Enrico RUSSO

Università di Genova

Network Fundamentals

CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

https://cybersecnatlab.it

License & Disclaimer

License Information

This presentation is licensed under the Creative Commons BY-NC License

To view a copy of the license, visit:

http://creativecommons.org/licenses/by-nc/3.0/legalcode

Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided “as is” without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

Goal

- Present the basic definitions of computer networks
- Present the ISO/OSI model and TCP/IP protocol
- Present layers 2, 3, 4 and examples

Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

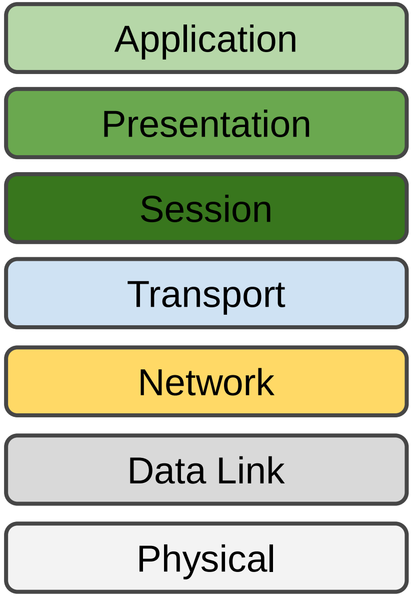
Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

ISO/OSI and TCP/IP

- ISO/OSI and TCP/IP represent the reference models for communication between different computers in the network. They both use a **layered** model.
  - Separate networking functions into logical smaller pieces: network problems can more easily be solved through a **divide-and-conquer** methodology.
  - Provide **modularity** and **clear interfaces**: they allows the standardization of interactions among devices.
  - Allow **extensibility**: new network functions are generally easier to add to a layered architecture.
- ISO/OSI model evolved as a **theoretical** model.
- TCP/IP as a **practical** model, founded on widely used implementation of network functions.

OSI Layers



Application

Presentation

Session

Transport

Network

Data Link

Physical

The Open Systems Interconnection (OSI) represents a guideline for network protocol design.

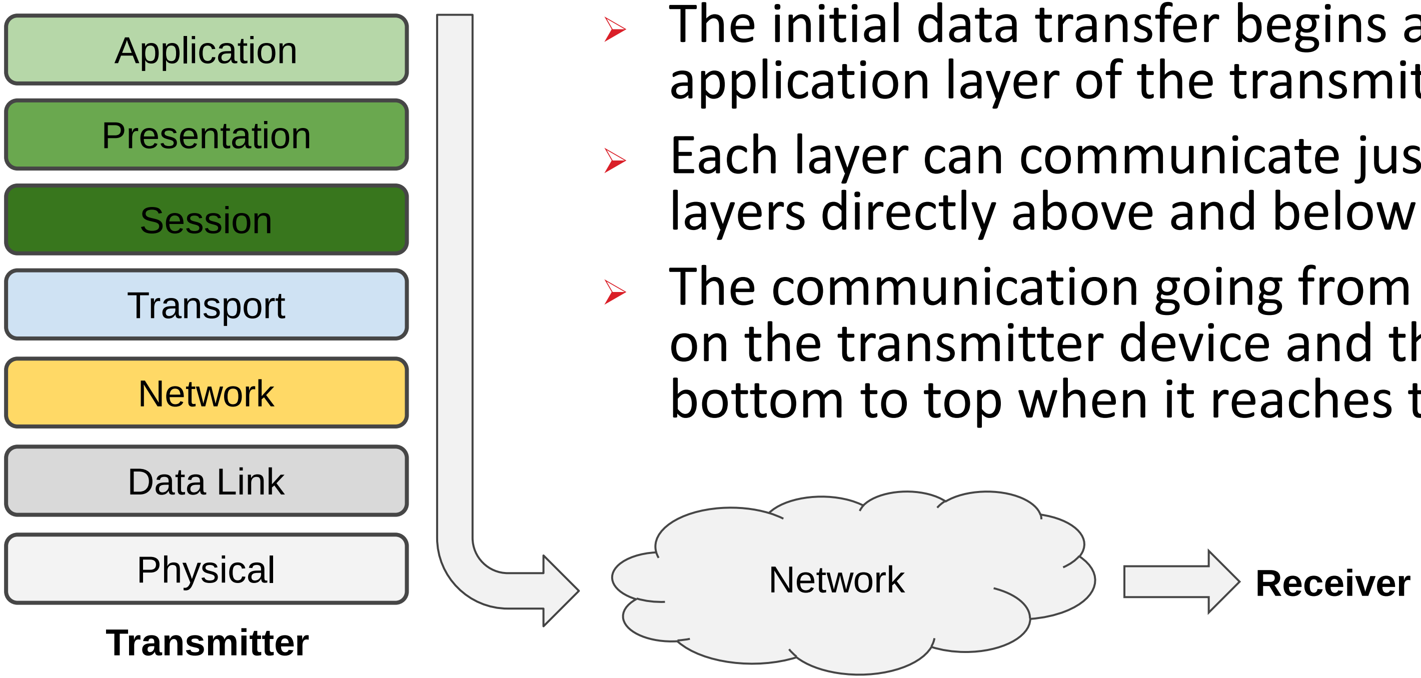
- A standard of the International Organization for Standardization (ISO)
- Seven layers

OSI Layers

Application	It provides the services to the user
Presentation	It is responsible for the formatting of information (e.g., compression and encryption)
Session	It is responsible for establishing, managing, and terminating sessions
Transport	It provides message delivery from process to process
Network	It is responsible for moving the packets form source to destination
Data Link	It combines bits into a structure of data and provides their error-free transfer
Physical	It provides a physical medium through which bits are transmitted

© CINI – 2021

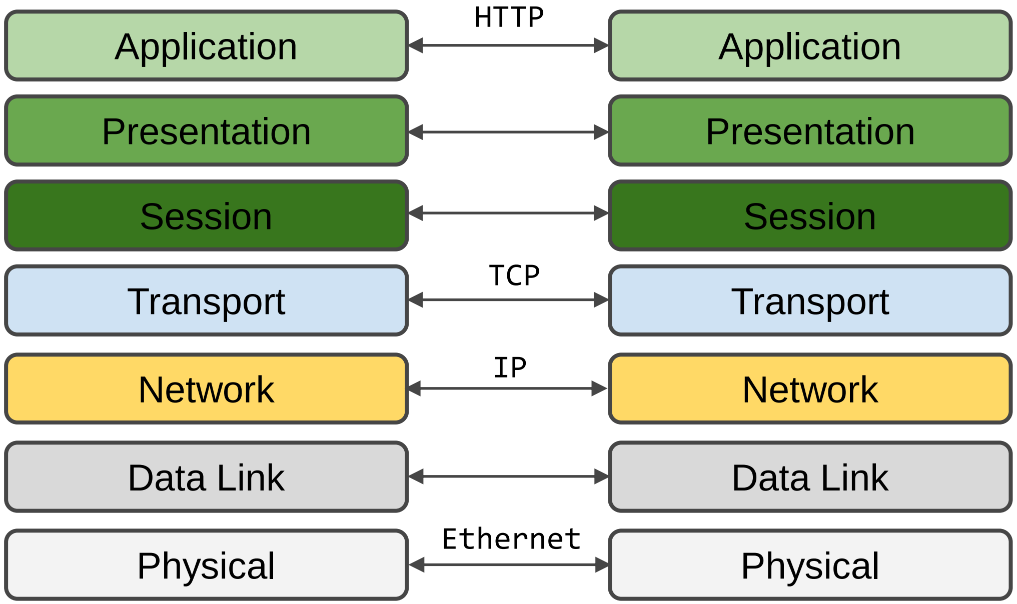
OSI Layers: data transfer



- The initial data transfer begins at the application layer of the transmitter
- Each layer can communicate just with the layers directly above and below it
- The communication going from top to bottom on the transmitter device and then from bottom to top when it reaches the receiver

© CINI – 2021

OSI Layers: protocols



Application HTTP Application

Presentation Presentation

Session Session

Transport TCP Transport

Network IP Network

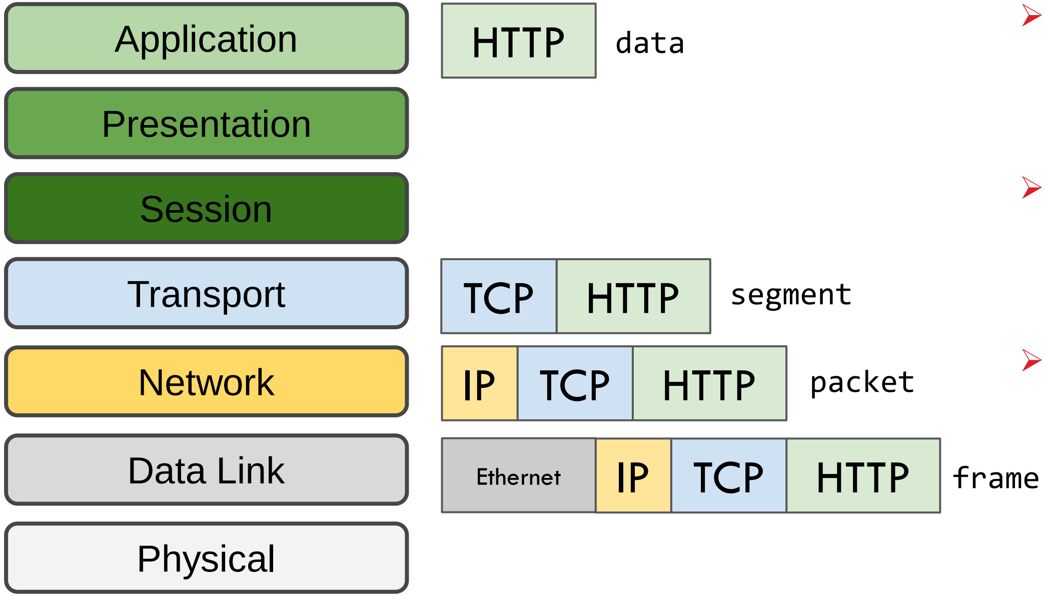
Data Link Data Link

Physical Ethernet Physical

- The model itself does not provide specific methods of communication
- Actual communication is defined by various *protocols*
- A protocol is a **standard procedure and format** that two data communication devices must understand, accept and use to be able to talk to each other

© CINI – 2021

OSI Layers: Protocols Data Unit (PDU)



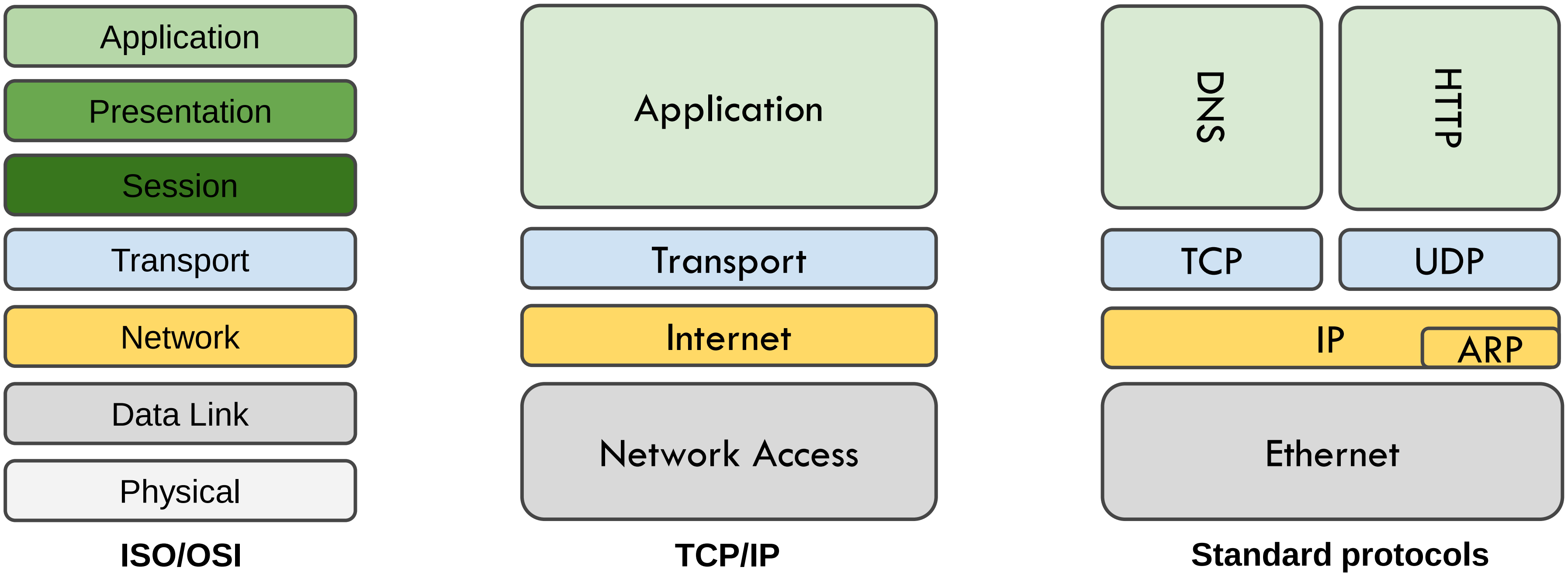
- The protocols at different layers exchange data with the aid of *data encapsulation*
- Each layer is responsible for adding a header or a footer to the data being transferred
- The encapsulation process creates a *Protocol Data Unit* (PDU), which includes the data being sent and all header or footer information added to it

TCP/IP

TCP/IP provides an alternative model used for the description of all network communications.

- is a four-layer model
- is based on standard protocols that the Internet has developed, and the name refers to the two widely used ones:
  - **Transmission Control Protocol** (TCP) which also implements the Transport layer of ISO/OSI model
  - **Internet Protocol** (IP) which also implements the Network layer of ISO/OSI model

TCP/IP model



© CINI – 2021

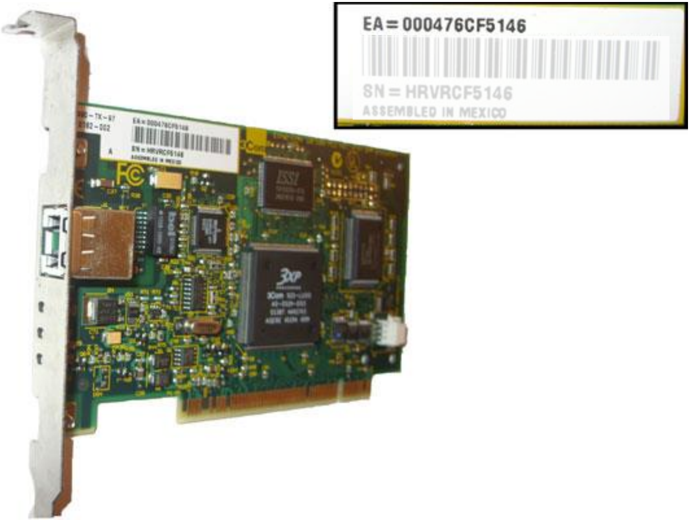
Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

Ethernet

Ethernet is a broadly deployed layer 2 protocol.

- Encapsulate data and transmit them in the form of *frames*
- Frames leverage the Media Access Control (MAC) addresses
  - 48 bits burned in the adapter ROM (first 3-bytes: the ID of the manufacturer*)
  - Every Ethernet device (e.g., a server, a switch, or a router) has a unique MAC address on its local network
  - A *Frame* includes the MAC address of the destination interface on the target system as well the MAC address of the source interface on the sending system



*https://www.wireshark.org/tools/oui-lookup.html

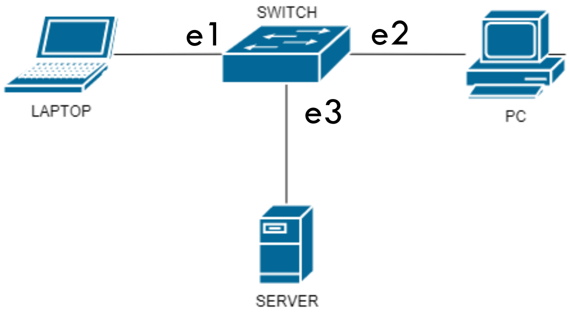
Bridges and Switches

Devices providing interconnectivity at Layer2 are called *(Transparent) Bridges or Switches.*

- They analyze all frames received, find the destination MAC address, and forward them to the appropriate port.
- To determine where to forward the traffic, they use a special table (MAC address table).

A basic switched network

- A switch device provides connection to a number of common devices.
- Let's assume that all the devices be powered on but have not sent any traffic.
- In this case, the MAC address table of the switch would be empty.

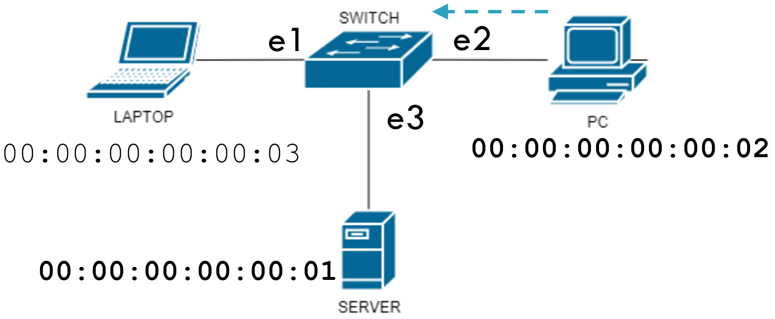


MAC address table (switch)

MAC address	Port

A basic switched network

- PC wants to send traffic to SERVER that has MAC address 00:00:00:00:00:01
  - Creates a frame containing 00:00:00:00:00:02 as the source address and 00:00:00:00:00:01 as the destination address.
  - Sends it off toward the switch.



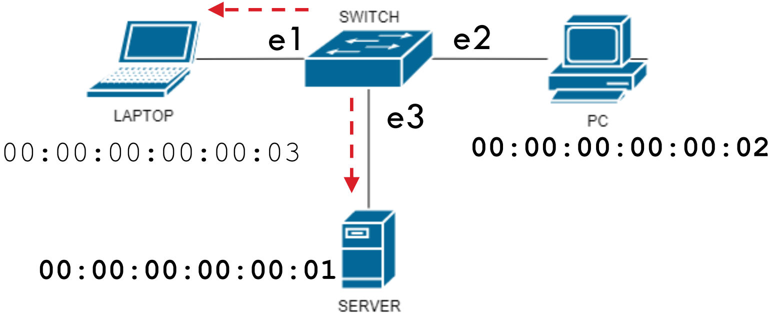
MAC address table (switch)

MAC address	Port

© CINI – 2021

A basic switched network

- The switch receives the traffic
  - Creates a new entry in its MAC address table for PC MAC address (PC → e2)
  - Performs a lookup on its MAC address table to determine whether it knows which port to send the traffic to
  - Since no matching entries exist in the switch's tables, it would **flood** the frame out all of its interfaces except the receiving port (**broadcast**).

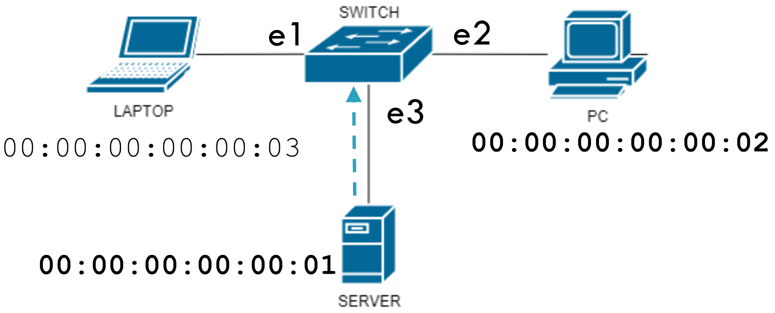


MAC address table (switch)

MAC address	Port
00:00:00:00:00:02	e2

A basic switched network

- The broadcast forwards the frame also to the target server.
- (Assuming that the server wants to respond to PC) It sends a new frame back toward the switch containing `00:00:00:00:00:01` as the source address and `00:00:00:00:00:02` as the destination address.
- The switch would receive the frame and create a new entry in its MAC address table for the Server MAC address (Server → e3).



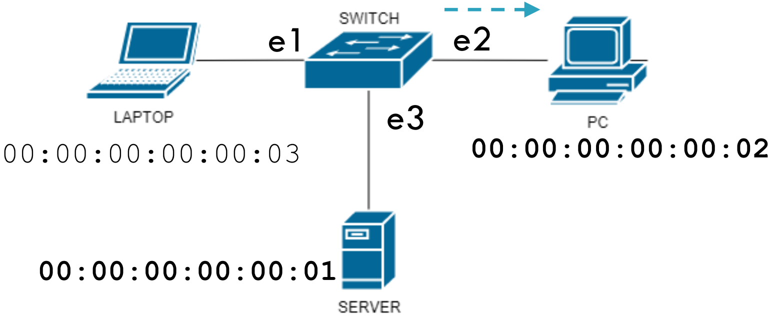
MAC address table (switch)

MAC address	Port
00:00:00:00:00:02	e2
00:00:00:00:00:01	e3

© CINI – 2021

A basic switched network

- Switch performs a lookup of its MAC address table to determine whether it knows which port to send the server frame to.
- In this case, it does, so it sends the return traffic out only its e2 port (PC), without flooding.

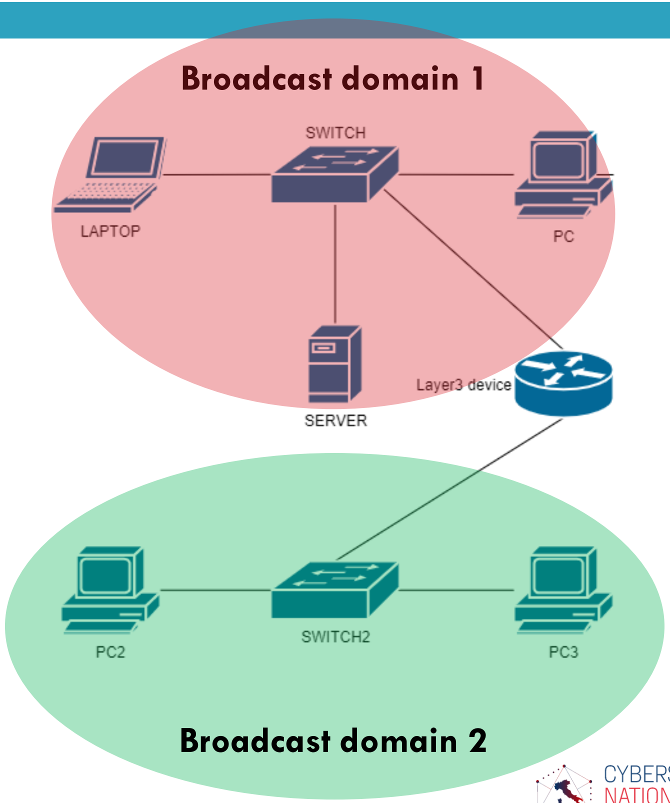


MAC address table (switch)

MAC address	Port
00:00:00:00:00:02	e2
00:00:00:00:00:01	e3

Broadcast domains

- Switching relies on broadcasts.
- All network nodes that can be reached at Layer 2 share the same **broadcast domain**.
- Layer 3 devices form boundaries between these domains.



© CINI – 2021

Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

Internet Protocol (IP)

The most significant protocol at layer 3 is the *Internet Protocol* or IP

- The standard for routing packets across interconnected networks (hence, the name internet)
- Encapsulate data and pass that data in the form of *packets*

© CINI – 2021

IP addressing

- An Internet Protocol address is also known as an **IP address**.
- A numerical label which assigned to each device connected to a computer network that uses the IP for communication.
- Two versions: IPv4 and IPv6
  - IPv6 is the new version that is being deployed to fulfill the need for more Internet addresses.
  - In this module, we focus on IPv4 (currently the most widely used).

IP addressing

- IPv4 address
  - 32 bits
  - Grouped 8 bits at a time (octet)
  - Each of the four octets is separated by a dot and represented in decimal format (dotted decimal notation)

```
11000000 10101000 01100100 11001000

  192   .   168   .   100   .   200
```

IP addressing - Home addressing

2^{32}

IPv4
available
addresses

IP addressing - Home addressing

192.168.100.200
(host address)

35
(house number)

2^{32}

IPv4
available
addresses

IP addressing - Home addressing

192.168.100.200
(host address)

↑

192.168.100
(network address)

↑

2^{32}

IPv4
available
addresses

35
(house number)

↑

via Dodecaneso
(street name)

↑

IP address and Netmask

- An IP address has two components: a *network* component (street name), and a *host* component (house number)
- The purpose of the *netmask* is to split the IP address into the two components
- When you combine, using a logical AND, the IP address and the netmask you reveal the network component

11000000 10101000 01100100 11001000 address

192 . 168 . 100 . 200

11111111 11111111 111111111 00000000 netmask (***/*****24**)

255 . 255 . 255 . 0

network 11000000 10101000 01100100 00000000 host

192 . 168 . 100

© CINI – 2021

Reserved IP addresses

- In every network, two addresses are used for special purposes. <u>These addresses are not available for nodes</u>
- **Network address**: is the first address in the network (all the host bits are 0) and it is used for identifying the network
- **Broadcast address**: is the last address in the network (all the host bits are 1). An IP packet having the broadcast address as the destination address is sent to all nodes of the IP network

11000000 10101000 01100100 11001000 address

192 . 168 . 100 . 200

11111111 11111111 111111111 00000000 netmask (**/24**)

255 . 255 . 255 . 0

11000000 10101000 01100100 **00000000** network address

192 . 168 . 100 . 0

11000000 10101000 01100100 **11111111** broadcast addr.

192 . 168 . 100 . 255

Default Netmasks

- Default netmasks have all ones (255) or all zeroes (0) in an octet

Address Class	Total # Of Bits For Network ID / Host ID	Default Subnet Mask			
Class A	8/24	255	0	0	0
Class B	16/16	255	255	0	0
Class C	24/8	255	255	255	0

Non-default Netmasks (example)

- 192.168.100.x/**25**, 7 bits for hosts ⇒ 126 addresses + network addr. + bcast addr.
- **first** network: 192.168.100.0-127
  - 192.168.100.0: network address
  - 192.168.100.1: first host
  - 192.168.100.126: last host
  - 192.168.100.127: broadcast address
- **second** network: 192.168.100.128-255
  - 192.168.100.128: network address
  - 192.168.100.129: first host
  - 192.168.100.254: last host
  - 192.168.100.255: broadcast address

```
  192   .   168   .   100
11000000 10101000 01100100 00000000
11000000 10101000 01100100 00000001
11000000 10101000 01100100 01111110
11000000 10101000 01100100 01111111

11000000 10101000 01100100 10000000
11000000 10101000 01100100 10000001
11000000 10101000 01100100 11111110
11000000 10101000 01100100 11111111
```

Private IP addresses

Private IP addresses are **not routed on the Internet,** and traffic cannot be sent to them from the Internet

- They are supposed to work within the local network, only.
  - Range from 10.0.0.0 to 10.255.255.255 — a 10.0.0.0 network with a 255.0.0.0 or an /8 (8-bit) mask
  - Range from 172.16.0.0 to 172.31.255.255 — a 172.16.0.0 network with a 255.240.0.0 (or a 12-bit) mask
  - A 192.168.0.0 to 192.168.255.255 range, which is a 192.168.0.0 network masked by 255.255.0.0 or /16

Neighbor Table and Address Resolution Protocol (ARP)

- An IP node wants to communicate with a system in the same layer 2 domain
  - It looks in its neighbor table, or **ARP table** (IP → MAC), to determine how to construct the Ethernet frame.
  - If the desired destination IP address is not in the ARP table, the node issues an ARP request, which is **broadcasted** to everyone in the layer 2 domain, that asks “*Please tell me the MAC address for the node with IP address X.X.X.X.*”.
  - Assuming the target device is available, the node with that IP address will respond.
  - An ARP request for a non-existing host takes a fixed number of retries (after a timeout) before concluding that the host isn't reachable.

IP Routing

- IP routing is the process of sending packets from a host on one network to another host on a different remote network
  - Nodes examine the destination IP address of a packet, determine the next-hop address, and forward the packet
  - Nodes use **routing tables** to determine a next hop address to which the packet should be forwarded

Router

- A router is the Layer 3 device that forwards data packets between computer networks.
- A router is connected to two or more data lines from different IP networks.

Internetworking: Routing Table

A routing table is used by nodes to determine the path to the destination network

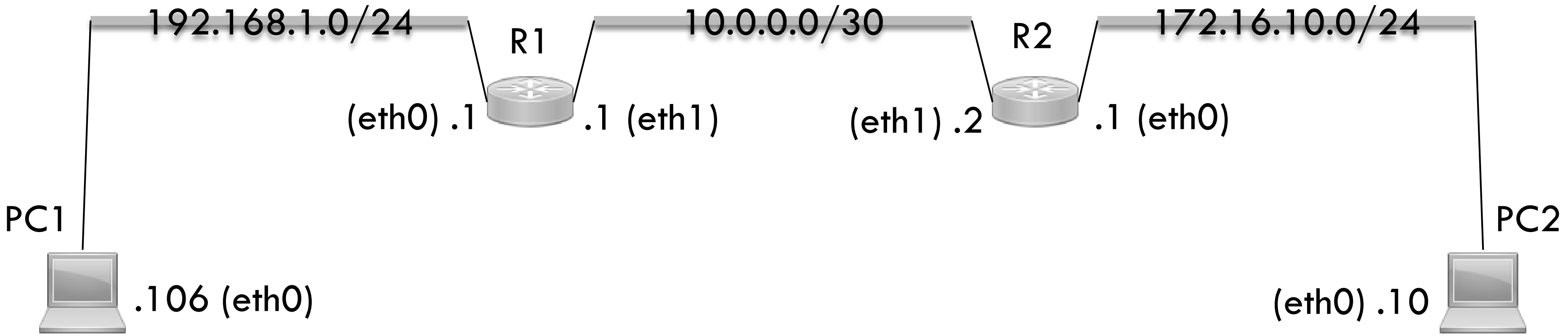
- Each routing table consists of the following entries:
  - **Network destination and subnet mask** – specifies a range of IP addresses
  - **Remote router** – IP address of the router used to reach that network
  - **Outgoing interface** – outgoing interface the packet should go out to reach the destination network

© CINI – 2021

Connected, static and default routes

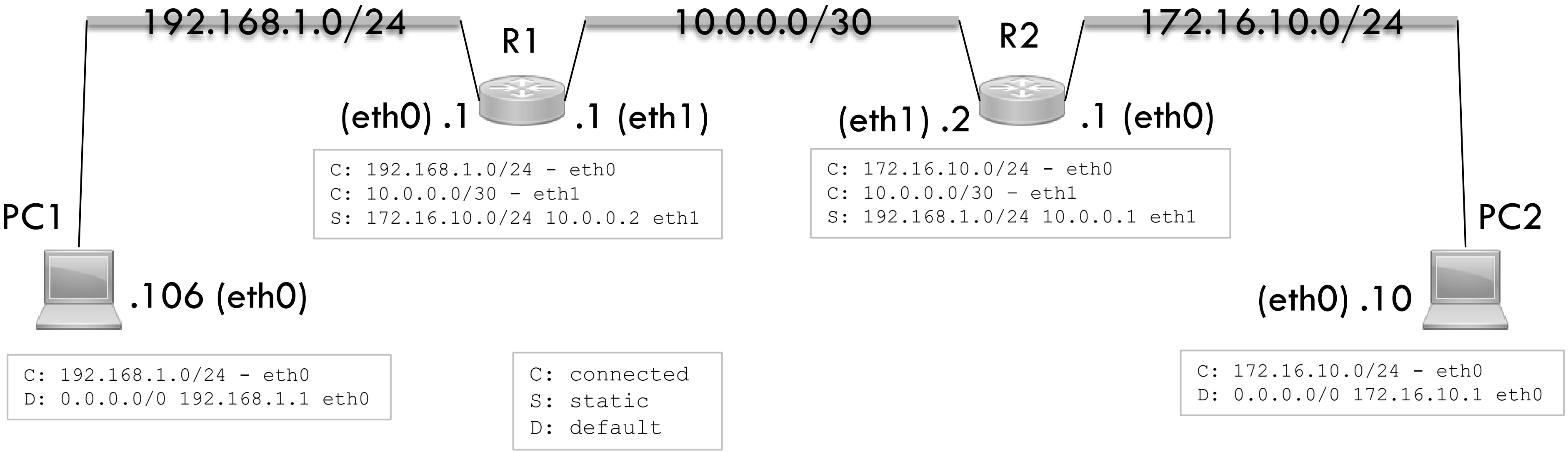
- Routing table entries can originate from the following sources:
  - **connected**: subnets directly connected to a node’s interface are added to the node’s routing table (interface has to have an IP address configured and must be in the up state)
  - **static**: by adding static routes, a node can learn a route to a remote network that is not directly connected to one of its interfaces. Static routes are configured manually specifying DESTINATION_NETWORK SUBNET_MASK NEXT_HOP_IP_ADDRESS
  - **default**: a forwarding rule for packets when no specific address of a next-hop host is available from the routing table

Routing tables (example)



© CINI – 2021

Routing tables (example)



```
C: 192.168.1.0/24 - eth0
C: 10.0.0.0/30 - eth1
S: 172.16.10.0/24 10.0.0.2 eth1
```

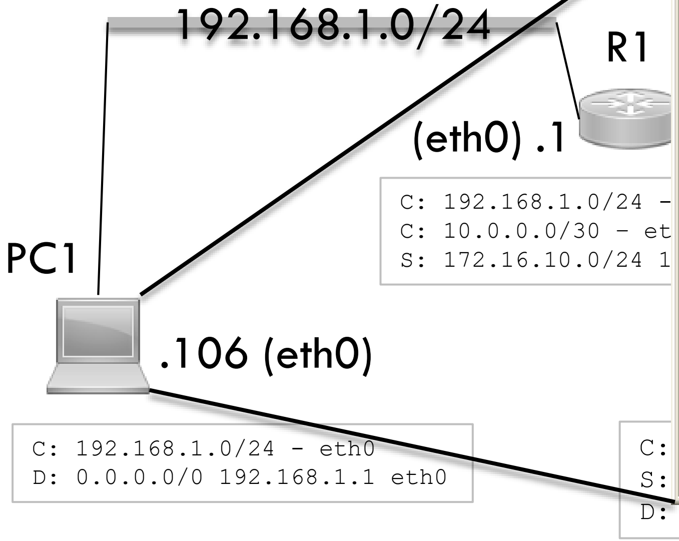
```
C: 172.16.10.0/24 - eth0
C: 10.0.0.0/30 - eth1
S: 192.168.1.0/24 10.0.0.1 eth1
```

```
C: 192.168.1.0/24 - eth0
D: 0.0.0.0/0 192.168.1.1 eth0
```

```
C: connected
S: static
D: default
```

```
C: 172.16.10.0/24 - eth0
D: 0.0.0.0/0 172.16.10.1 eth0
```

Routing tables (example)

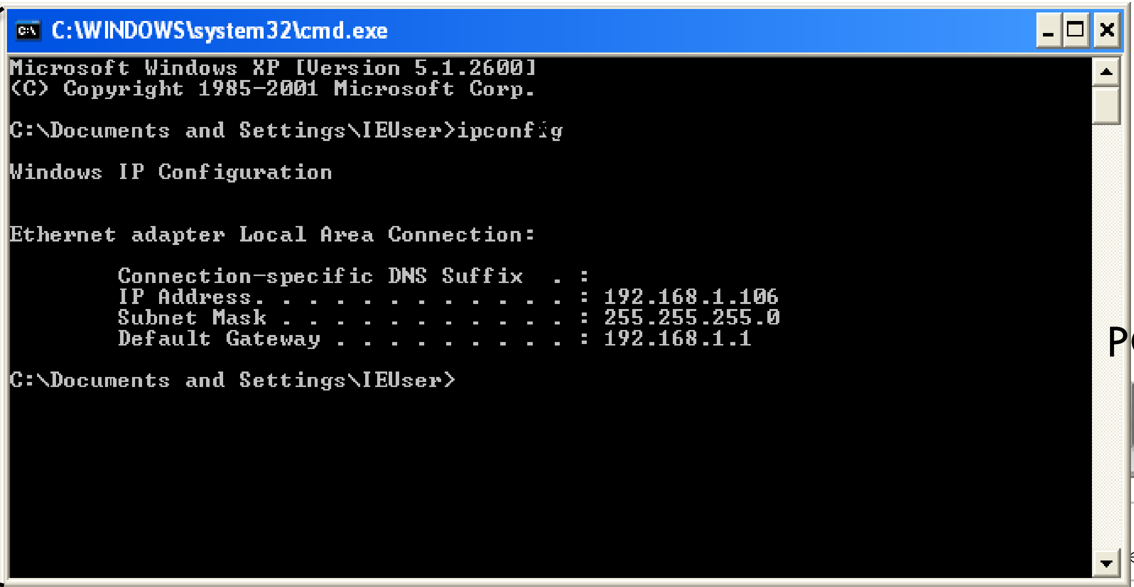


```
C: 192.168.1.0/24 –
C: 10.0.0.0/30 – et
S: 172.16.10.0/24 1
```

```
C: 192.168.1.0/24 – eth0
D: 0.0.0.0/0 192.168.1.1 eth0
```

```
C:
S:
D: default
```

```
eth0
```



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\IEUser>ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.106
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\Documents and Settings\IEUser>
```

Outline

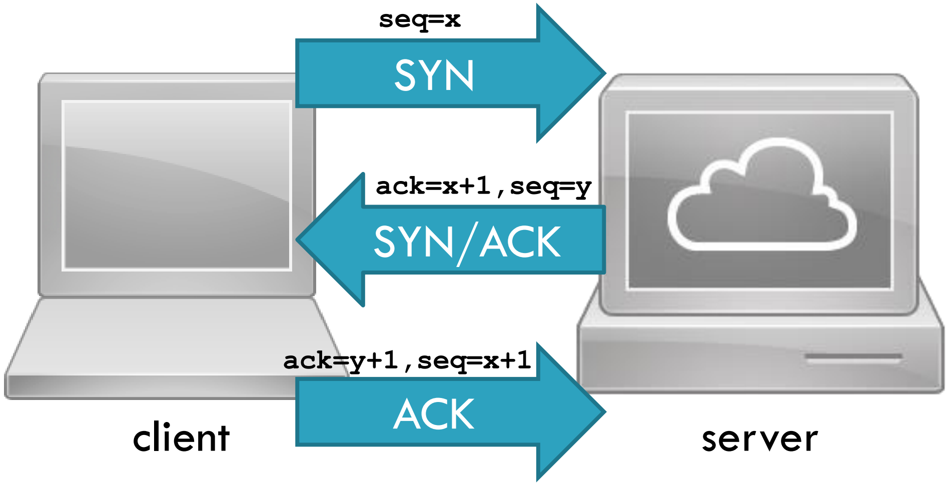
- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

TCP vs UDP

- TCP and UDP are the most common Layer 4 protocols
  - TCP first creates a connection before any message is sent, whereas UDP does not
  - While both do error checking by checksums, UDP won't recover from one. TCP includes error recovery, thanks to acknowledgments
  - TCP rearranges data packets in the specific order while UDP protocol has no fixed order
  - Since UDP has no connection establishment, no connection state, and small packet header overhead is simpler and faster than TCP
  - UDP is commonly used for applications that are “lossy” (can handle some packet loss), such as streaming audio and video.

Three-Way Handshake

- TCP uses a *three-way handshake* to establish a reliable connection
- The use of sequence (seq) and acknowledgment (ack) numbers allows both sides to detect missing or out-of-order segments



© CINI – 2021

Layer 4 addressing: ports

- Layer 4 is in charge of the **process-to-process** communication. Transmitter and receiver are identified using **ports**
  - 16-bit unsigned integer (0-65535, 0 reserved)
    - **Well-known ports** (0-1023): used by system processes that provide widely used types of network services (require superuser privileges)
    - **Registered ports** (1024-49151): assigned by a central authority (the Internet Assigned Numbers Authority, IANA) for specific services
    - **Ephemeral ports** (49152–65535): contain dynamic or private ports that cannot be registered by IANA

Layer 4 addressing: ports

- The use of well-known and registered ports allows the requesting process to easily locate the corresponding server application processes on other hosts
  - For example, a web browser knows that the web server process listens on port 80/TCP
- Despite these agreements, <u>any service can listen on any port</u>
  - For example, a web server process can listen on port 8080/TCP instead of the well-known one.

Outline

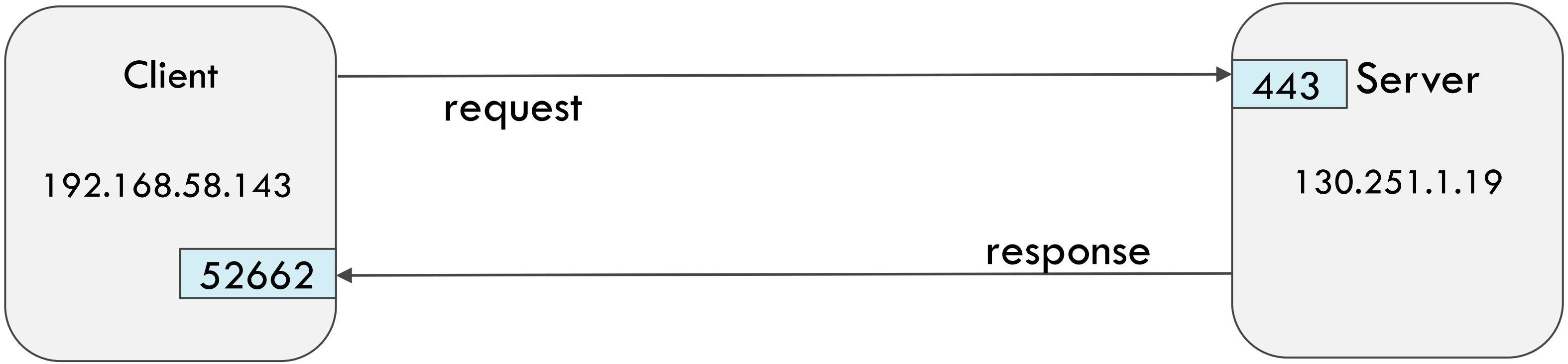
- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

The client-server model

- TCP/IP relies on the **client-server** model for enabling the process communication between network nodes.
  - It is a relationship in which one program (*client*) requests a service or resource from another program (*server*).
  - The client needs to know of the existence of and the address of the server.
  - The server does not need to know the address of (or even the existence of) the client prior to the connection being established.

The client-server model (example)

Source IP	Source Port	Destination IP	Destination Port	Data
192.168.58.143	52662 (ephemeral)	130.251.1.19	443 (well-known)	https://...



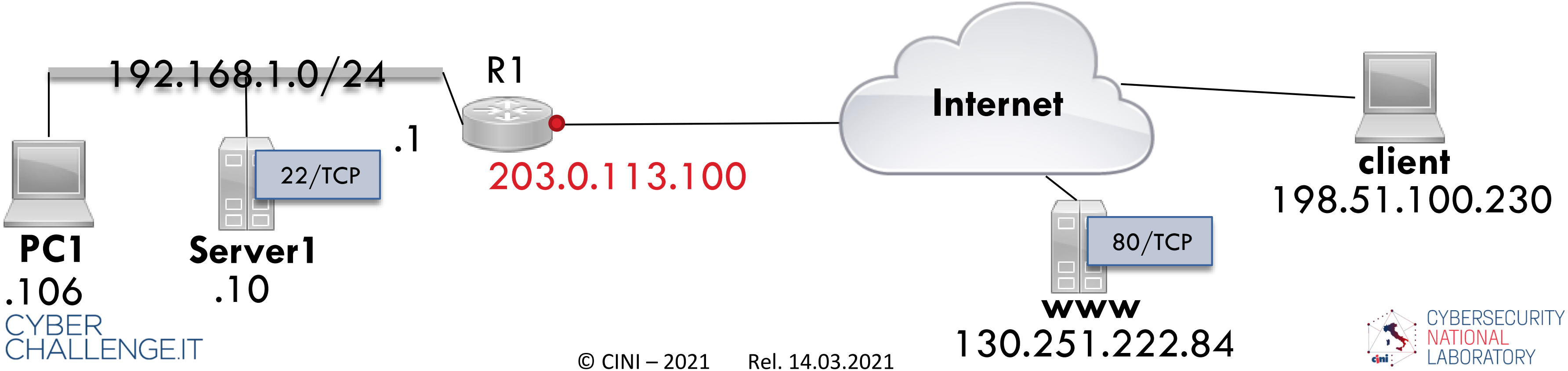
Data	Destination Port	Destination IP	Source Port	Source IP
	52662 (ephemeral)	192.168.58.143	443 (well-known)	130.251.1.19

Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- **Network Address Translation**
- Application layer examples: DNS and HTTP

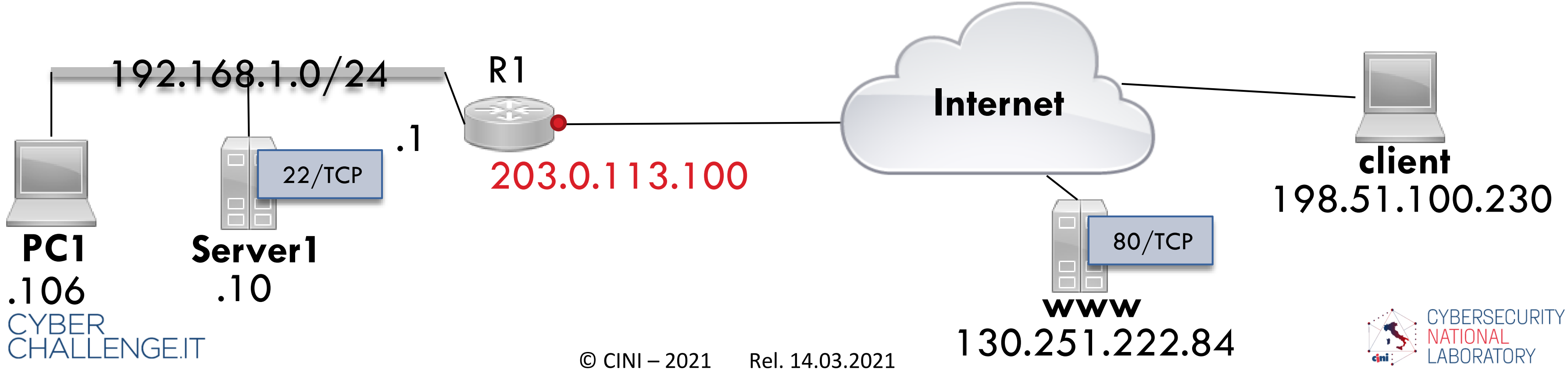
Network Address Translation

- Network Address Translation (NAT) generally involves rewriting the source and/or destination addresses of IP packets as they pass through a router or firewall
  - 192.168.1.0/24 is a private network and it is <u>not routable</u> on the Internet



Source NAT and Masquerade

- Masquerade is a **source NAT** rule, i.e., it is related to the source address of a packet
- The popular usage of NAT Masquerade is to translate a private address range to a single public IP address



Source NAT and Masquerade (example)

- PC1 and Server1 accessing www (request)

SNAT table (dynamic)

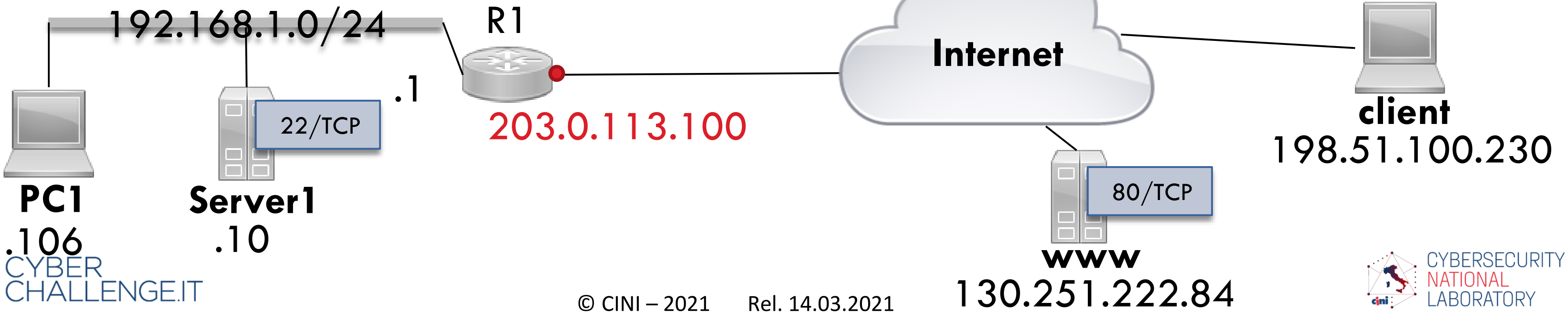
203.0.113.100	52000,80	192.168.1.106
203.0.113.100	53000,80	192.168.1.10

SRCIP	SRCPORT	DSTIP	DSTPORT
192.168.1.106	52000	130.251.222.84	80
192.168.1.10	53000	130.251.222.84	80

R1 →

R1 →

SRCIP	SRCPORT	DSTIP	DSTPORT
203.0.113.100	52000	130.251.222.84	80
203.0.113.100	53000	130.251.222.84	80



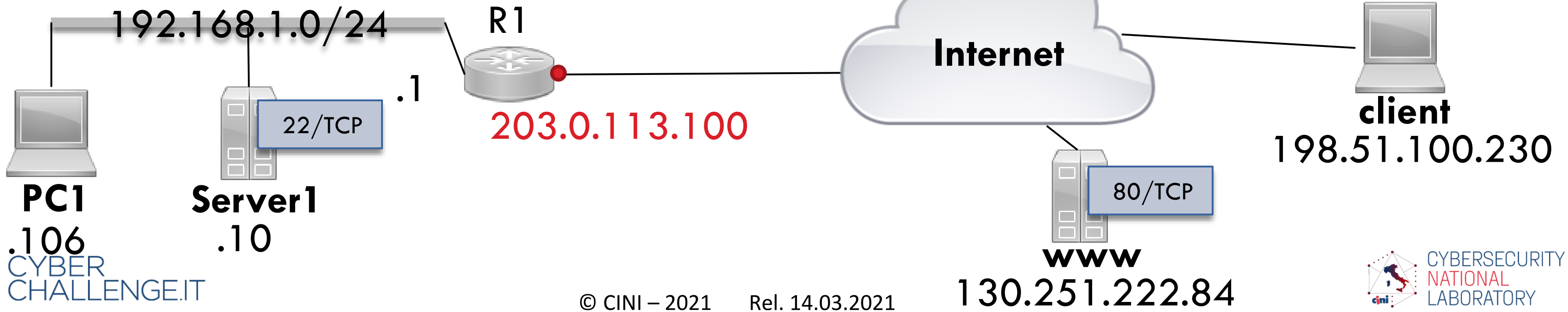
Source NAT and Masquerade (example)

- PC1 and Server1 accessing www (response)

SNAT table (dynamic)

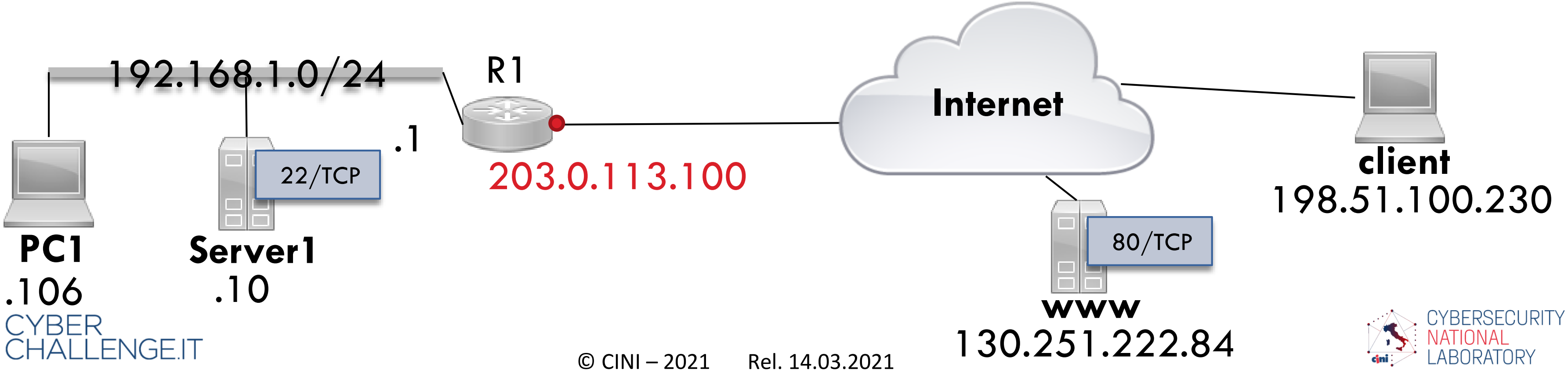
203.0.113.100	52000,80	192.168.1.106
203.0.113.100	53000,80	192.168.1.10

SRCIP	SRCPORT	DSTIP	DSTPORT		SRCIP	SRCPORT	DSTIP	DSTPORT
130.251.222.84	80	192.168.1.106	5200	R1 ←	130.251.222.84	80	203.0.113.100	5200
130.251.222.84	80	192.168.1.10	5300	R1 ←	130.251.222.84	80	203.0.113.100	5300



Port forwarding

- Port forwarding is a **destination NAT** rule, i.e., it is related to the destination address of a packet
- It maps external IP addresses and ports to Internal IP addresses and ports, allowing access to internal services from the Internet



CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

Port forwarding (example)

- Client connecting to Server1 (request)

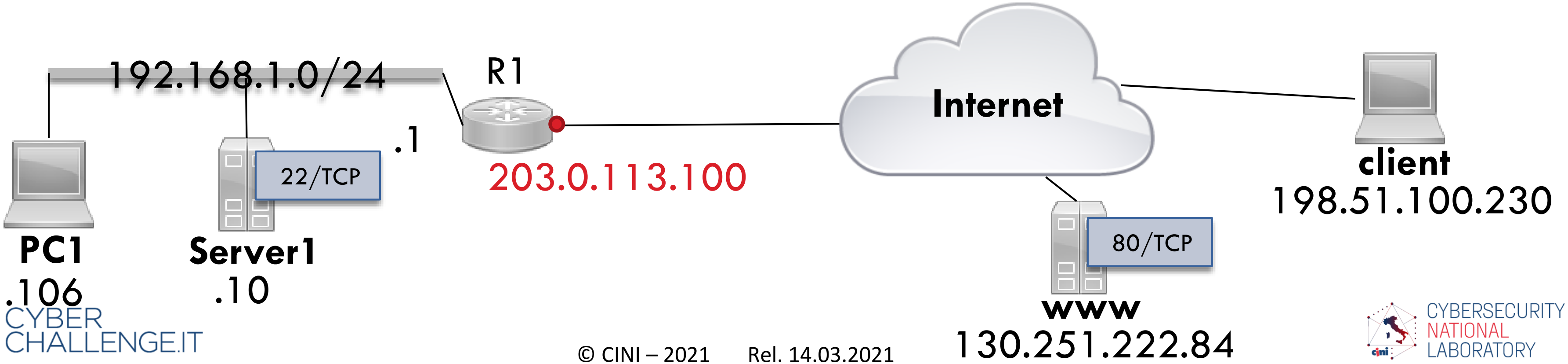
DNAT table (static)

Public IP address	Ext. port	Private IP address	Int. Port
203.0.113.100	22	192.168.1.10	22

SRCIP	SRCPORT	DSTIP	DSTPORT
198.51.100.230	54000	192.168.1.10	22

R1 ←

SRCIP	SRCPORT	DSTIP	DSTPORT
198.51.100.230	54000	203.0.113.100	22



Port forwarding (example)

- Client connecting to Server1 (response)

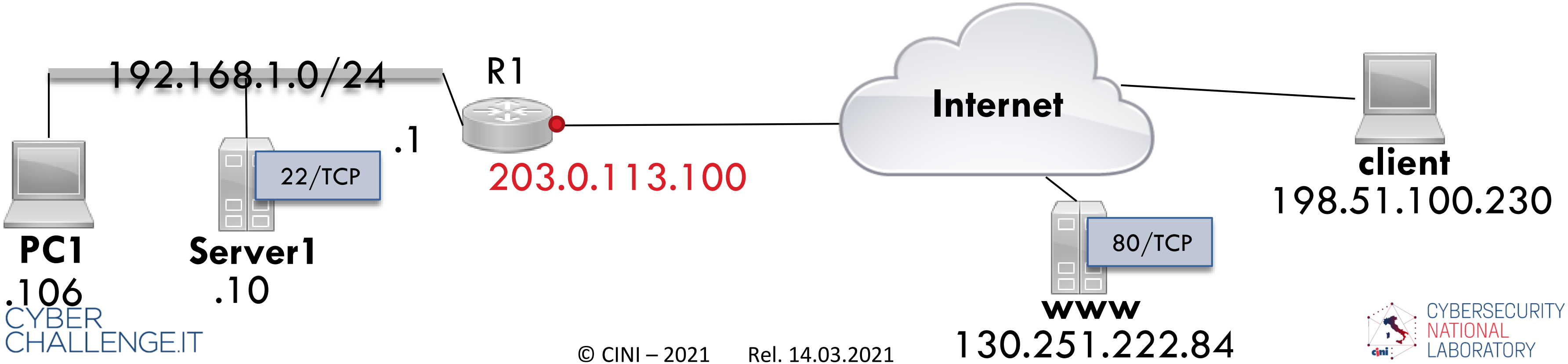
DNAT table (static)

Public IP address	Ext. port	Private IP address	Int. Port
203.0.113.100	22	192.168.1.10	22

SRCIP	SRCPORT	DSTIP	DSTPORT
192.168.1.10	22	198.51.100.230	54000

R1 →

SRCIP	SRCPORT	DSTIP	DSTPORT
203.0.113.100	22	198.51.100.230	54000



Outline

- ISO/OSI model, TCP/IP model and protocols
- Layer 2: Ethernet and bridges
- Layer 3: IP and routing
- Layer 4: TCP and UDP
- Client/Server model
- Network Address Translation
- Application layer examples: DNS and HTTP

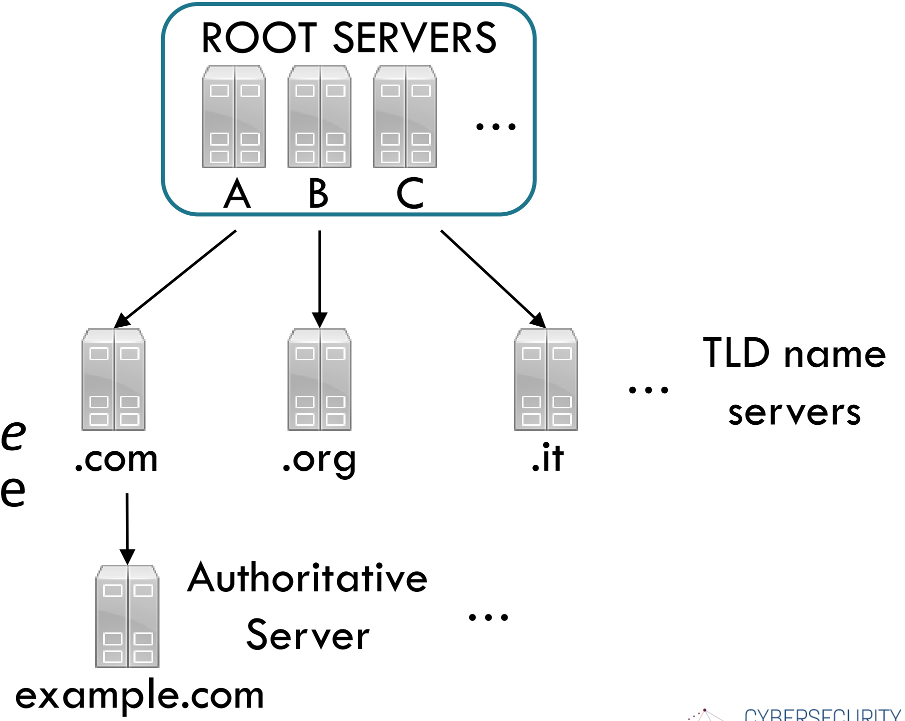
© CINI – 2021

Domain Name System (DNS)

- The *Domain Name System* (or *DNS*) converts human readable domain names (e.g., www.example.com) into IP addresses (e.g., 1.2.3.4)
- DNS is also the standard Internet mechanism for storing and accessing several other kinds of information about hosts (e.g., a *Mail eXchanger* record is used for routing a domain's incoming mail to a specific server)
- A DNS server uses well-known port 53 UDP/TCP

The DNS process (simplified)

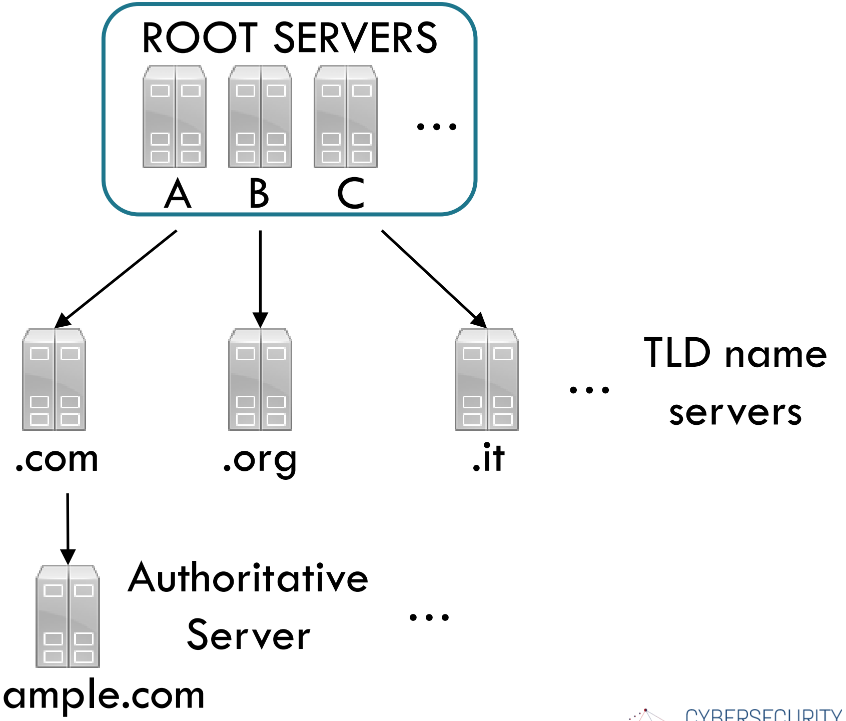
- An application or device (DNS client) issues a DNS address lookup (query), providing a hostname such as “example.com”
    1. The first server the query interacts with is the *recursive resolver*, which is responsible for finding the correct IP address for that hostname



© CINI – 2021

The DNS process (simplified)

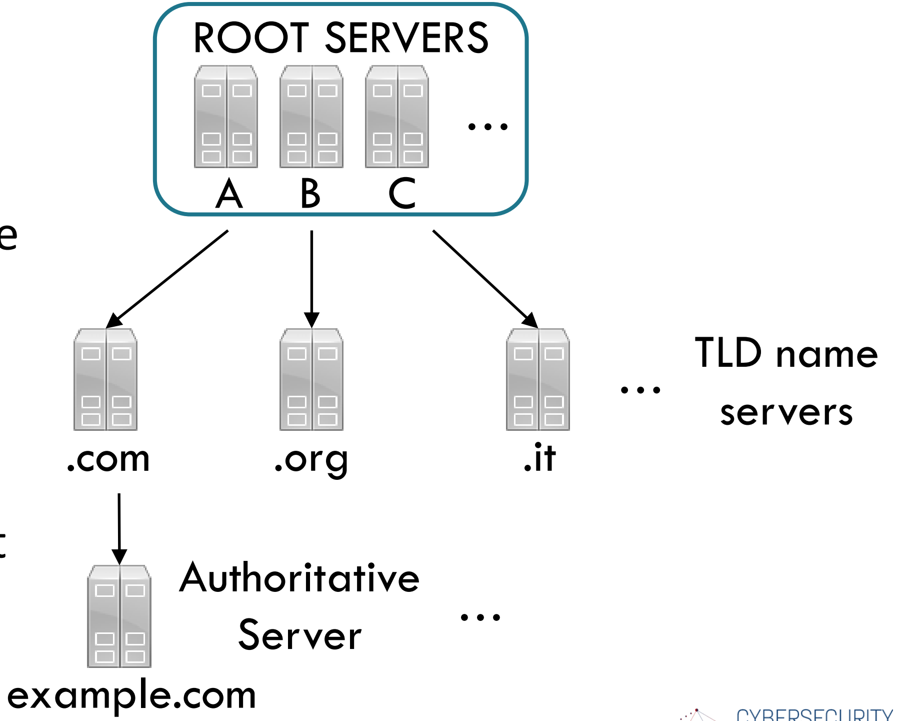
2. To begin answering the query, the recursive resolver asks a root server for DNS information about *.com*
3. The recursive resolver obtains the address of the related Top Level Domain (TLD) DNS server



© CINI – 2021

The DNS process (simplified)

4. The recursive resolver asks the TLD for information about the second level domain *example.com*
5. The recursive resolver obtains the address of the name server responsible for the specific domain *example.com* (authoritative server)
6. When the resolver reaches the authoritative DNS name server, it receives the IP address and returns it to the DNS client.



Hypertext Transfer Protocol (HTTP)

- HTTP is a protocol which allows the fetching of resources, such as HTML documents
- HTTP is a client-server protocol
  - Requests are sent by one entity, namely the user-agent (e.g., a Web browser)
  - On the opposite side of the communication channel, is the server, which provides the document as requested by the client
  - A HTTP server uses the well-known port 80 TCP

© CINI – 2021

Uniform Resource Locators (URLs)

- URL is the mechanism used by browsers to retrieve any published resource on the web

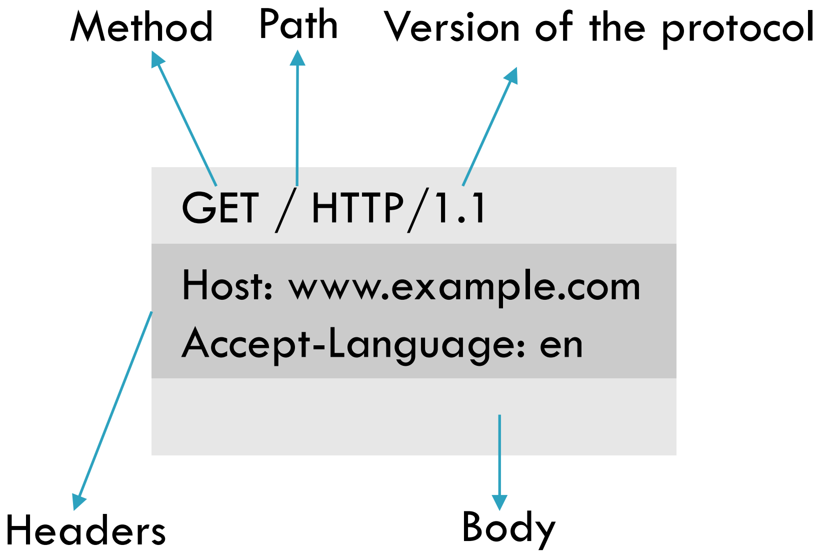
`http://`	`www.example.com`	`:80`	`/path/to/myfile.html`	`?key1=value1&key2=value2`	`#SomewhereInTheDoc`
the protocol to be used	the name of the web server	the port (usually omitted if it is the well-known)	the path to the resource on the web server.	extra parameters provided to the web server	fragment identifier: refers to a specific location within the resource being returned.

HTTP messages

- Client and server exchange HTTP messages.
  - **HTTP Requests**: sent by the client to trigger an action on the server.
  - **HTTP Responses**: the answer from the server.
- HTTP messages are plain text, i.e., line-oriented sequences of characters.

© CINI – 2021

HTTP messages: requests



Method Path Version of the protocol

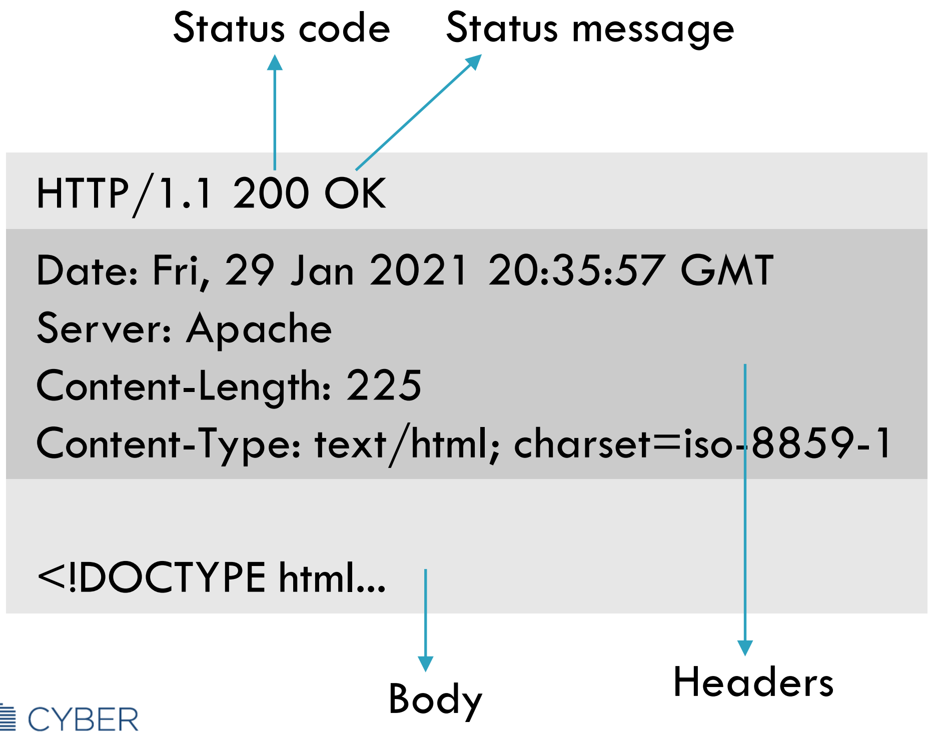
```
GET / HTTP/1.1
Host: www.example.com
Accept-Language: en
```

Headers Body

- **Method** defines the operation the client wants to perform. Typically, a client wants to fetch a resource (GET) or post the value of an HTML form (POST), though more operations may be needed in other cases
- **Path** corresponds to the URL of the resource stripped from elements that are obvious from the context (i.e., protocol, port, and domain)
- **Headers** (optional) convey additional information for the servers
- **Body** (optional): for some methods (e.g., POST) contains the resource sent

© CINI – 2021

HTTP messages: responses



```
HTTP/1.1 200 OK
Date: Fri, 29 Jan 2021 20:35:57 GMT
Server: Apache
Content-Length: 225
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE html...
```

- **Status code** indicates if the request was successful, or not, and why
- **Status message** is a non-authoritative short description of the status code
- **Headers** are like those for requests
- **Body** (optional) contains the fetched resource

Enrico RUSSO

Università di Genova

Network Fundamentals

CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

https://cybersecnatlab.it