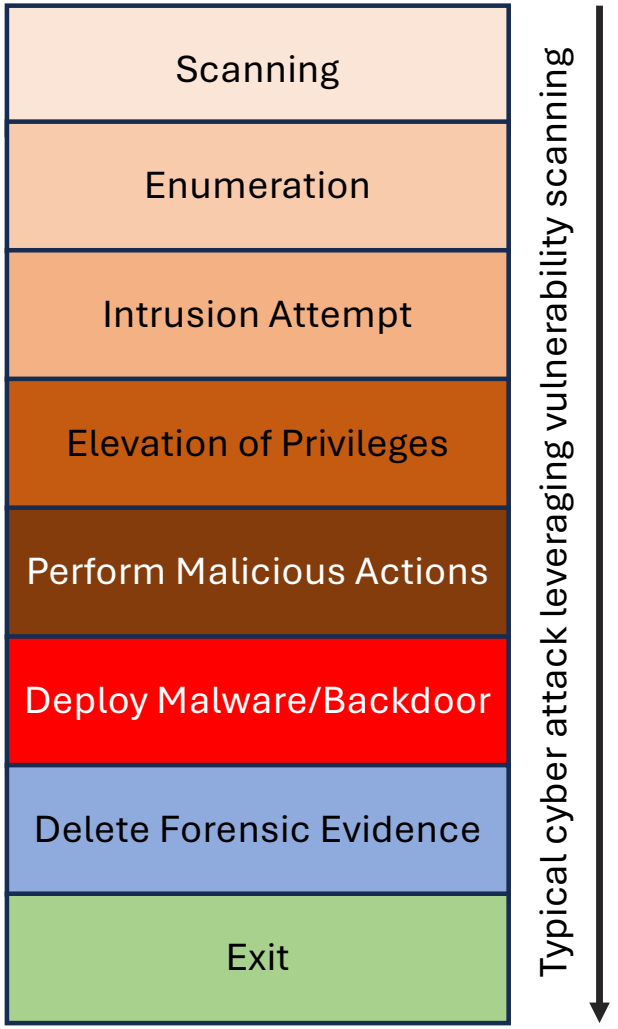
# (Network) Vulnerability Scanning

- 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.

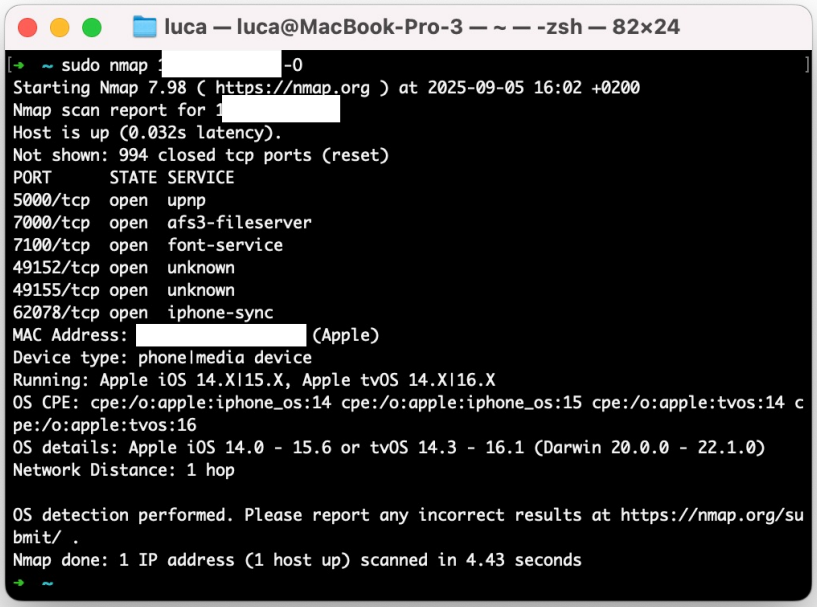


# Network Scanning

- 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*

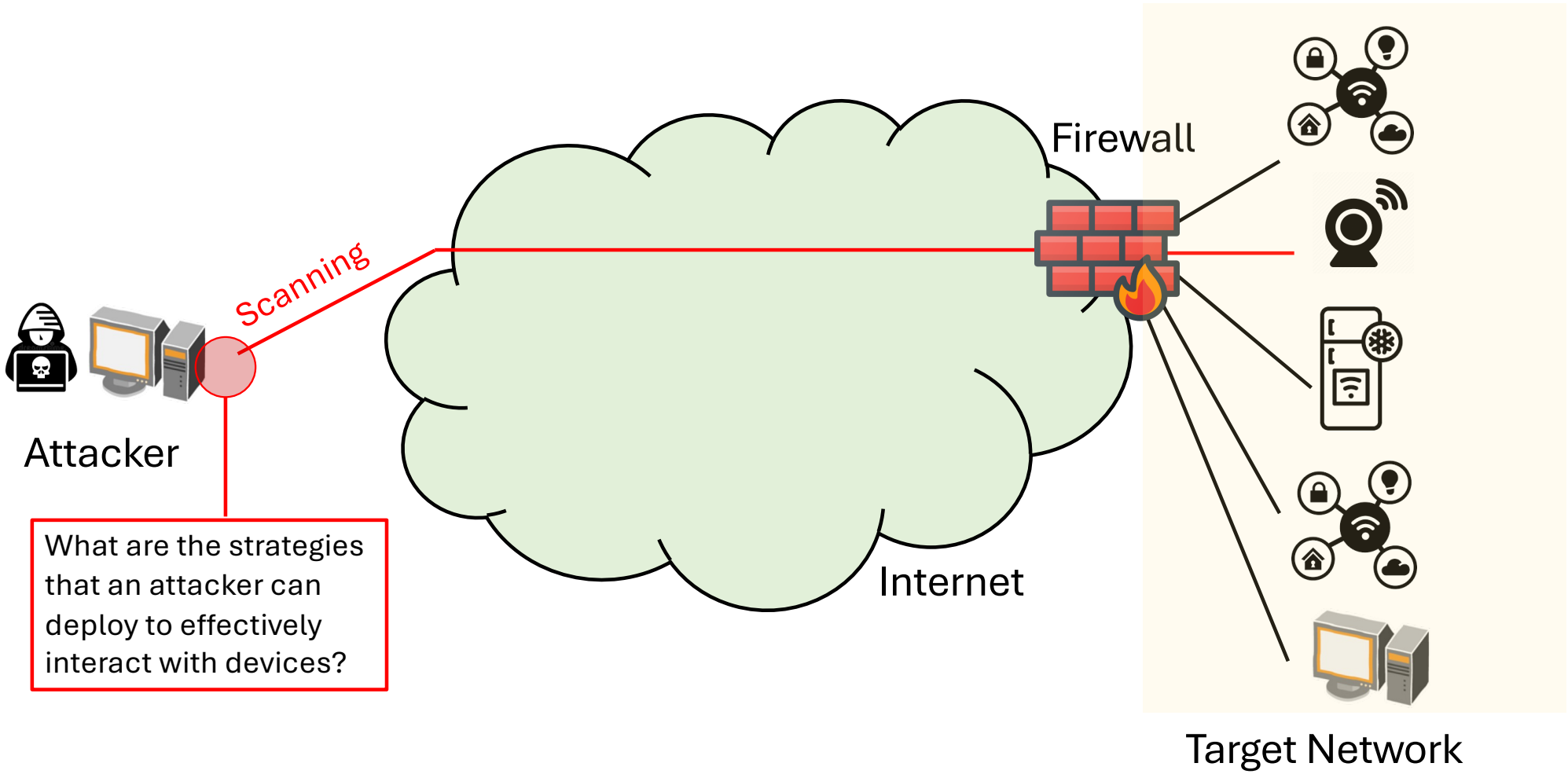
# Network Scanning

- 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.

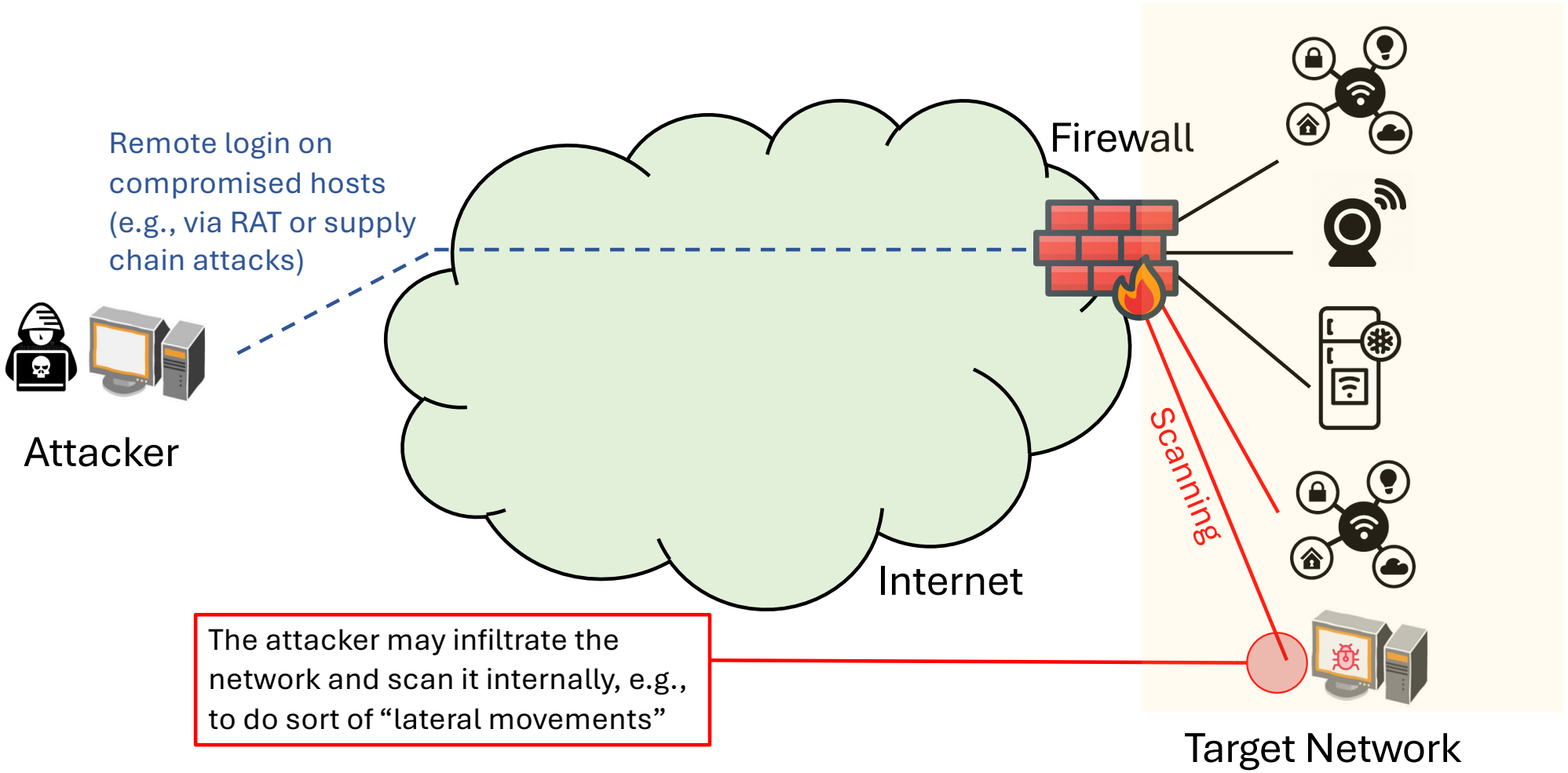



Example of OS fingerprinting

# Scanning Scenario: Remote Campaign

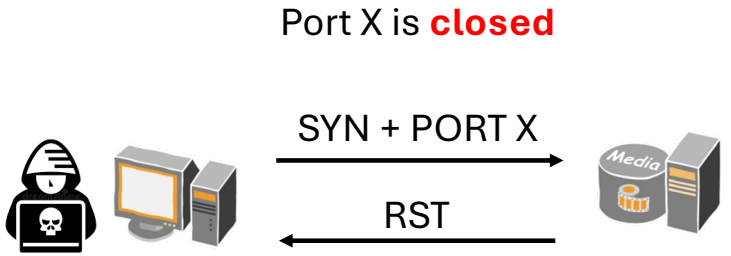


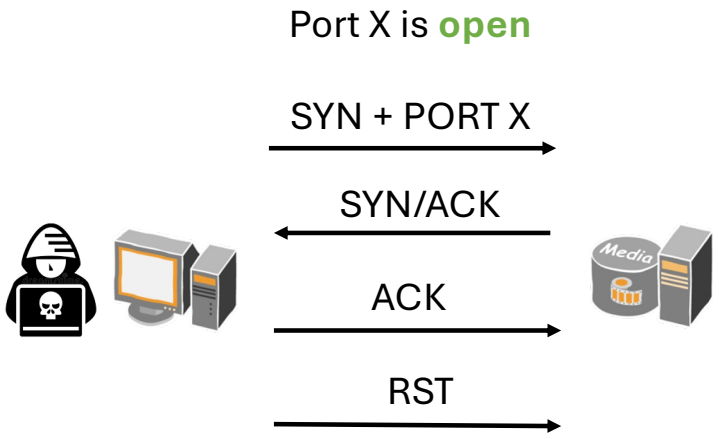
# Scanning Scenario: “Local” Campaign



# Scanning Strategies

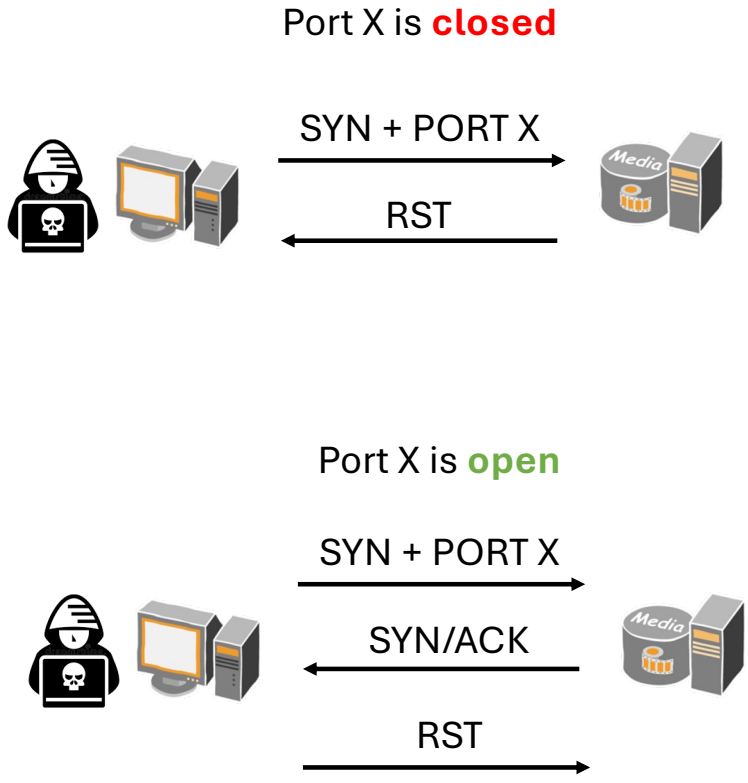
- TCP Scan (also known as open scan or vanilla scan):
  - is the simplest port scanning approach
  - uses the standard *connect( )* system call.
- Procedure:
  - if the port of the target is open, the OS completes the TCP three-way handshake and the scanner closes the connection
  - otherwise, an error 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…)



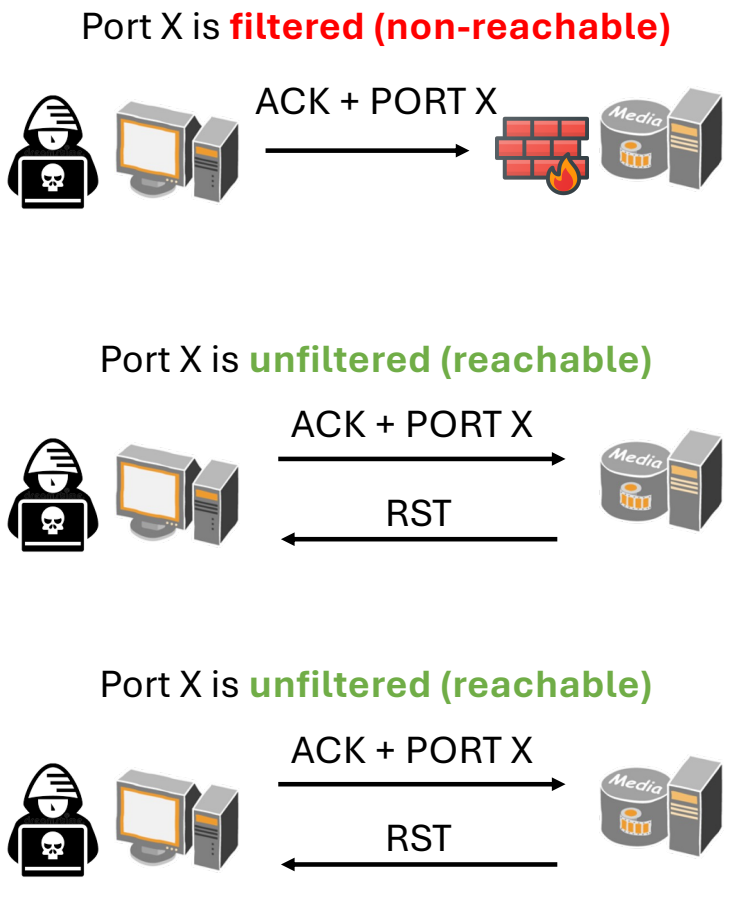
# Scanning Strategies

- 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.



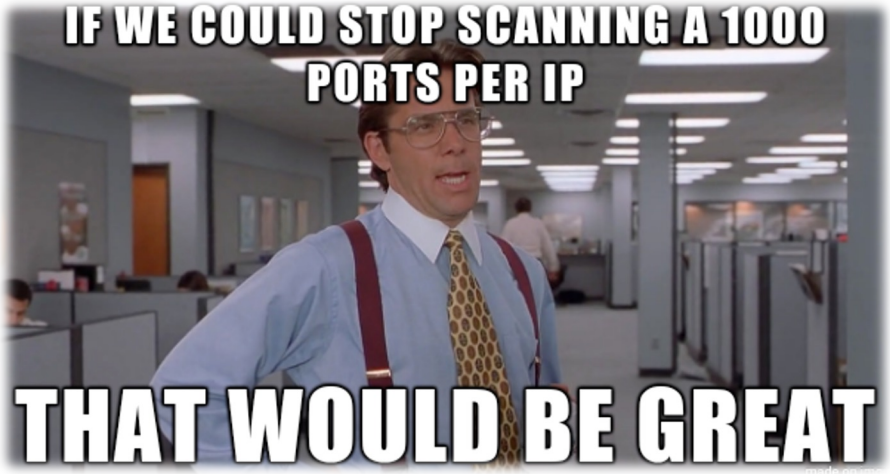
# Scanning Strategies

- 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.



# Other Scanning Strategies

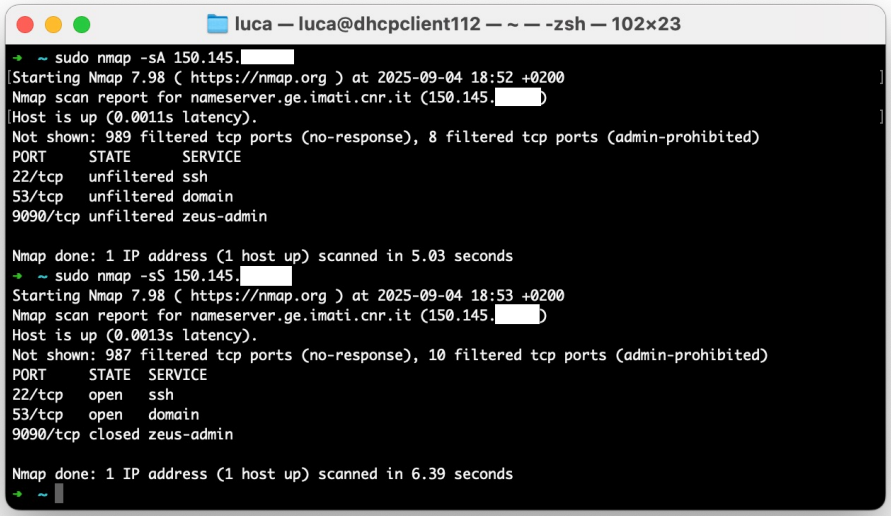
- UDP scanning:
  - if an UDP packet is sent to a closed port, some OSes respond with an ICMP port unreachable message
  - application-specific UDP packets are sent hoping to generate an application layer response.




r/networkingmemes

# Network Scanning: Nmap

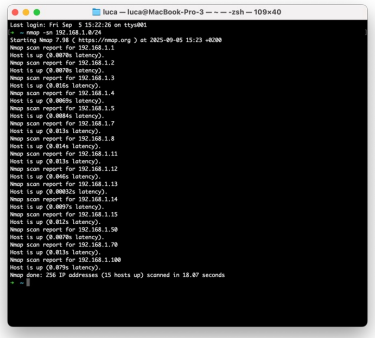
- Nmap is a free and network scanner:
  - many functionalities for probing hosts , fingerprint nodes and discover services
  - *https://nmap.org*
  - on macOS: *brew install nmap*
  - give it a try!
- The tool:
  - is great for security auditing
  - can monitor host and services
  - cab check various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




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.



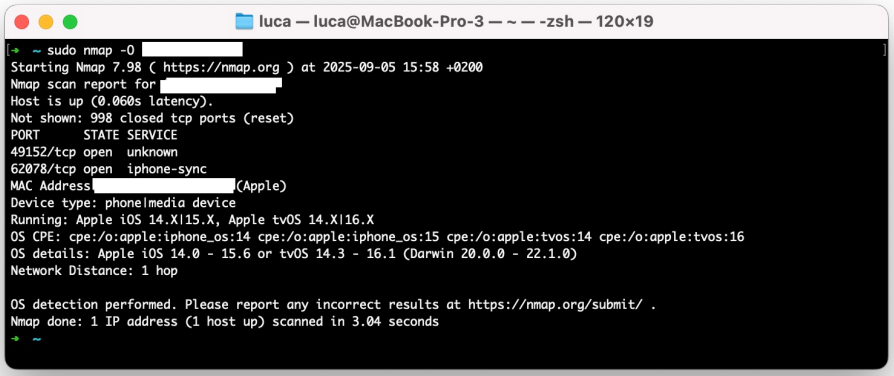
Active hosts

- 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.

Active hosts and opened ports

# Example: Scanning with Nmap

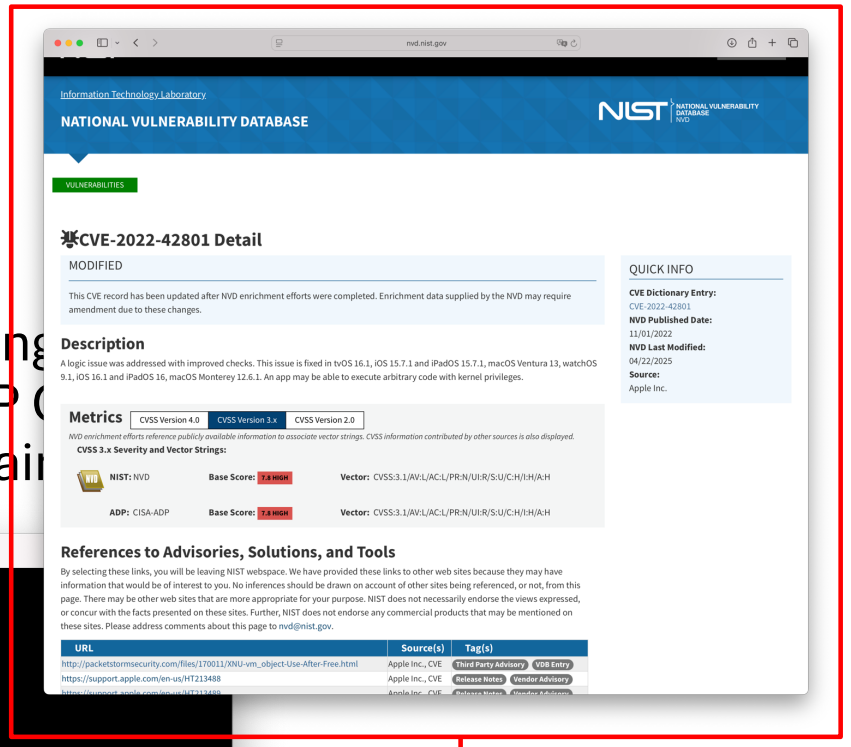
- Perform OS fingerprint:
  - *nmap –O host*
  - requires sudo!
  - **explanation**: the OS is “guessed” by matching responses from probes against a DB. Typical traits analyzed are: TCP Options, Window Sizes, initial value for the TTL, order of Flags, reaction against specific ICMP stimuli.

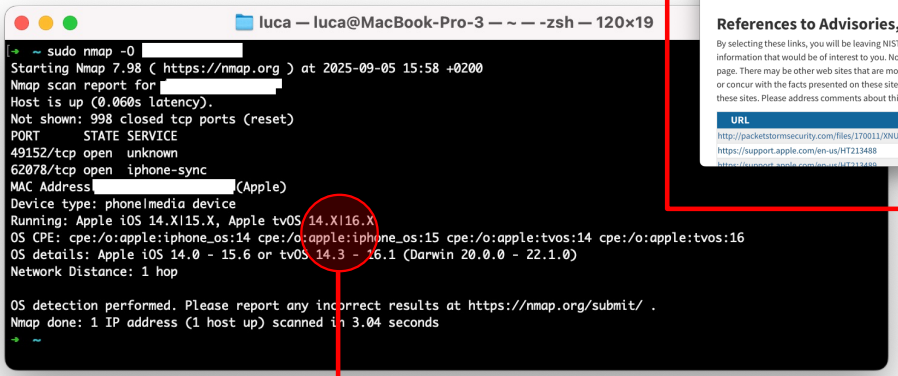



Correct guessing of my iPhone

# 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 value for the TTL, order of Flags, reaction agai



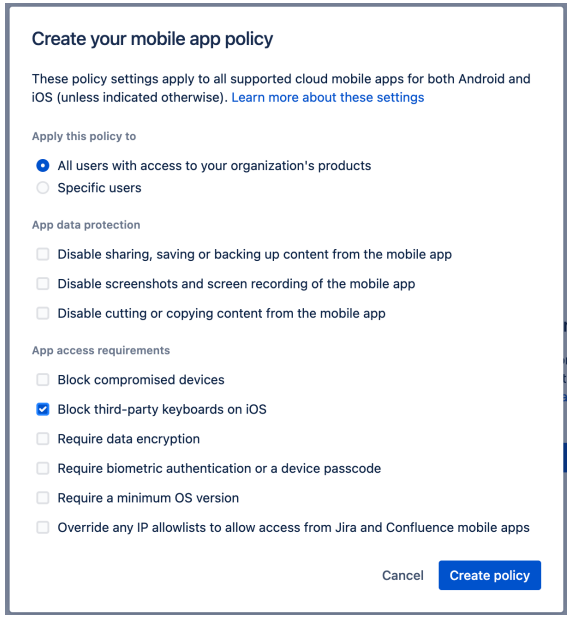

Correct guessing of my iPhone

# 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
  - …



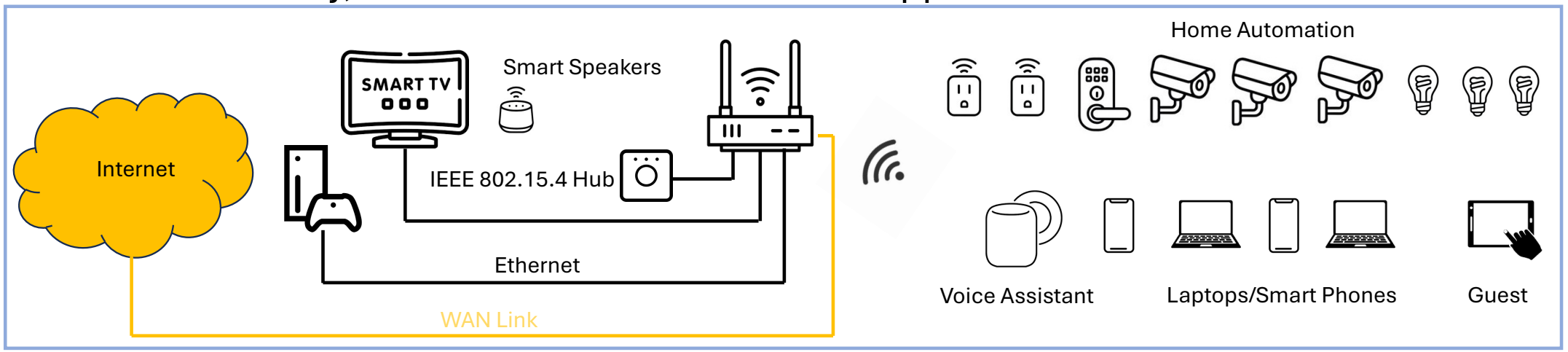

**Example: Atlassian**: https://support.atlassian.com/security-and-access-policies/docs/mobile-policy-mam-security-controls-and-supported-apps/

# Network Segmentation

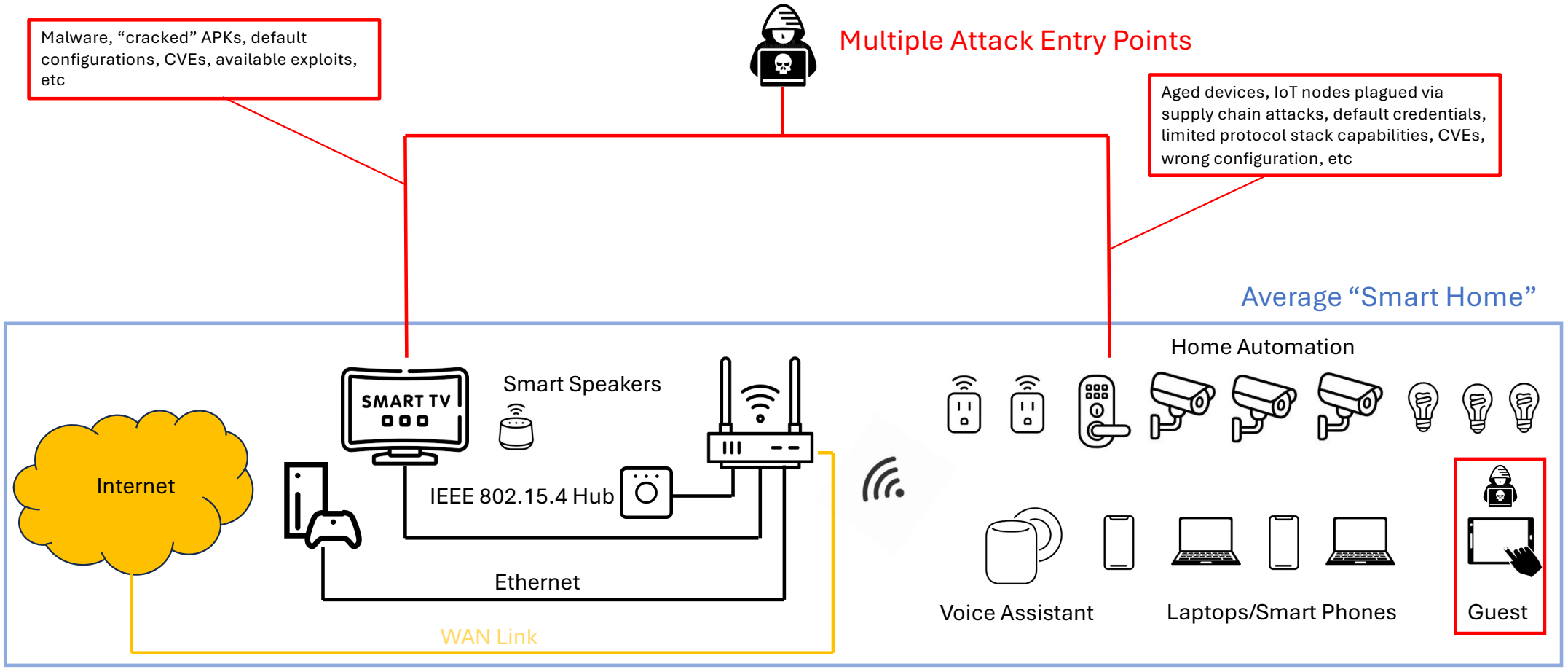
- Network segmentation divides a larger network into smaller and isolated sections:
  - sections are often called zones
  - also known as partitioning
  - isolation improves security and performance
  - basically, is an attack surface reduction approach.

# Network Segmentation

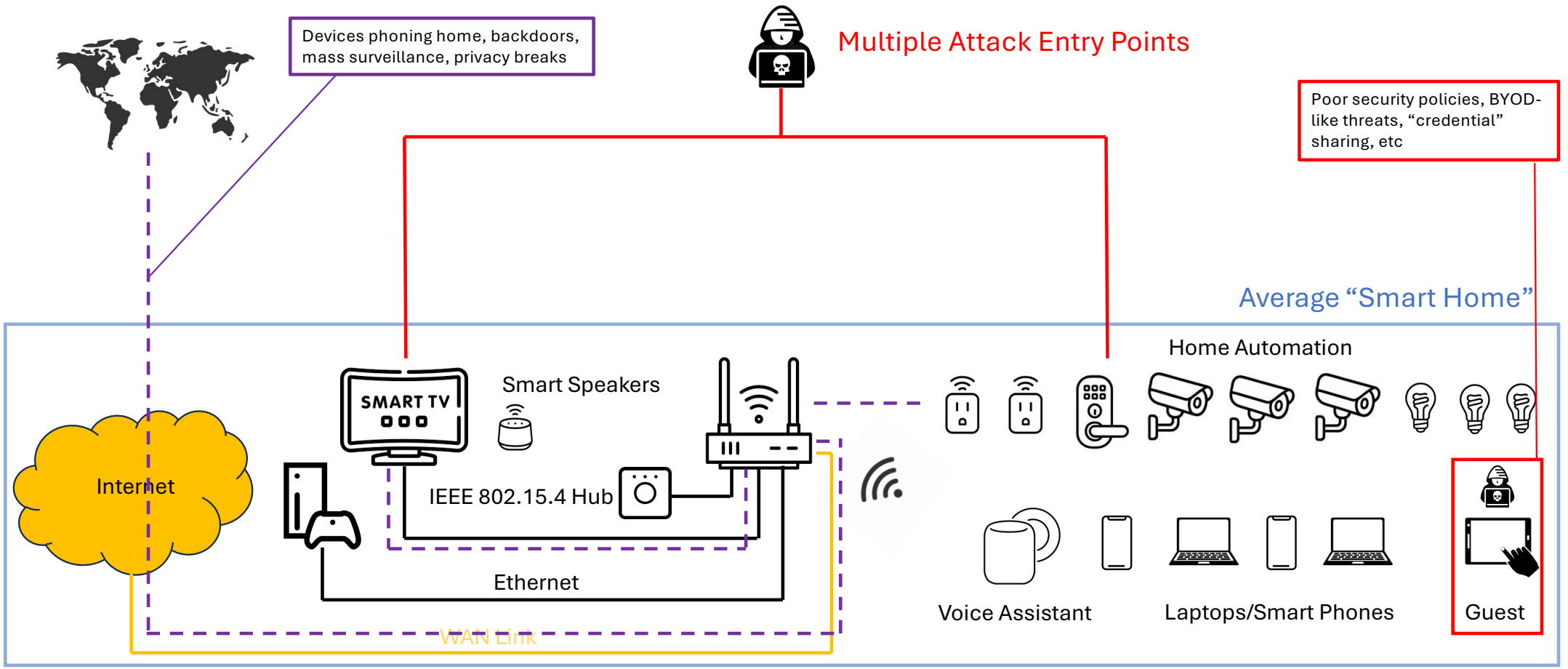
- Network segmentation divides a larger network into smaller and isolated sections:
  - sections are often called zones
  - also known as partitioning
  - isolation improves security and performance
  - basically, is an attack surface reduction approach.



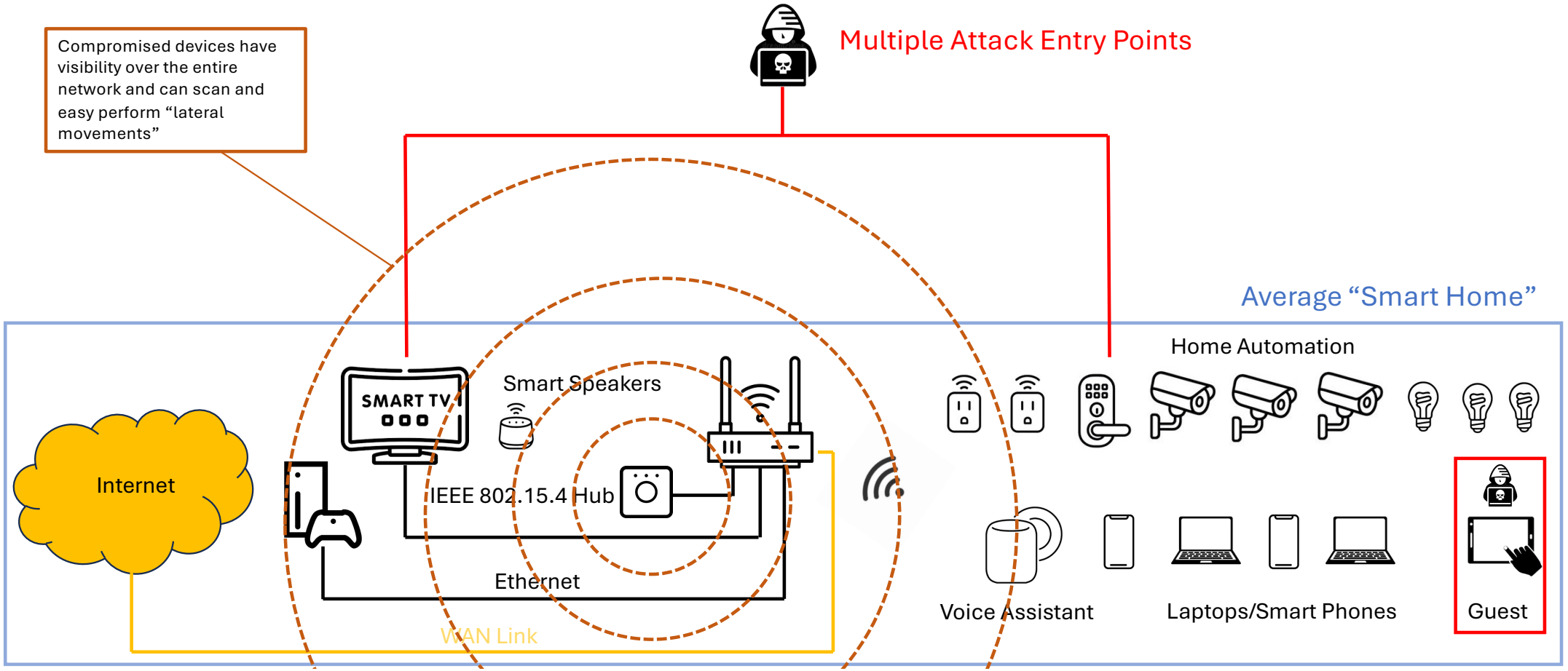
# Network Segmentation



# Network Segmentation



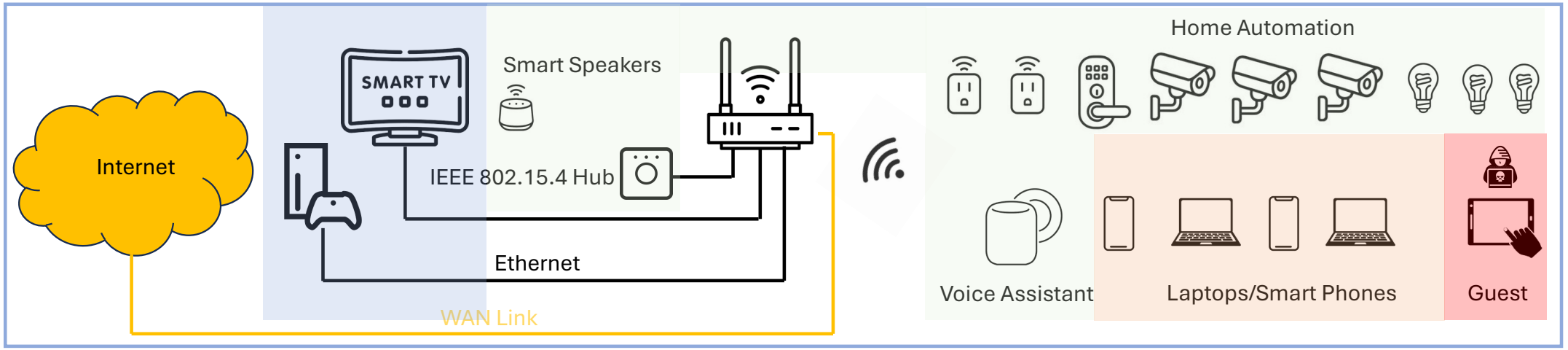
# Network Segmentation



# Network Segmentation

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

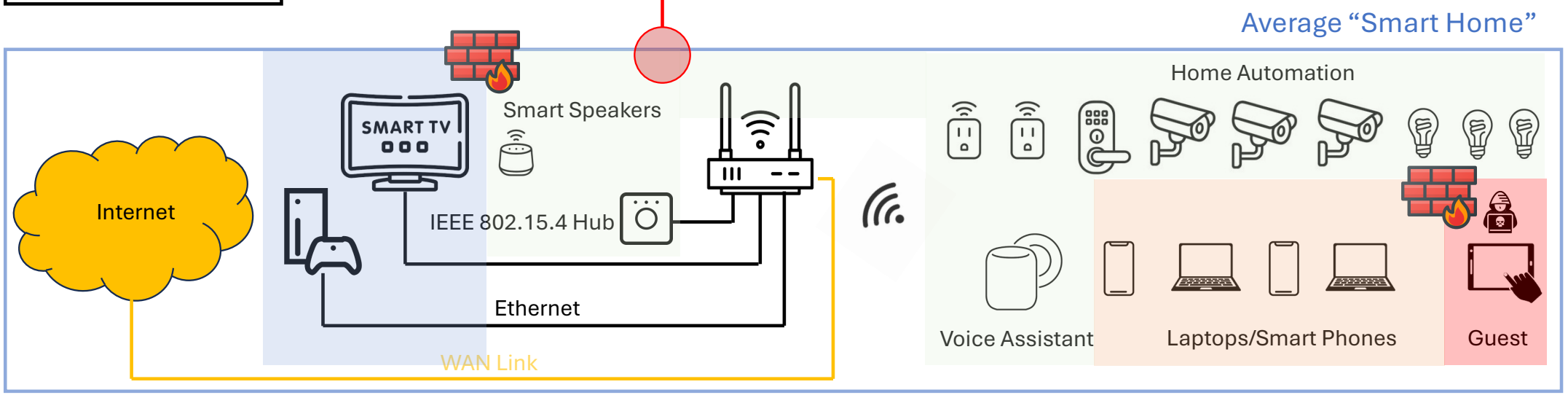
Average “Smart Home”



# Network Segmentation

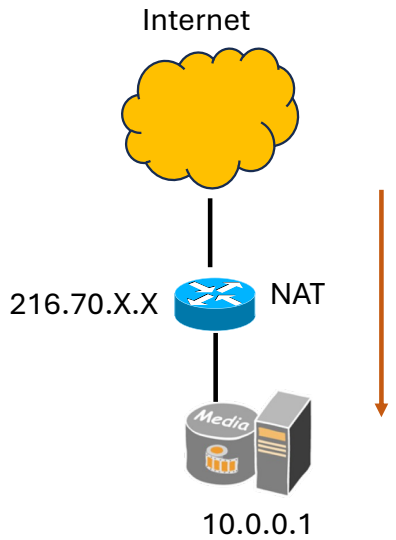
VLANs/Subnets
Firewall Rules
Ad-hoc SSIDs
Access Control List
MAC Filtering

**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

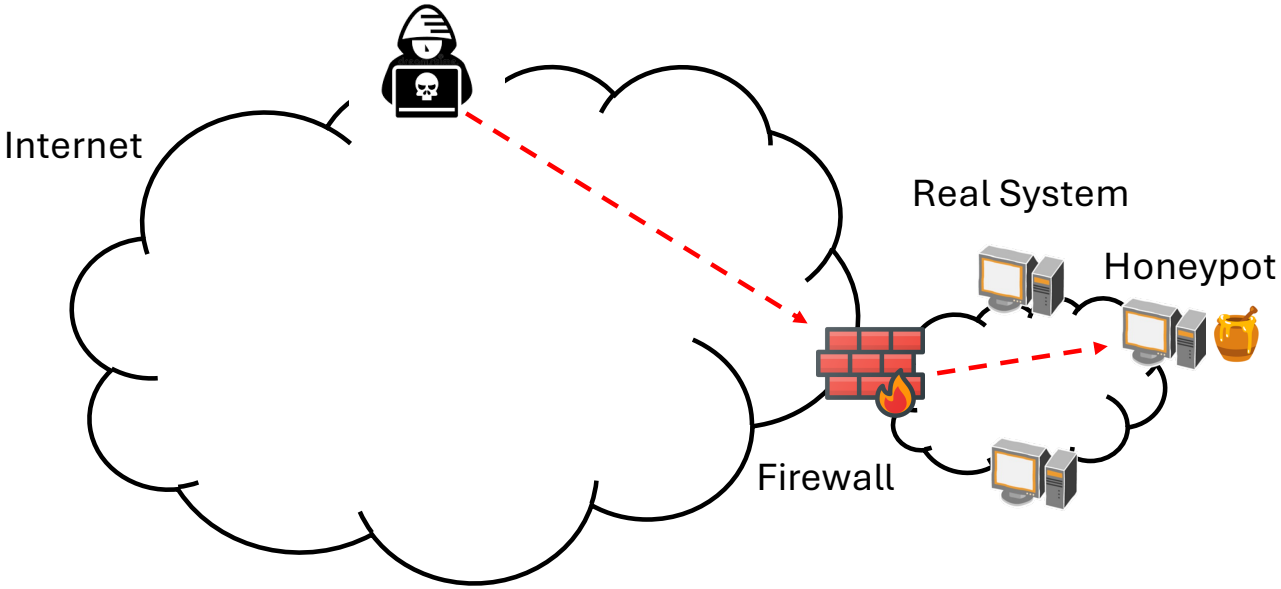
- Network Address Translation (NAT):
  - allows multiple devices on a private network to share the public IP addresses:
  - mainly designed for handling the shortage of IPv4 addresses
  - it can be used to improve security of a network .
- Several types of NATs:
  - out of scope in this course
  - issues 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.

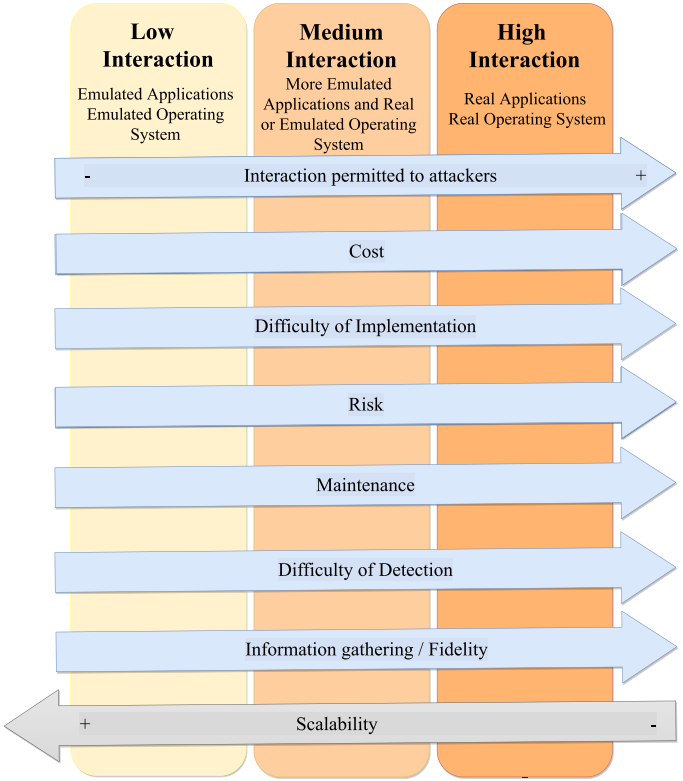
# Honeypot

- 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.

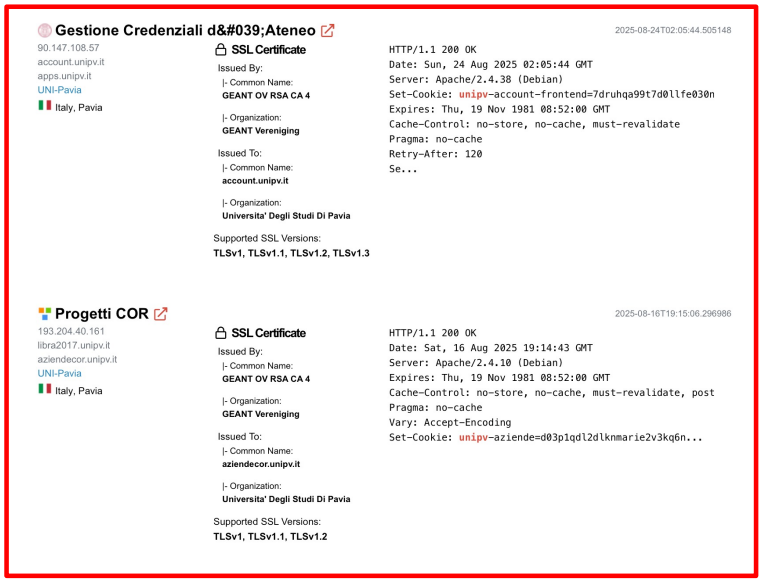



Interaction taxonomy of Franco et al.

J. Franco, A. Aris, B. Canberk, A. S. Uluagac, "A Survey of Honeypots and Honeynets for Internet of Things, Industrial Internet of Things, and Cyber-Physical Systems", IEEE Communications Surveys & Tutorials, Vol. 23, No. 4, pp. 2351-2383, Fourth Quarter 2021.

# 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.




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.