

# Introduction to Computer Networks and Internet

Architecture et Réseaux

<https://lms.univ-cotedazur.fr/course/view.php?id=14141>

Dino Lopez Pacheco

<http://www.i3s.unice.fr/~lopezpac/>



<https://lms.univ-cotedazur.fr/course/view.php?id=14141>

Reseaux-GX

# Foreword

- The material presented in this course is a compilation of several Internet howto's and websites, command manual pages, API documentations, IETF RFCs, and books.
- Specially, the following slides are directly inspired from the official slides and book “Computer Networking: A Top Down Approach”. Chapter 1. 6<sup>th</sup> edition. Jim Kurose, Keith Ross. Pearson.
  - Exercises are also inspired from this book
- This course should allow you to understand the general architecture, components and basic configuration of a network.
  - Labs sessions for knowledge reaffirmation

# Organisation générale du cours

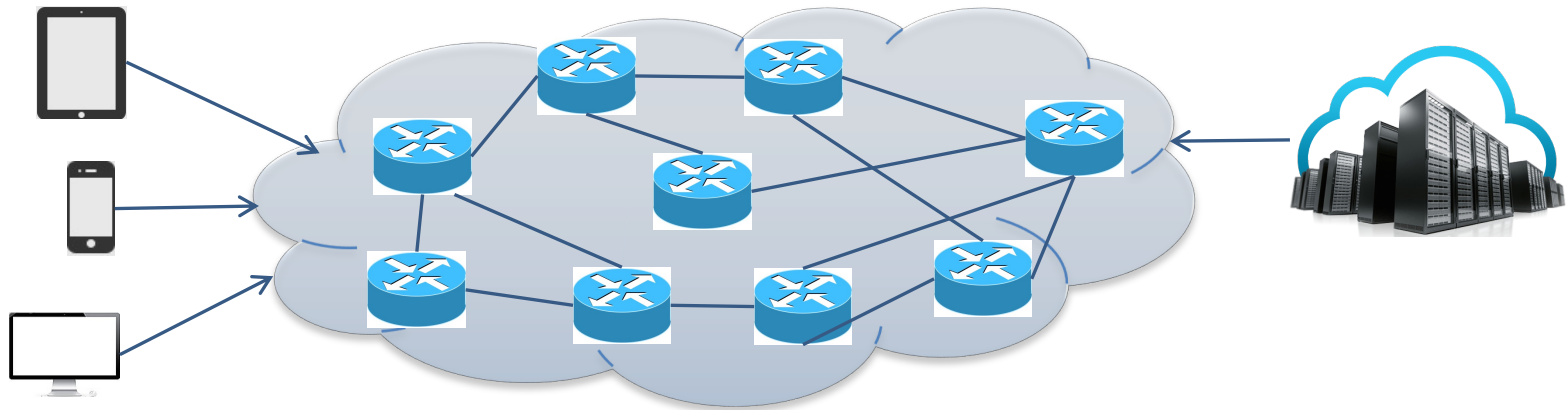
- Ce cours est divisé 2 parties
  - 1<sup>ère</sup> partie - Dino Lopez Pacheco, Pr. Jean Martinet, Frédéric Rallo, Pr. Yves Roudier – Introduction **Réseau** et Programmation Sockets
  - 2<sup>ème</sup> partie – Benoît Miramond – **Architectures de programmes**
- Pour la première partie
  - 4 encadrants des TDs pour 4 groupes
  - 1 groupe sera distribué de manière équitable sur les 4 salles avec les encadrants

# Interaction Application Distribué – Infrastructure réseau

Architecture des App. Dist. : Protocoles et Standards

Le Protocole HTTP : Pierre angulaire des Services Web

Infrastructure et Protocoles Réseau  
Interconnexion de Dispositifs



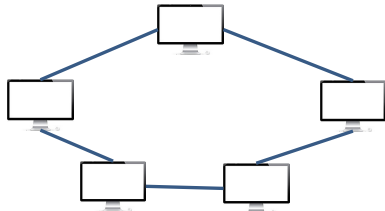
# Evaluation

- Chaque partie est évalué d'une manière qui corresponde le mieux au contenu à présenter
- En ce qui concerne la première partie
  - Examens courts écrit (0,5h)
  - QCMs surprises
  - Comportement générale en cours et TPs -> Les TPs peuvent être notés

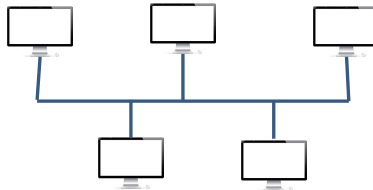
# General Networking Context

# Device interconnection: the network topologies

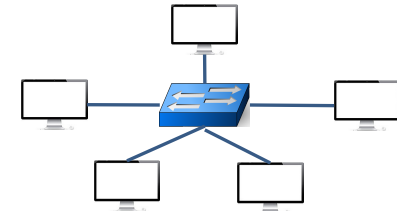
Ring



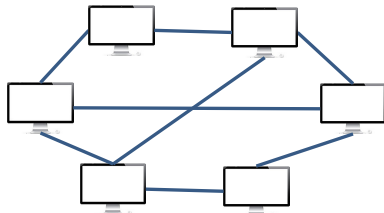
Bus



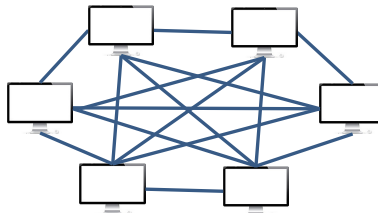
Star



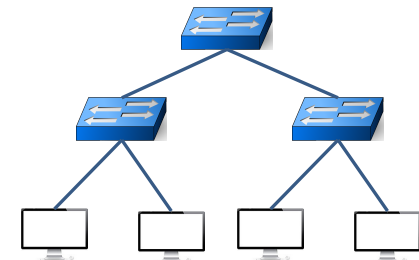
Mesh



Full mesh



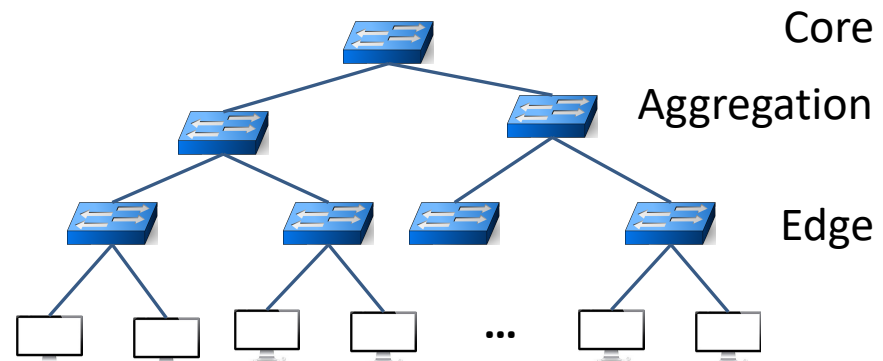
Tree





# Data Center Tree Topology

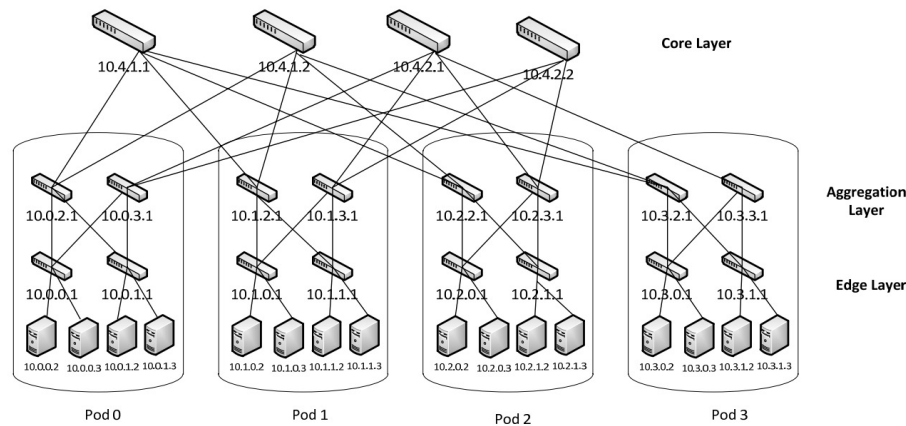
- 3-level Tree Switch
  - Core
  - Aggregation
  - Edge
- Symmetric well balances topology
  - This still needs redundancy



# Fat Tree Topology

- K-ary fat tree

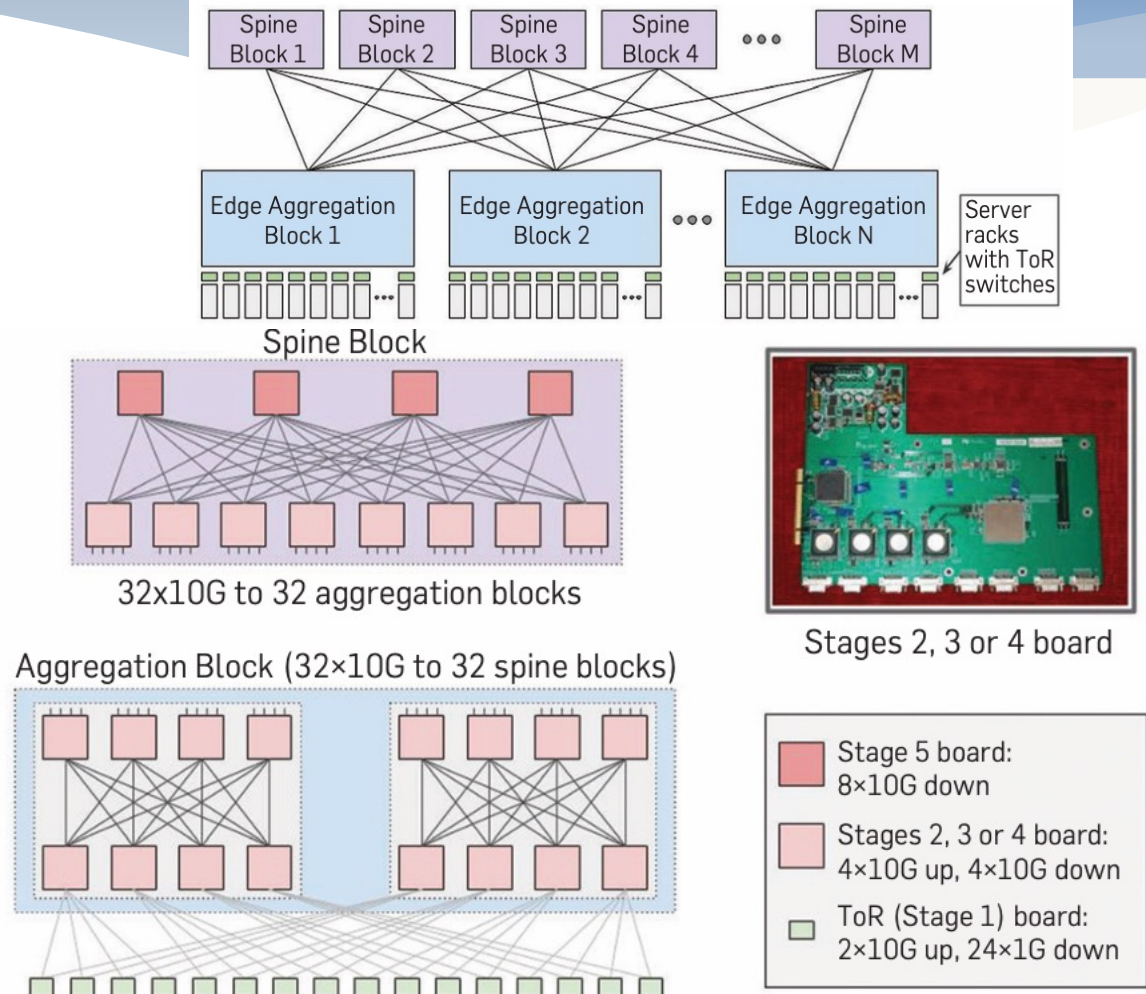
- K PoDs (Point of Delivery)
- $(k/2)^2$  servers per pod
- $K/2$  edge and aggregation switches per pod
- Each edge switch connects to  $k/2$  servers and  $k/2$  aggregation switches
- Each aggregation switch connects to  $k/2$  edge switch and  $k/2$  core switches
- $(k/2)^2$  core switches, each connected to k pods



From V. Sharma and R. Mishra, "A Comprehensive Survey on Data Center Network Architectures," *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2020, pp. 222-228, doi: 10.1109/ICRITO48877.2020.9197934.

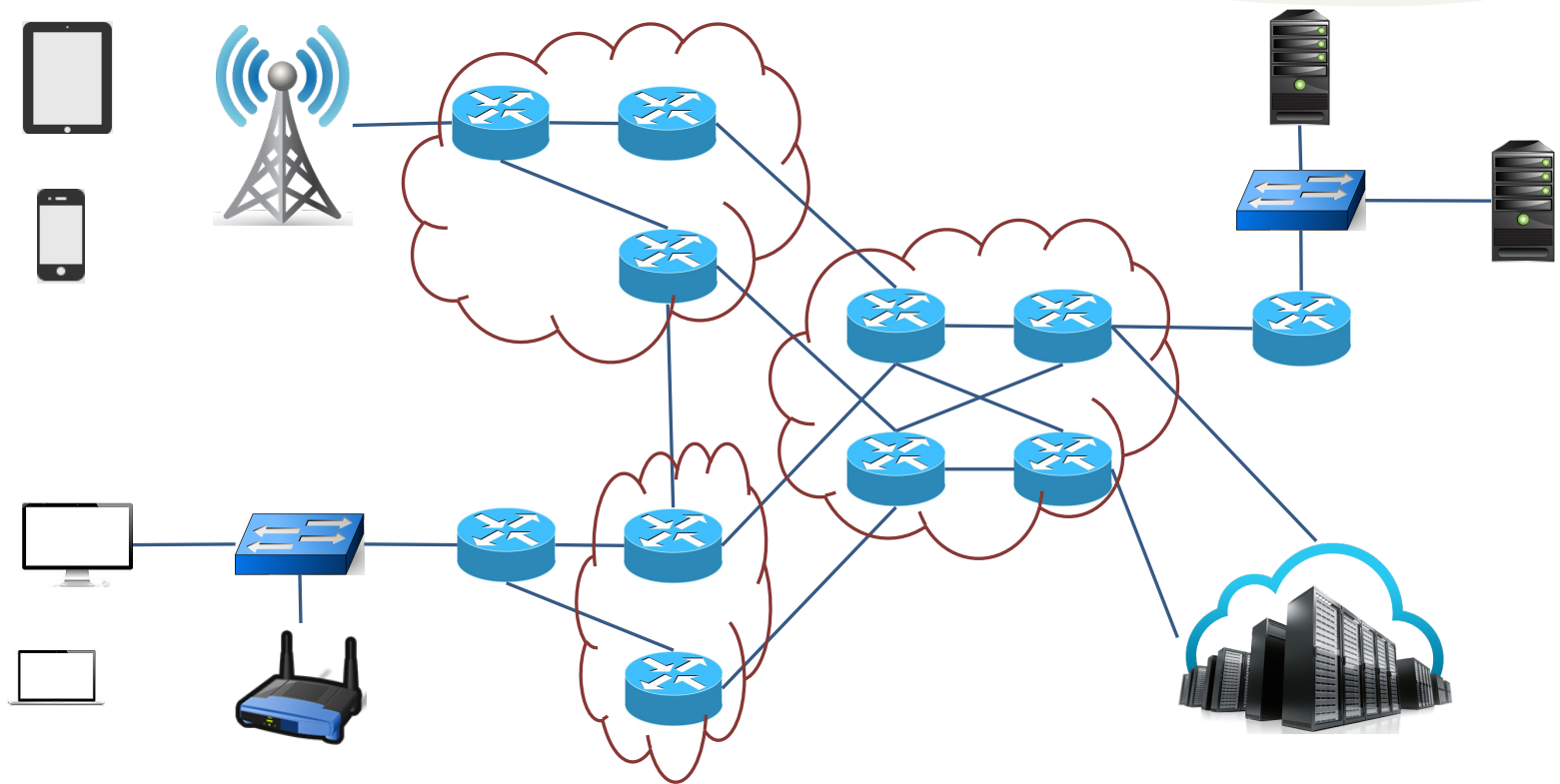
# A real DC fabric

- Redundant links
- The old Google DC fabric
  - 2005
  - 10TB of bisection bandwidth



From "Arjun Singh et al. 2015. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. SIGCOMM Comput. Commun. Rev. 45, 4 (October 2015), 183–197. DOI:<https://doi.org/10.1145/2829988.2787508>

# Global View

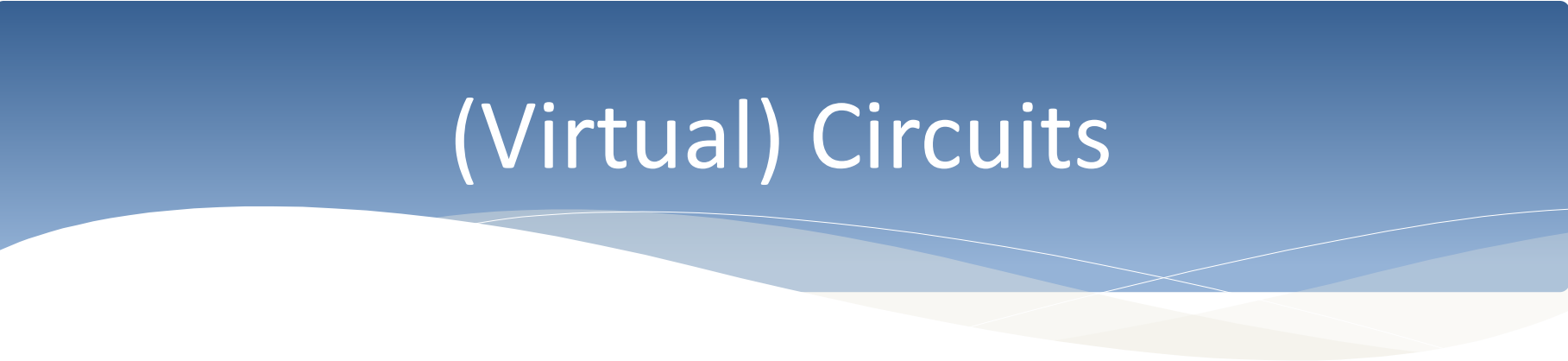


# Switching

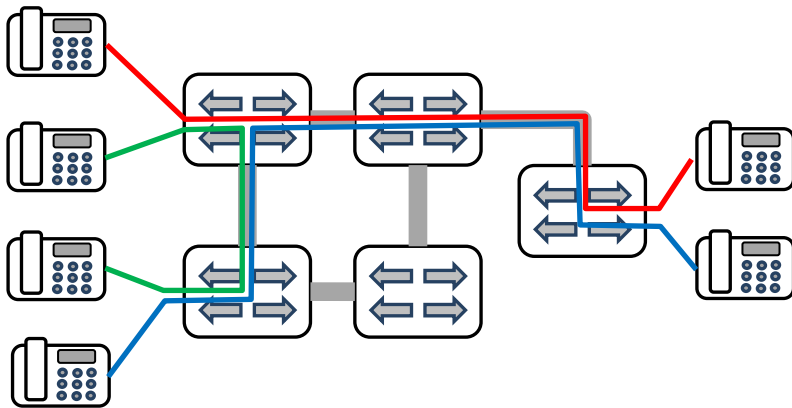
- To interconnect two users' devices (*end points*), intermediate devices will probably be needed
- Intermediate devices will transfer the traffic between several users
- Intermediate devices need to switch from one conversation to another one
  - Circuit switching
    - \* Fixed and Mobile telephone networks
    - \* Optical Networks (rings)
  - Message switching
    - \* Application-layer protocols
  - Packet switching
    - \* Internet

# Circuit switching

- In circuit switched networks, the resources are ***reserved*** for the entire duration of the communication
  - Takes a long time to create a (virtual) circuit
  - Users' data travels at the links speed
  - Silent periods lead to wasted resources
- The end-to-end reserved path is known like a circuit
- A circuit can be implemented using
  - Frequency Division Multiplexing (FDM)
  - Time Division Multiplexing (TDM)

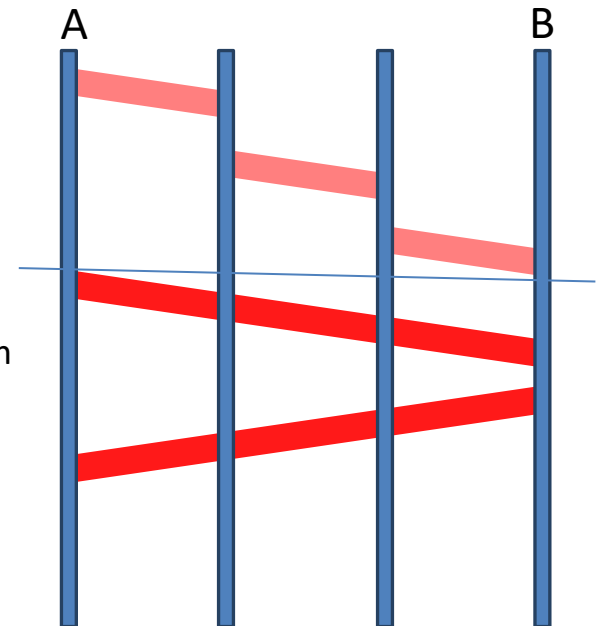


# (Virtual) Circuits

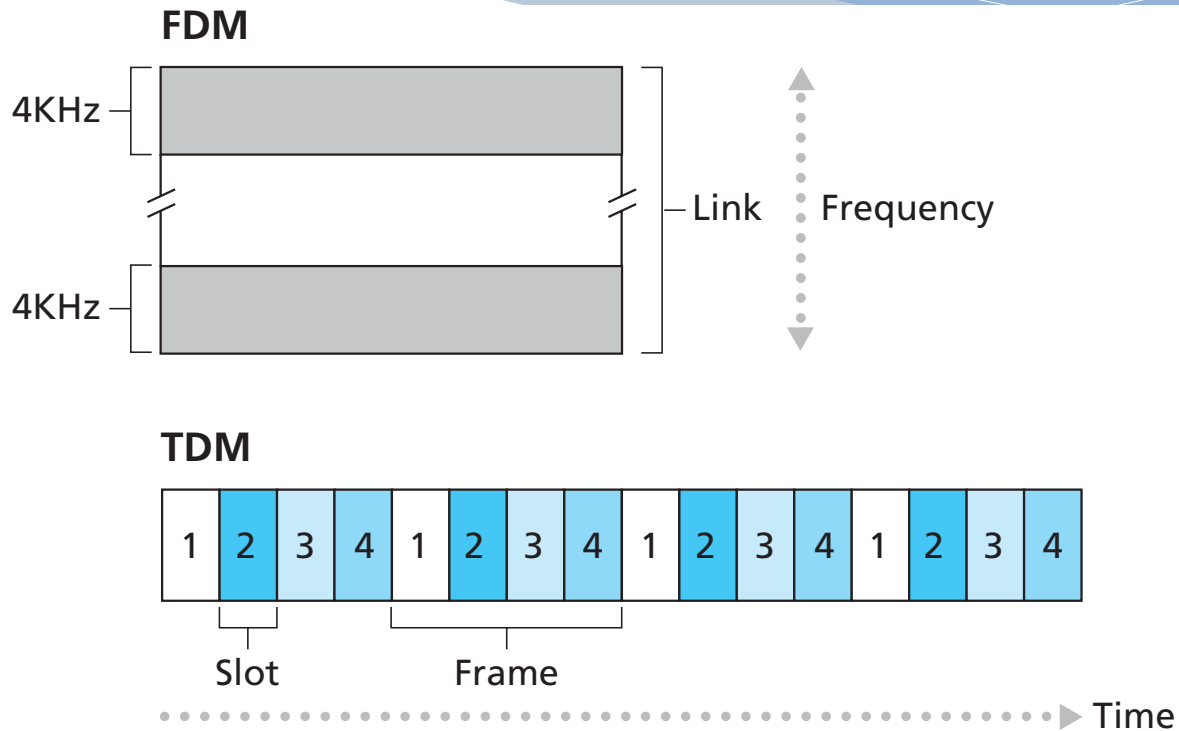


Dialing B's  
number

Conversation  
is ongoing



# FDM vs TDM

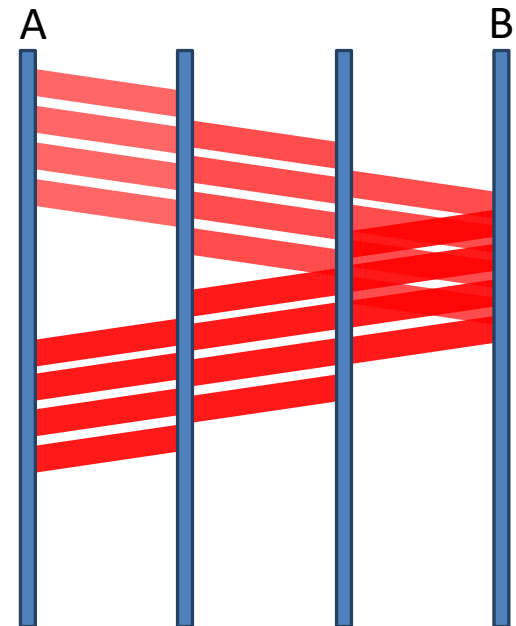
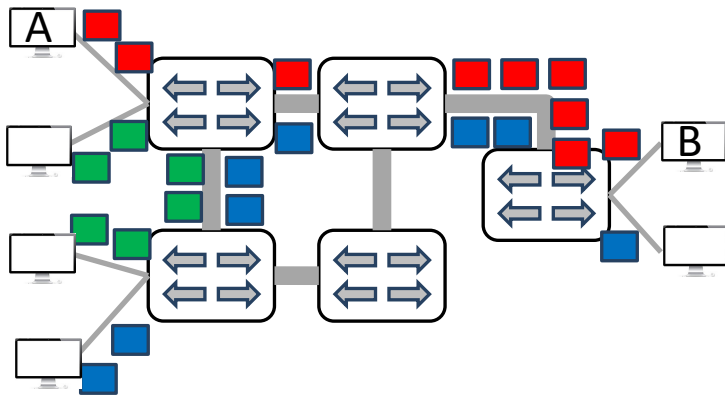


Key:

**2** All slots labeled "2" are dedicated to a specific sender-receiver pair.



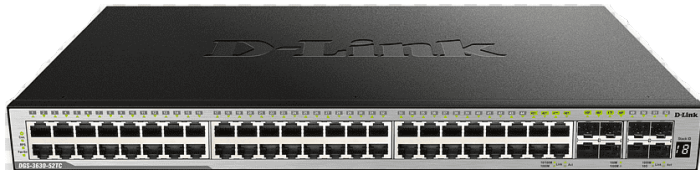
# Packet Switching



# Internet elements

- Internet is a network which interconnects computing devices
  - Internet Service Providers
  - Network of networks
- Computing devices: hosts or end systems
  - Laptops, PC, servers
  - Today: Mobile devices, smart objects
- Communication links
  - Twisted pair, coaxial cable, radio frequency
  - ?
- Packet switches
  - Routers, link-layer switches
  - Several other devices
- Protocols
  - Standard and proprietary protocols

# Internet elements by graphics



Switch (enterprises)



Switch (at home)



Wi-Fi access point (by Extreme Networks)

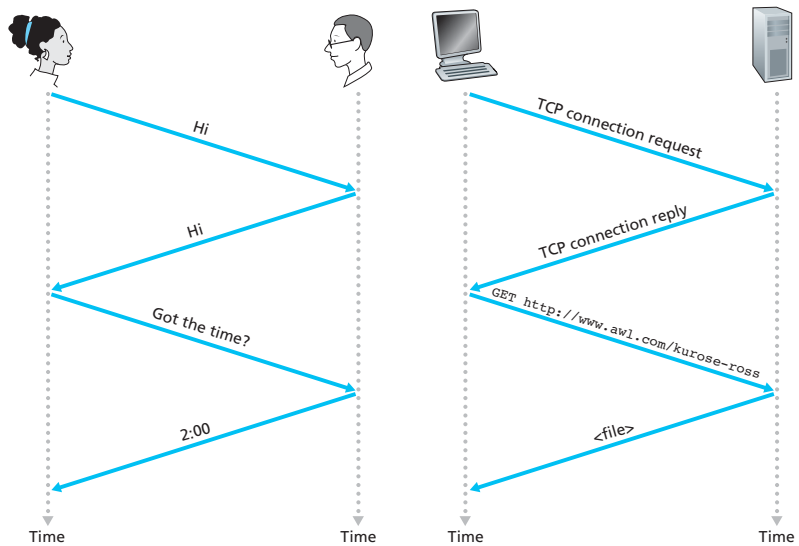


Ethernet cables  
(twisted pair cable)

# Internet Applications

# Internet Applications

- Distributed applications
  - Peer to peer
  - Client-server
- Information exchanges need to follow rules → protocols

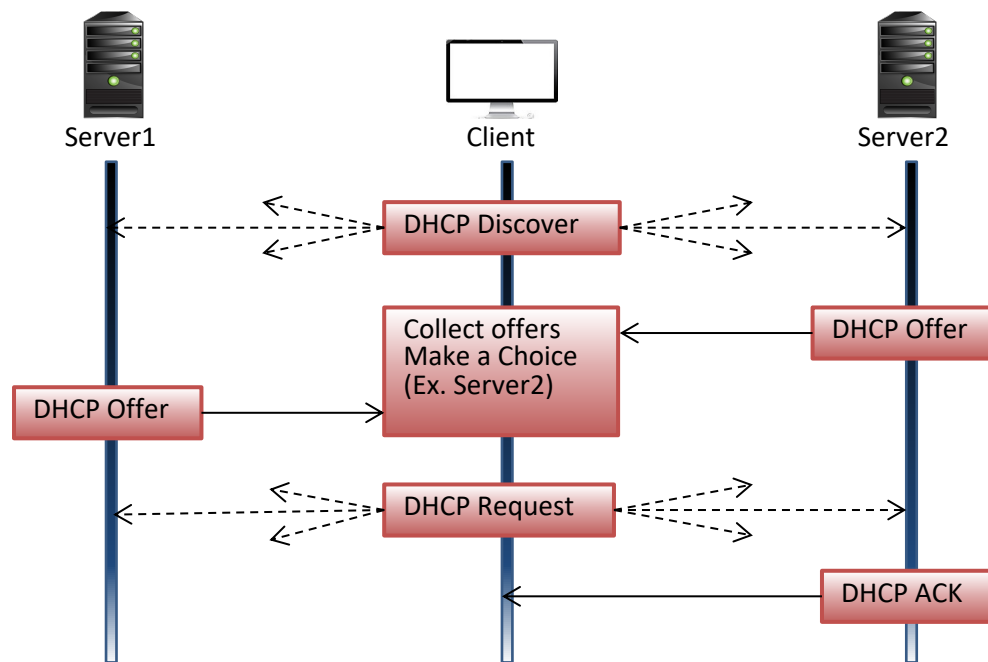


# Some network protocols

- Different protocols for different objectives and at different levels of the internet protocol stack
- Fairly used protocols
  - Easy network client configuration → DHCP
  - Translate friendly-user computer names to their IDs -> DNS
  - Download/upload/modify/delete documents -> HTTP, ?? ...
  - Obtain the current time -> NTP
  - ...
- Protocols to transfer users' data between two applications
  - TCP/UDP
- Network management, signaling and control
- ...

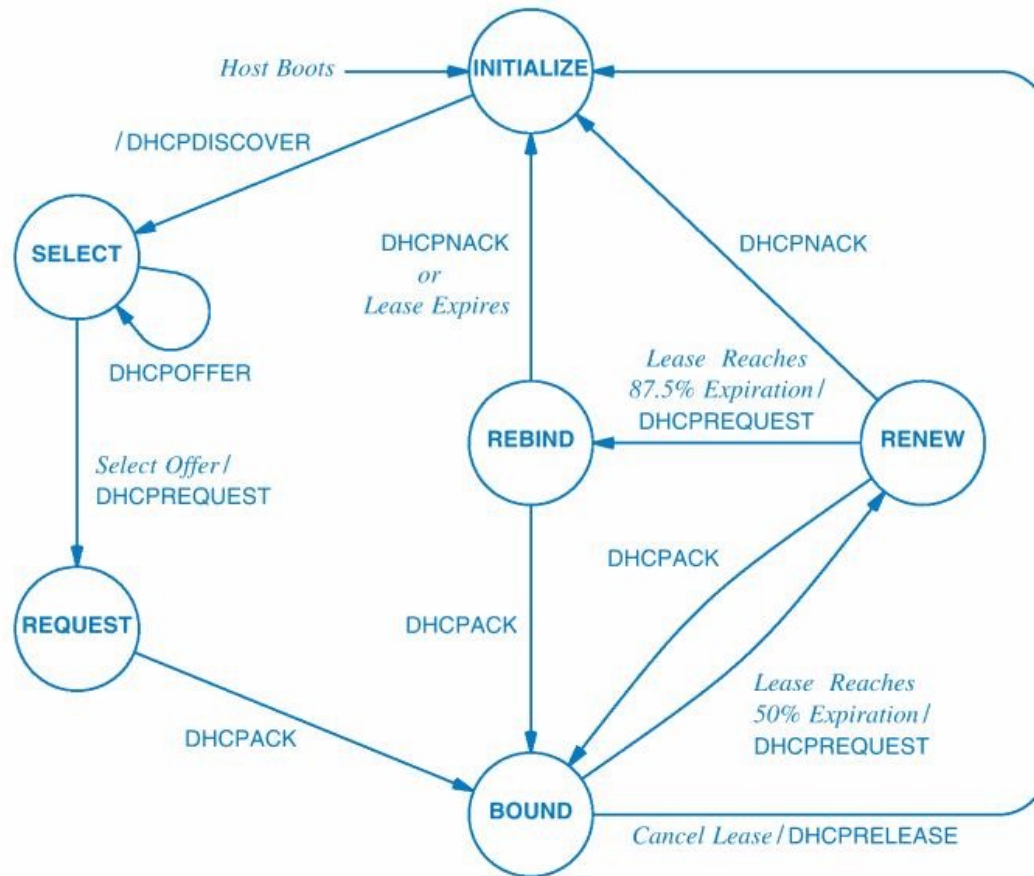
# Dynamic Host Configuration Protocol

- Automatic end host configuration
  - IP address, network mask, router address, DNS server, network domain, ...
- DHCP runs over the UDP protocol
  - Ports number 67 and 68
- Flexible but minimum control on the network settings
- Users rarely verify the obtained configuration → security issues



Expected DHCP behavior

# DHCP state diagram



DHCP state diagram from Douglas E. Comer (2006). "Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture", 5ed, Prentice Hall



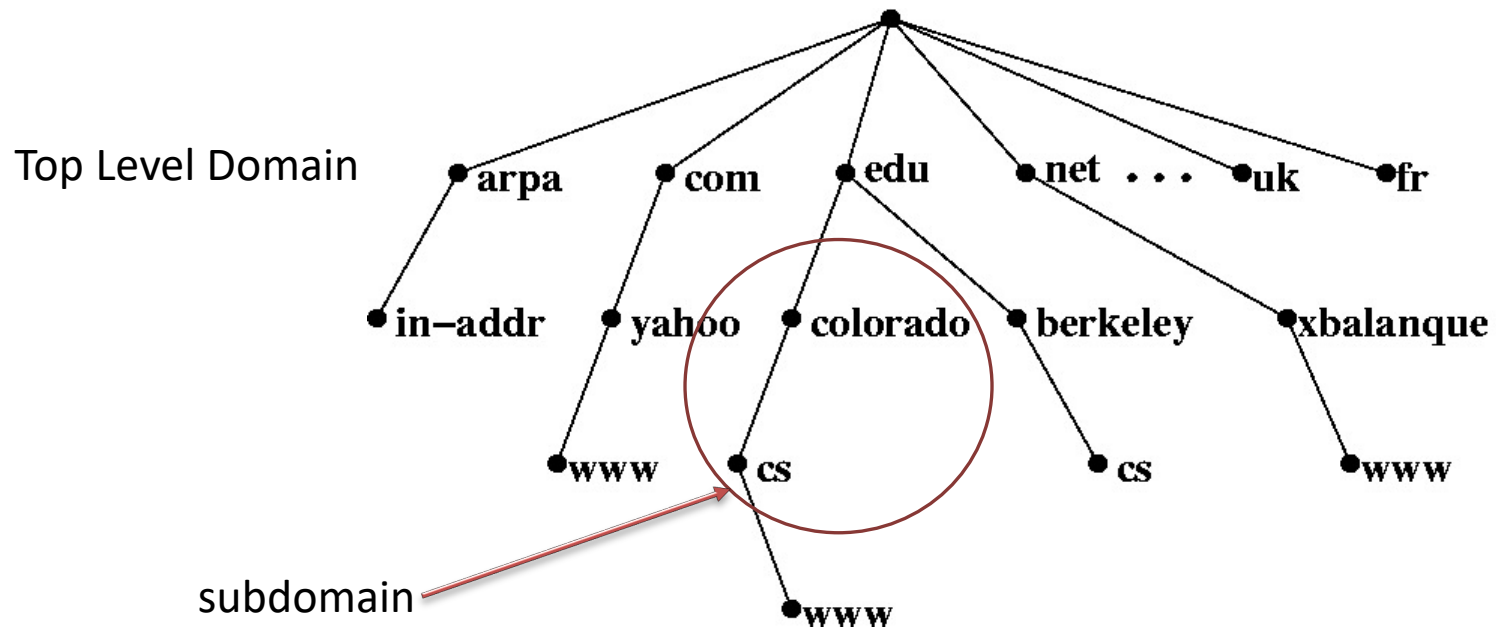
# Domain Name System

- Translate user-friendly names to IP addresses
- Hierarchical architecture
- Some definitions
  - Name server → DNS server software
  - Resolver → DNS client software
  - Fully Qualified Domain Name -> FQDN
  - Partially Qualified Domain Name -> PQDN
  - Top level domains -> domains just below the root (unnamed domain)
    - \* Generic top level domains -> gTLD (.com, .edu, .gov,...)
    - \* Country code top level domains -> ccTLD (.fr, .br, .jp, ...)
  - subdomain -> domain inside a domain
- Uses TCP and UDP on port 53

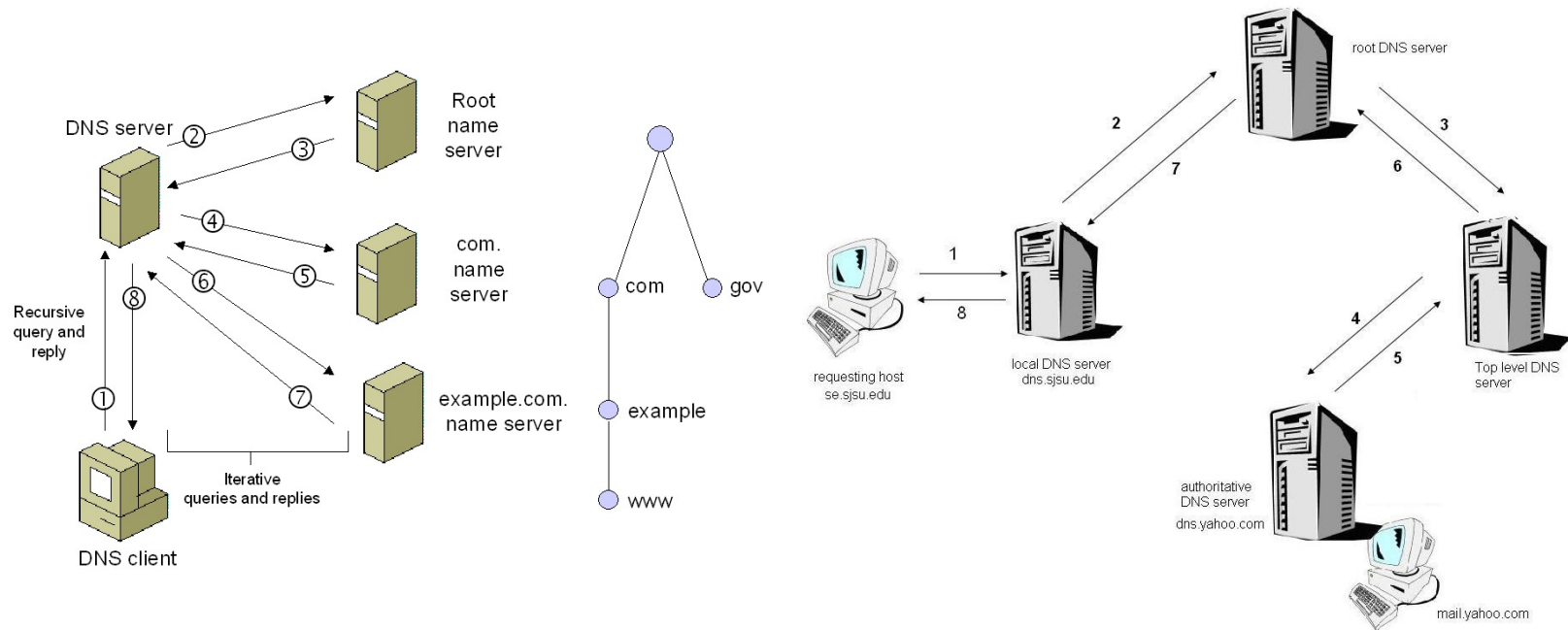
# Some DNS records

- DNS relies on a distributed data base
  - Authoritative DNS servers → servers providing the answer from the original source file for a given record
  - Non-authoritative DNS servers -> servers providing the answer from a cache or by forwarding a reply from an authoritative server
- Different resource records
  - A: Name to IPv4 address
  - AAAA: Name to IPv6 address
  - MX: Mail eXchange server of a domain
  - NS: server name of a domain

# DNS name space



# Iterative and Recursive resolutions



# HyperText Transfer Protocol

- Employed to access data on the World Wide Web
  - Universal Resource Locators (URLs)
  - `http://www.i3s.unice.fr/~roudier/index.php?studentid=abc1234&courseId=12#section1`
    - \* `https://en.wikipedia.org/wiki/URL`
- Simple request-response protocol
  - Client (e.g. a browser) sends a request to an HTTP server
  - The HTTP server (e.g. Apache) sends a reply, potentially containing the required document
- Web services are often built on top of HTTP
  - Mobile Apps

# HTTP messages

HTTP-message = Request | Response

Request | Response = start-line  
                          \*(message-header CRLF)  
                          CRLF  
                          [ message-body ]

start-line = Request-Line | Status-Line

Request-Line = Method SP Request-URI SP HTTP-Version CRLF

Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF

# Request / Response examples

```
GET /index.html HTTP/1.1
```

```
Host: www.example.com
```

```
User-Agent: Mozilla/5.0
```

```
Connection: keep-alive
```

```
blank line
```

```
HTTP/1.1 200 OK
```

```
Server: Lighttpd/1.1
```

```
Content-Type: text/html; charset=UTF-8
```

```
Content-Length: 1846
```

```
blank line
```

```
<html>
```

```
...
```

```
</html>
```

# HTTP methods and response status codes

- Common methods (verbs)
  - GET: fetch a resource
  - HEAD: fetch information about a resource
  - PUT: replace a resource
  - DELETE: delete a resource
  - POST: submit information to an entity
  - ...
- GET and POST are commonly used in web browsers
- GET, PUT, POST, DELETE frequently used in REST APIs
- Response codes categories
  - 1xx informational
  - 2xx success
  - 3xx redirect
  - 4xx client error
  - 5xx server error
- Some common responses status codes
  - 200 OK
  - 301 Moved Permanently
  - 404 Not Found
  - 400 Bad Request
  - 401 Unauthorized
  - 500 Internal Server Error
  - 501 Not Implemented
  - 550 Permission denied



# Network Layers and Protocols

# Packetization

- IP networks has been designed for packet-switched networks
- Users' applications generate a stream of data (e.g. audio, video, etc.)
  - Need to break data into small segments for network transmissions
- Users' data must be processed to reach the network interfaces and travel
  - Packetizing, end to end signaling, encoding/decoding, ...
  - Several solution at different level (packets going through the air, copper cable, other?)
  - Different needs (end users' data, network control, ...?)
  - Need to keep interoperability between the different solutions and needs

# The Layering approach

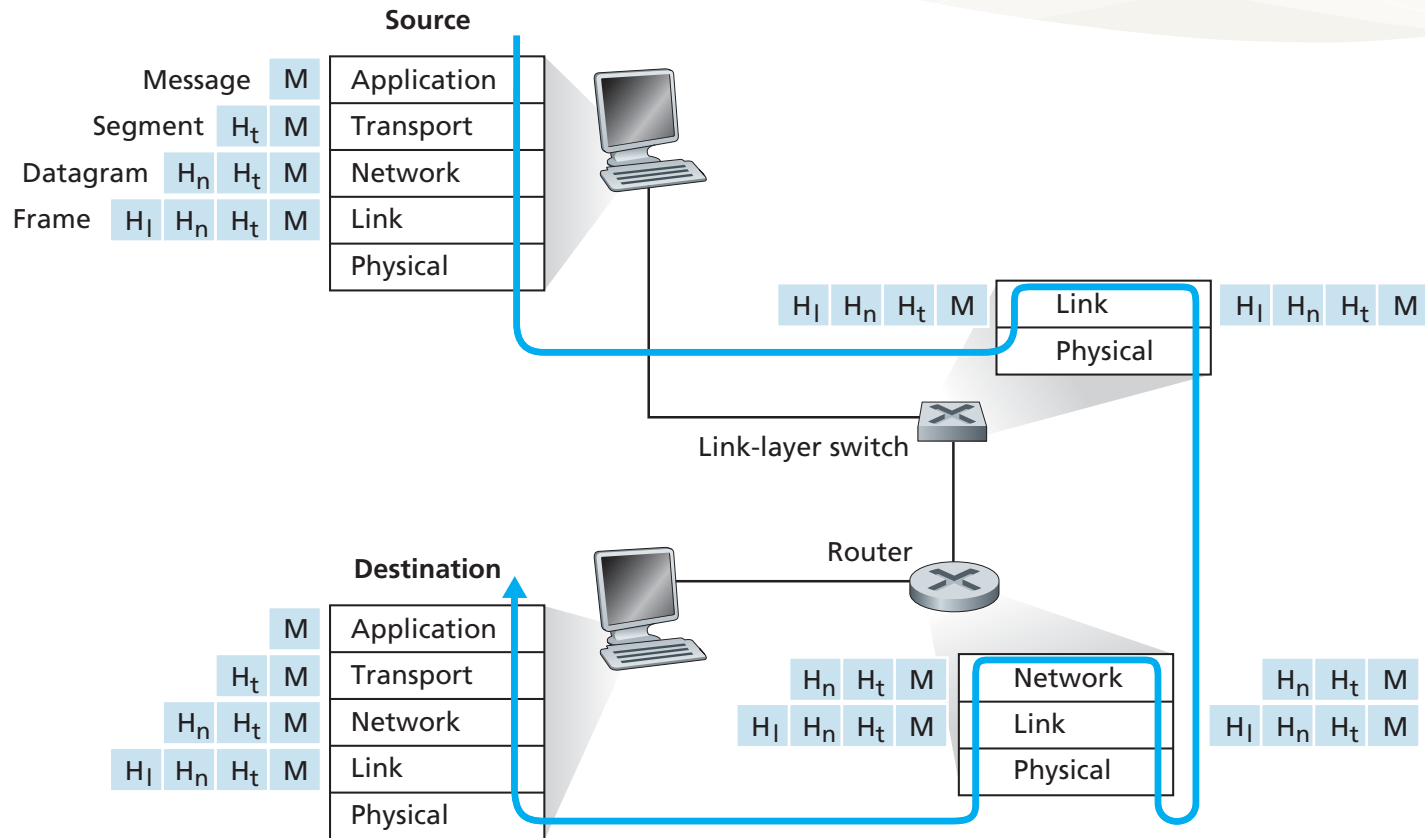
The OSI model

Application
Presentation
Session
Transport
Network
Link
Physical

The Internet protocol stack

Application	HTTP
Transport	TCP,UDP
Network	IP
Link	ARP
Physical	Ethernet

# Encapsulation



# Encapsulation example: The HTTP case

Layer	Protocol	Content (information added to a packet)
Application	HTTP	HTTP request → GET / HTTP/1.1 ...
Transport	TCP	TCP Header → Src Port, Dst Port, ...
Network	IPv4	IP Header → Src IP Addr, Dst IP Addr, ...
Link	Ethernet - MAC	MAC Header → Src Eth Addr, Dst Eth Addr, ...
Physical	Ethernet – 10GBASE-10	Synch signaling, inter-frame gaps

# Encapsulation in real life

The image shows a Wireshark 1.12.7 packet capture window titled "out.pcap [Wireshark 1.12.7 (Git Rev Unknown from unknown)]". The interface includes a menu bar, a toolbar, a filter field, and a packet list table.

Source	Destination	Protocol	Length	Info
10.0.0.1	10.0.0.2	TCP	74	57896→8000 [SYN] Seq=0 Win=29
10.0.0.2	10.0.0.1	TCP	74	8000→57896 [SYN, ACK] Seq=0 A
10.0.0.1	10.0.0.2	TCP	66	57896→8000 [ACK] Seq=1 Ack=1
10.0.0.1	10.0.0.2	HTTP	206	GET / HTTP/1.1
10.0.0.2	10.0.0.1	TCP	66	8000→57896 [ACK] Seq=1 Ack=14
10.0.0.2	10.0.0.1	TCP	83	[TCP segment of a reassembled
10.0.0.1	10.0.0.2	TCP	66	57896→8000 [ACK] Seq=141 Ack=

Below the packet list, the packet details pane shows the structure of the selected packet (Frame 4):

- Frame 4: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits)
- Ethernet II, Src: 00:00:00 00:00:01 (00:00:00:00:00:01), Dst: 00:00:00 00:00:02 (00:00:00:00:00:02)
- Internet Protocol Version 4, Src: 10.0.0.1 (10.0.0.1), Dst: 10.0.0.2 (10.0.0.2)
- Transmission Control Protocol, Src Port: 57896 (57896), Dst Port: 8000 (8000), Seq: 1, Ack:
- Hypertext Transfer Protocol

The packet bytes pane shows the raw data in hexadecimal and ASCII:

```
0000 00 00 00 00 00 02 00 00 00 00 00 01 08 00 45 00 .....E.
0010 00 c0 9f 27 40 00 40 06 87 0e 0a 00 00 01 0a 00 ...'@.@. ....
0020 00 02 e2 28 1f 40 45 b4 ba 62 3d ad dd d3 80 18 ..(.@E. .b=....
0030 00 3a 14 b5 00 00 01 01 08 0a 00 63 25 44 00 63 .....c%D.c
0040 25 44 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31 %DGET / HTTP/1.1
0050 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 57 67 ..User-A gent: Wg
0060 65 74 2f 31 2e 31 36 2e 31 20 28 6c 69 6e 75 78 et/1.16. 1 (linux
0070 2d 67 6e 75 29 0d 0a 41 63 63 65 70 74 3a 20 2a -gnu)..A ccept: *
0080 2f 2a 0d 0a 41 63 63 65 70 74 2d 45 6e 63 6f 64 /*..Acce pt-Encod
0090 69 6e 67 3a 20 69 64 65 6e 74 69 74 79 0d 0a 48 ing: ide ntity..H
00a0 6f 73 74 2e 20 31 20 2e 20 2e 20 2e 20 2e 20 2e ...f s t . . . . .
```

The status bar at the bottom indicates: Internet Protocol Version 4 (ip...), Packets: 24 · Displayed: 24 (10...), Profile: Default.

# Multiplexing

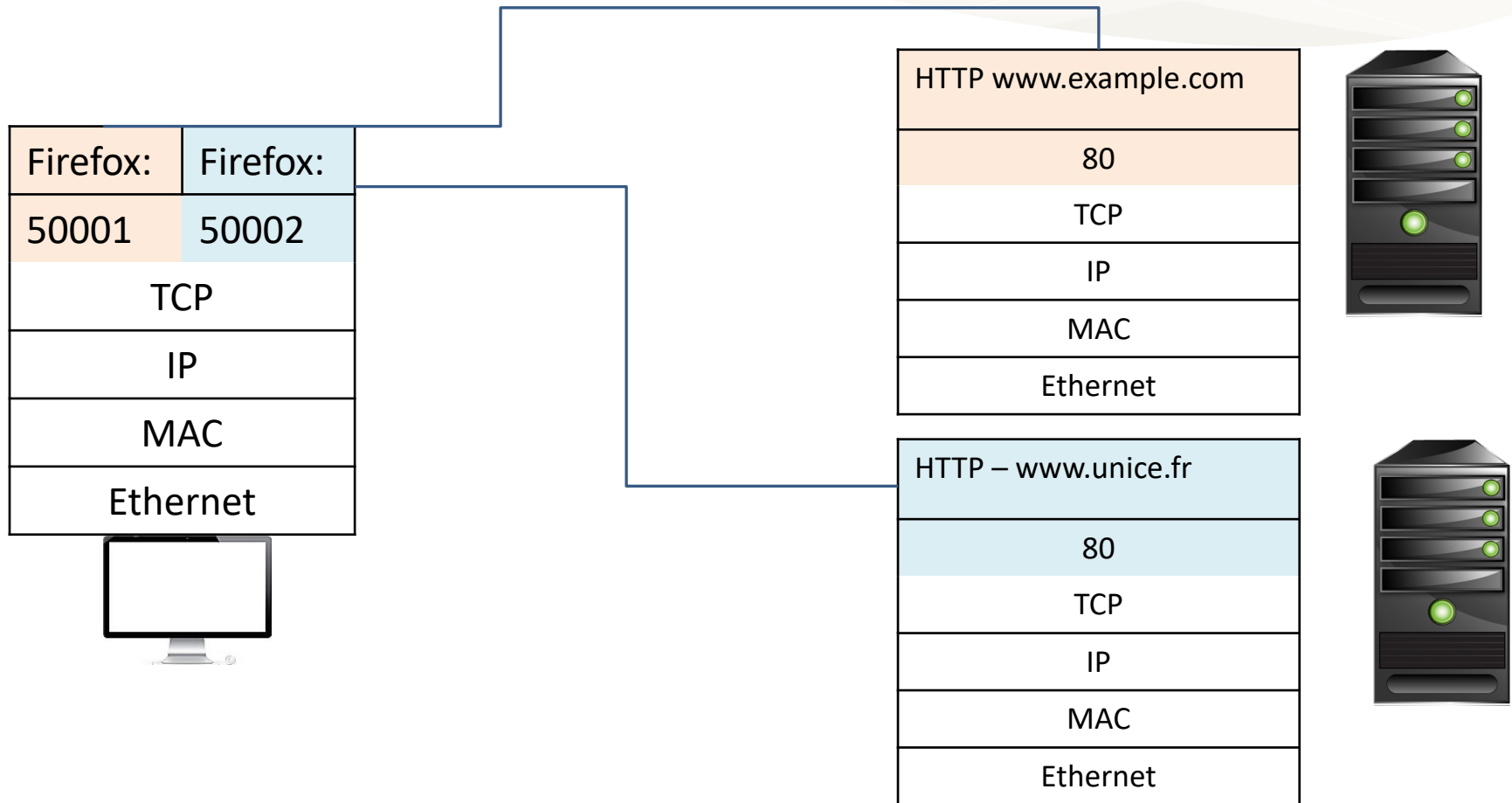
# Ports



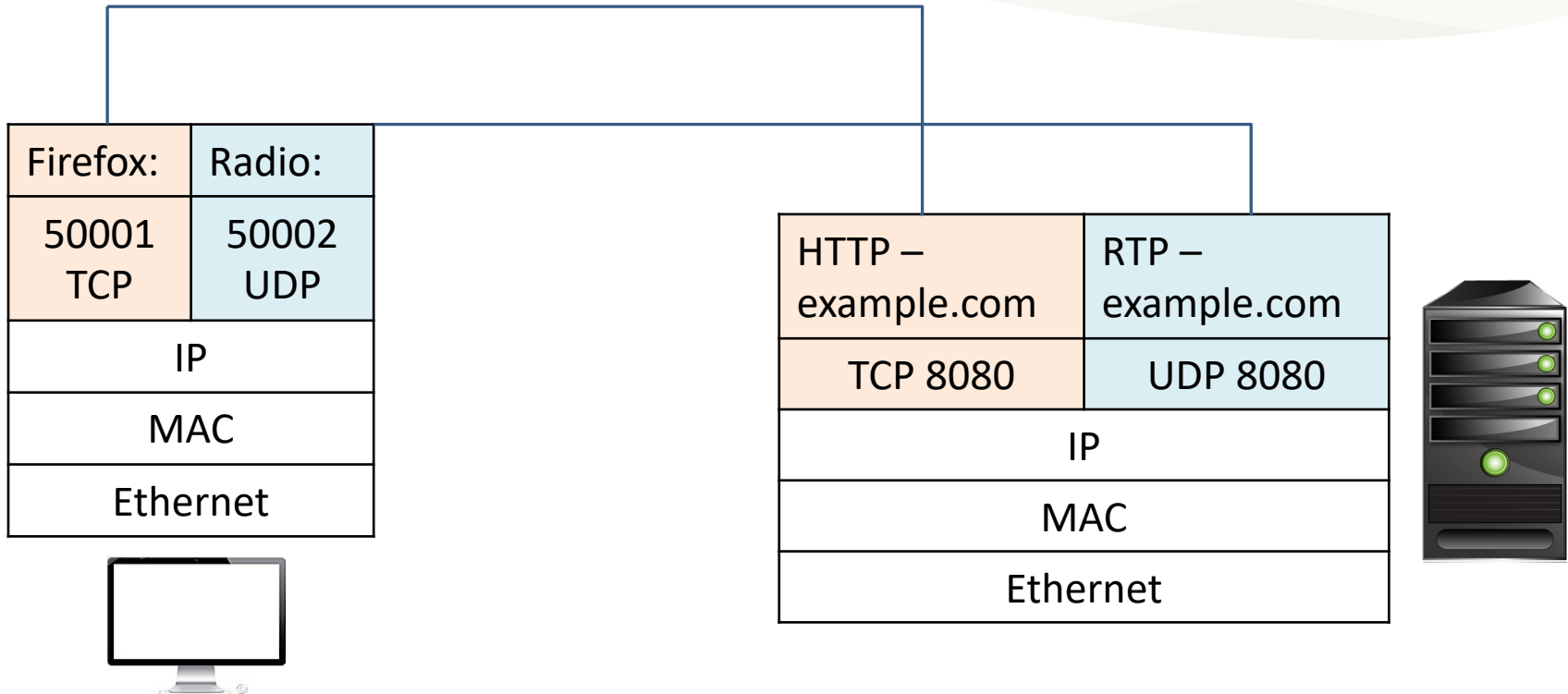
Image by WorldInMyEyes from Pixabay



# Multiplexing – same protocol, different port numbers



# Multiplexing – same port number, different protocols



# Network Delays

# Packet forwarding in packet-switched networks

- Store-and-forward devices
- A device receives the bits of a packet and store the information in a buffer
- Once the entire packet has been received, the packet is processed/forwarded

# Bandwidth definition

## bandwidth

**noun** [ C usually singular or U ]

UK  /'bænd.wɪtθ/ US  /'bænd.wɪtθ/

---

**bandwidth** **noun** [C usually singular or U] (INFORMATION)



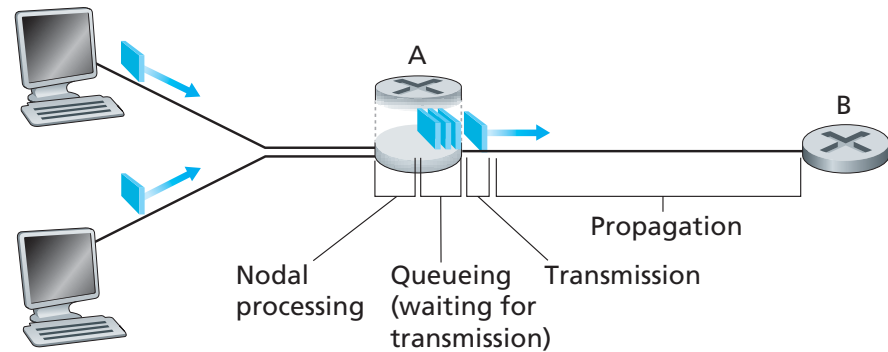
**a measurement of the amount of information that can be sent between computers, through a phone line, etc.:**

- *The system will handle signals that need **high** bandwidth, for instance those that encode TV pictures.*
- *high-bandwidth **services/applications***

By Cambridge Dictionnary

# Delay in packet-switched networks

- The total delay seen by a packet in a node (the nodal delay) is composed of
  - Processing delay: should a packet be forwarded and through which port
  - Queuing delay: how long a packet waits in the buffer before being forwarded
  - Transmission delay: time needed to put an entire packet in the link
- Propagation delay: the speed at which bits propagates in a link



From "Computer Networking: A Top-Down Approach". PEARSON, 6<sup>th</sup> edition. James F. Kurose and Keith W. Ross

# End-to-end delay

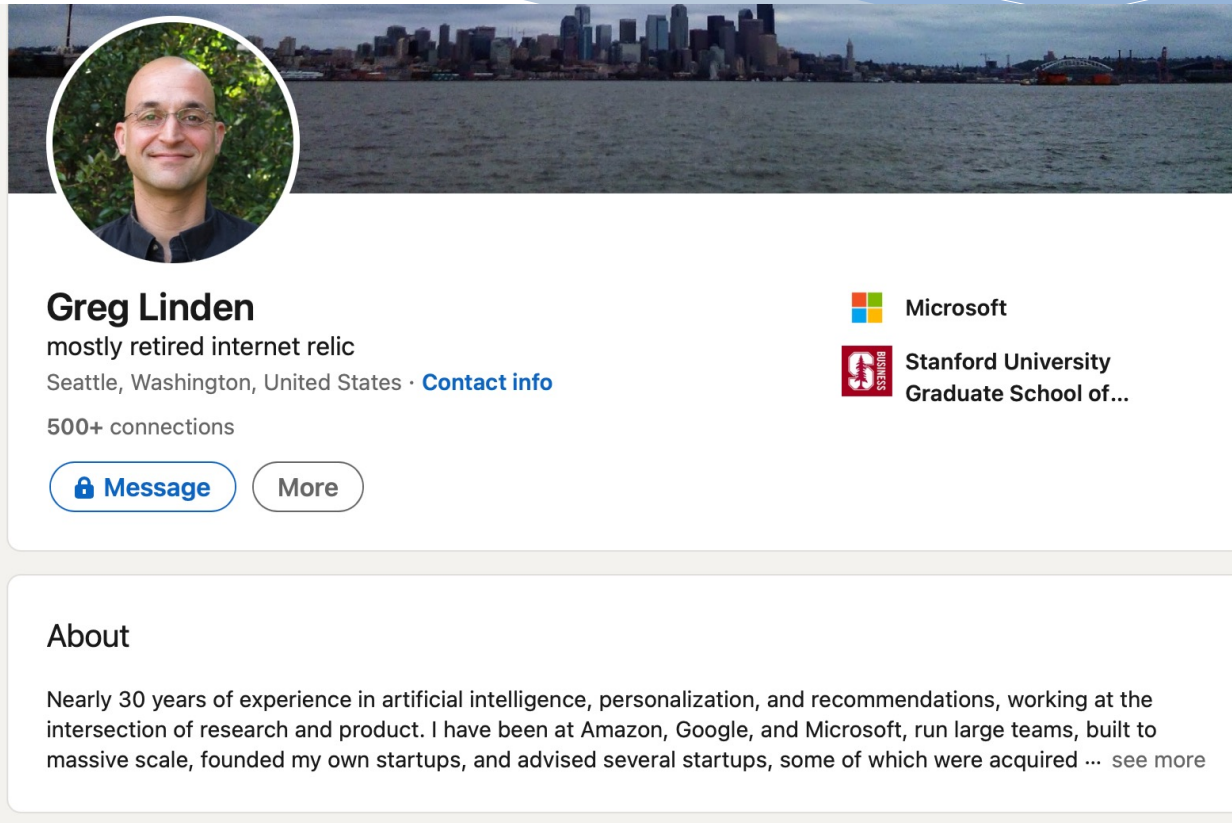
- When a packet is sent, it will travel through a series of  $N$  links and  $N-1$  forwarding devices
- A packet composed of  $L$  bits, which is transmitted through a network access card with output rate  $R$  bits/sec, will need  $L/R$  seconds to be transmitted = transmission delay
  - The total transmission delay of a packets going through  $N$  links (i.e.  $N-1$  forwarding devices) will need  $NL/R$  seconds
- The network might not be symmetric
  - The path followed by a packet from Host A to Host B can be different from the path between Host B and Host A
  - In a single forwarding device, two different ports might have different instantaneous queue size
  - Forward delay can be different to the backward delay

# Queuing delay

- The traffic intensity is defined as the ratio  $aL/R$ , where  $a$  is the average arrival rate of packets (pkts/s)
- The number of packets stored in a buffer will grow when the incoming arrival rate exceeds the output link capacity
  - $aL/R > 1$
- What about  $aL/R \leq 1$ ? Queuing delay depends on the nature of the packets arrival rate
  - If the minimum inter-arrival packet period is  $L/R$  packets, then, there is no queue
  - Packet burst (all packets arrive at the same time) will lead to queuing delay
- When a buffer becomes full, new incoming packets cannot be stored and they are dropped (leading to the so-called packet losses)



# Why Latency Matters





A screenshot of a Facebook profile for Greg Linden. The profile picture is a circular headshot of a man with glasses. The cover photo is a wide landscape image of a city skyline across a body of water. The profile name is 'Greg Linden', with a bio 'mostly retired internet relic' and location 'Seattle, Washington, United States'. It shows '500+ connections' and buttons for 'Message' and 'More'. The 'About' section lists affiliations with Microsoft and Stanford University, followed by a paragraph of experience and a 'see more' link.

**Greg Linden**  
mostly retired internet relic  
Seattle, Washington, United States · [Contact info](#)  
500+ connections  
[Message](#) [More](#)

**About**

Nearly 30 years of experience in artificial intelligence, personalization, and recommendations, working at the intersection of research and product. I have been at Amazon, Google, and Microsoft, run large teams, built to massive scale, founded my own startups, and advised several startups, some of which were acquired ... [see more](#)

 Microsoft  
 Stanford University  
Graduate School of...

At Amazon “Every 100ms delay costs 1% of sales”

# Why Latency Matters

<http://highscalability.com/latency-everywhere-and-it-costs-you-sales-how-crush-it>

Latency matters. Amazon found every 100ms of latency cost them 1% in sales. Google found an extra .5 seconds in search page generation time **dropped traffic by 20%**. A broker could lose **\$4 million in revenues per millisecond** if their electronic trading platform is 5 milliseconds behind the competition.

**Forbes**

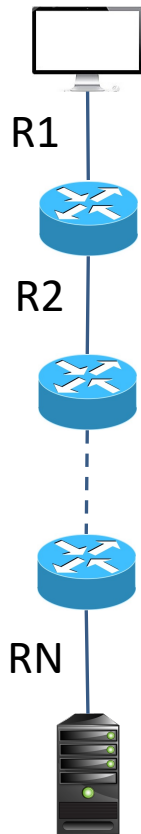
## Why Brands Are Fighting Over Milliseconds



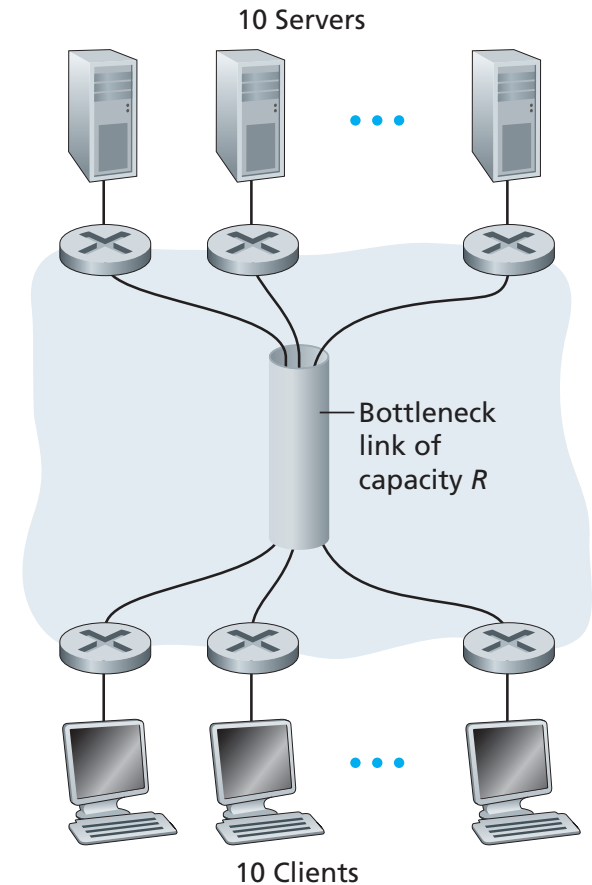
**Steve Olenski** Former Contributor ©  
CMO Network

# End-to-End throughput (client performance)

E2E average throughput =  $\min(R_1, R_2, \dots, R_N)$

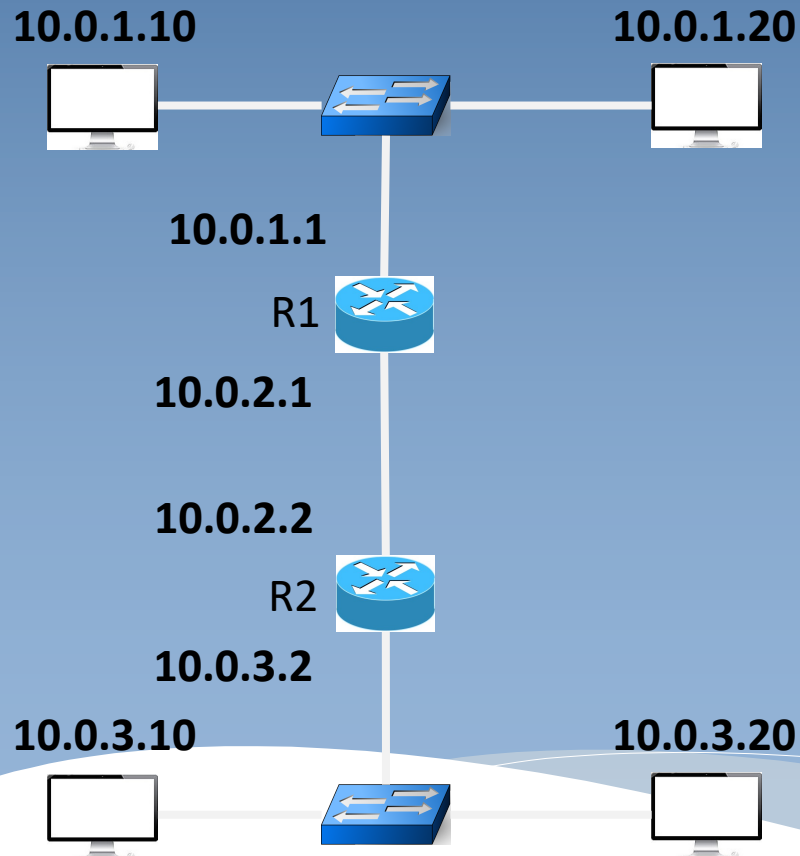


E2E average throughput = ??



# IPv4 Addressing

## Inter LAN Communication



- IP addresses identify a device in the network.
- An IP address is composed of 4 bytes and represented in decimal format with a “.” between two bytes.
- In an IP address, some bits identify the network address (network ID).
- Setting all the bits of the network ID to 1 gives the network mask (*netmask*) address

# Targeting a network device

- To identify a router or an end host, to send it a packet for instance, such a device will need an address
- IP addresses identify the subnet and indicates the host ID of a device in Internet
  - Answer to the questions: Is the device located in my LAN or somewhere else?
  - How should the packet should be routed?
- Let's talk about IPv4 address scheme

# Classful addresses

- Classful networks uses an addressing scheme where the address space is divided in 5 classes
  - Class A: 8 bits for the network ID - Netmask: 255.0.0.0
    - \* Start address: 0.0.0.0 → 00000000.00000000.00000000.00000000
    - \* End address: 127.255.255.255 → 01111111.11111111.11111111.11111111
  - Class B: 16 bits for the network ID – Netmask: 255.255.0.0
    - \* Start address: 128.0.0.0 → 10000000.00000000.00000000.00000000
    - \* End address: 191.255.255.255 → 10111111.11111111.11111111.11111111
  - Class C: 24 bits for the network ID – Netmask: 255.255.255.0
    - \* Start address: 192.0.0.0 → 11000000.00000000.00000000.00000000
    - \* End address: 223.255.255.255 → 11011111.11111111.11111111.11111111
  - Class D: multicast
    - \* Start address: 224.0.0.0 → 11100000.00000000.00000000.00000000
    - \* End address: 239.255.255.255 → 11101111.11111111.11111111.11111111
  - Class E: reserved
    - \* Start address: 240.0.0.0 → 11110000.00000000.00000000.00000000
    - \* End address: 255.255.255.255 → 11111111.11111111.11111111.11111111

# Classful addresses (cont)

- The classful address scheme led to
  - Address exhaustion
  - Big routing tables
- Note that the first and the last addresses are not used to identify a host
  - The first address (*aka the zero address*) identifies the network address
    - \* The network address of a host with @IP 134.51.12.36 is 134.51.0.0
    - \* The network address of a host with @IP 212.54.12.36 is ??
  - The last address (*the all one address*) is the network broadcast address (e.g. to send a message to an entire LAN)
    - \* The broadcast network address for 134.51.0.0 is 134.51.255.255
    - \* The broadcast network address for 212.54.12.36 is ??
  - How many host can be addressed in a network?
    - \*  $2^n - 2$  where  $n$  is the number of bits used for the host id's.



# Classless Inter-Domain Routing

- CIDR is the current addressing scheme in Internet
- The number of bits used in a network address does not depend anymore on the first bits of the IP address
  - What is the network address for 134.59.17.36?
- CIDR uses a routing prefix to identify the number of bits composing a network address
  - Ex. 134.59.17.36/20
  - For 134.59.17.36/20, what is the network address?
    - \* Write in binary format: 10000110.00111011.00010001.00100100
    - \* Take the first 20 bits and the remaining ones set it to zero:  
10000110.00111011.00010000.00000000
    - \* Network address: 10000110.00111011.00010000.00000000 =  
134.59.16.0

# Classless Inter-Domain Routing (cont)

- For 134.59.17.36/20
  - What is the netmask?
    - \* Write the address in binary format: 10000110.00111011.00010001.00100100
    - \* Take the first 20 bits and set it to 1. Put the remaining ones to zero: 11111111.11111111.11110000.00000000
    - \* Write in decimal: 255.255.240.0
  - what is the network broadcast address?
    - \* Write in binary format: 10000110.00111011.00010001.00100100
    - \* Take the first 20 bits and set the remaining bits to one: 10000110.00111011.00011111.11111111
    - \* Write the value in decimal: 134.59.31.255
  - What is the first available IP address for a host?
    - \* Take the network address and put the last bit to one: 134.59.16.1
  - What is the last available IP address for a host?
    - \* Take the network broadcast address and set the last bit to zero: 134.59.31.254

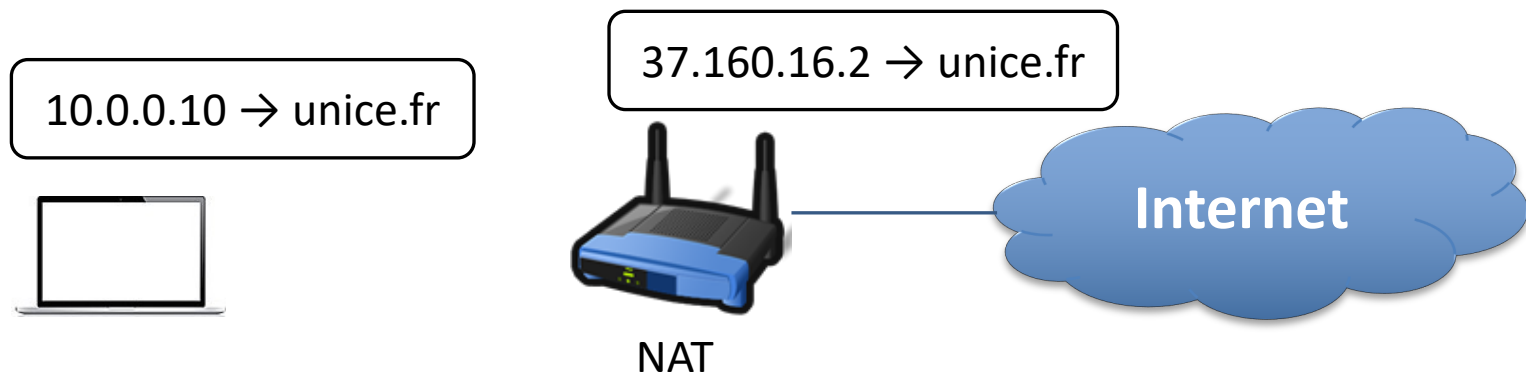
# Subnetting

- Divide a network address into subnets
  - Take the minimum number of bits according to your needs
- Example, divide the 192.168.1.0/24 network into 4 subnets
  - 4 subnets => 4 different subnet identifiers (addresses)
  - 2 bits to create 4 different identifies
    - \* Red numbers are given in binary
    - \* 192.168.1.0/26 (192.168.1.0000 0000)
    - \* 192.168.1.64/26 (192.168.1.0100 0000)
    - \* 192.168.1.128/26 (192.168.1.1000 0000)
    - \* ??

# Public vs Private IP addresses

- IP addresses can be public or private
  - Public addresses can be routed through the Internet
  - Private addresses must never be routed through the Internet
- Devices with an IP private address uses Network Address Translators (NATs) to reach the Internet
- IP addresses are in the following range
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16

# Example with private addresses



# Configuring Ethernet network interface cards in Linux through command line

- To display the addresses of network interfaces, use “ip a s”
- To assign an IP address and netmask to a given interface. E.g. “ip a add 10.0.0.1/24 dev eth0”
- To remove an IP address from a network interface, you can use “ip a del 10.0.0.1/24 dev eth0”
  - In absence of the CIDR notation, the “ip” command will assume a /32 network mask length.

# Routing

# Basics on routing

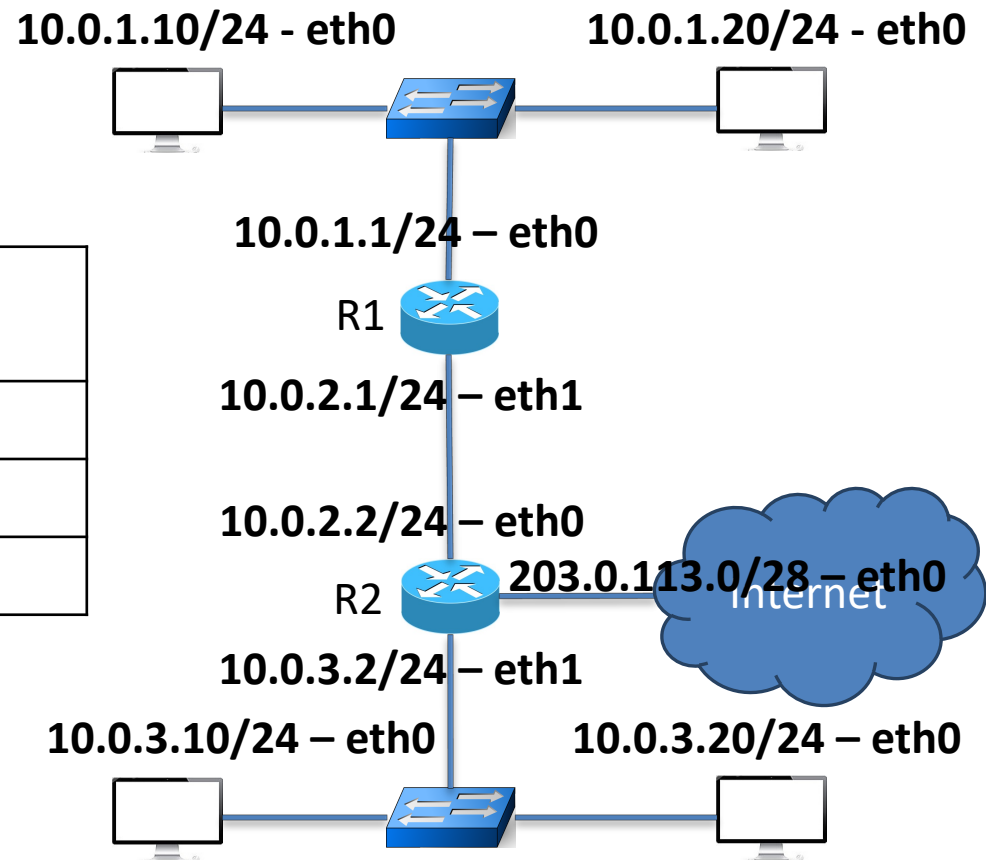
- Each local network has one or more gateways (routers)
- Each link of a router belongs to a different network address
- To reach an external network (external hosts), hosts and routers must keep a routing table
  - Compression through default routes
    - \* The smallest routing table
  - Routes provides the cheapest path to the destination
    - \* E.g. in hop numbers to go through



# Routing table of an end host

Routing table of host 10.0.1.10  
*Note the use of a default route*

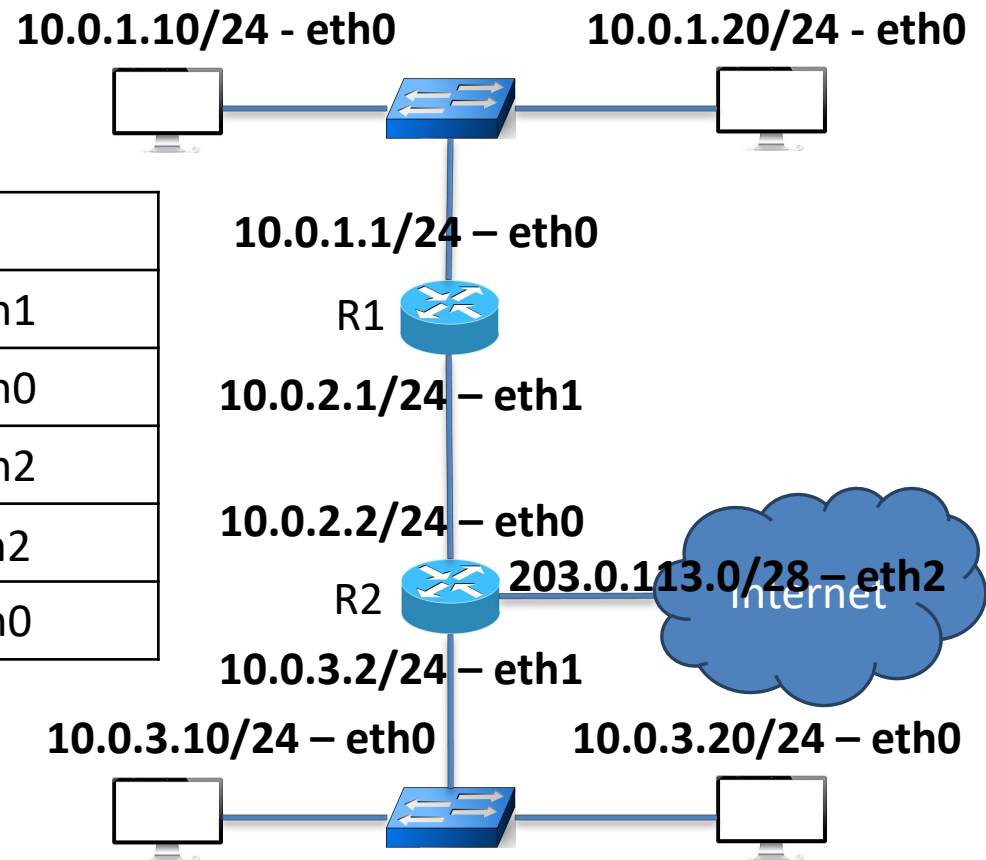
Target Network	IP Gateway	Device
10.0.1.0/24	*	Eth0
<b>10.0.2.0/24</b>	<b>10.0.1.1</b>	<b>Eth0</b>
*	10.0.1.1	Eth0



# Routing table of a router

Routing table of router R2  
*Note the use of a default route*

Target Network	IP Gateway	Device
10.0.3.0/24	*	Eth1
10.0.2.0/24	*	Eth0
203.0.113.0/28	*	Eth2
*	203.0.113.14	Eth2
10.0.1.0/24	10.0.2.1	Eth0



# Choosing routes

- In general, you must
  - Minimize the routing entries
    - \* Take the *best* default route
  - Minimize the distance between two networks

# Statics route in a Linux box

R2 is a Linux box. We want to build its routing table

- To display the routing table “ip r s”
- Attached networks (e.g. 10.0.3.0/24) are added by default
- Add route to 10.0.1.0/24
  - ip r add 10.0.1.0/24 via 10.0.2.1 dev eth0
- Add the default route
  - ip r add default via 10.0.2.1 dev eth0
- Remove a route (if needed)
  - ip r del 10.0.1.0/24 via 10.0.2.1 dev eth0

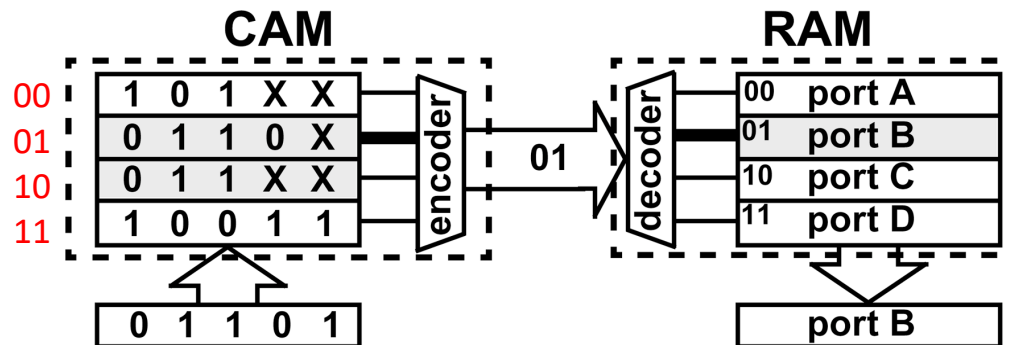
Routing table at R2

Target Network	IP Gateway	Device
10.0.3.0/24	*	Eth1
10.0.2.0/24	*	Eth0
203.0.113.0/28	*	Eth2
*	203.0.113.14	Eth2
10.0.1.0/24	10.0.2.1	Eth0

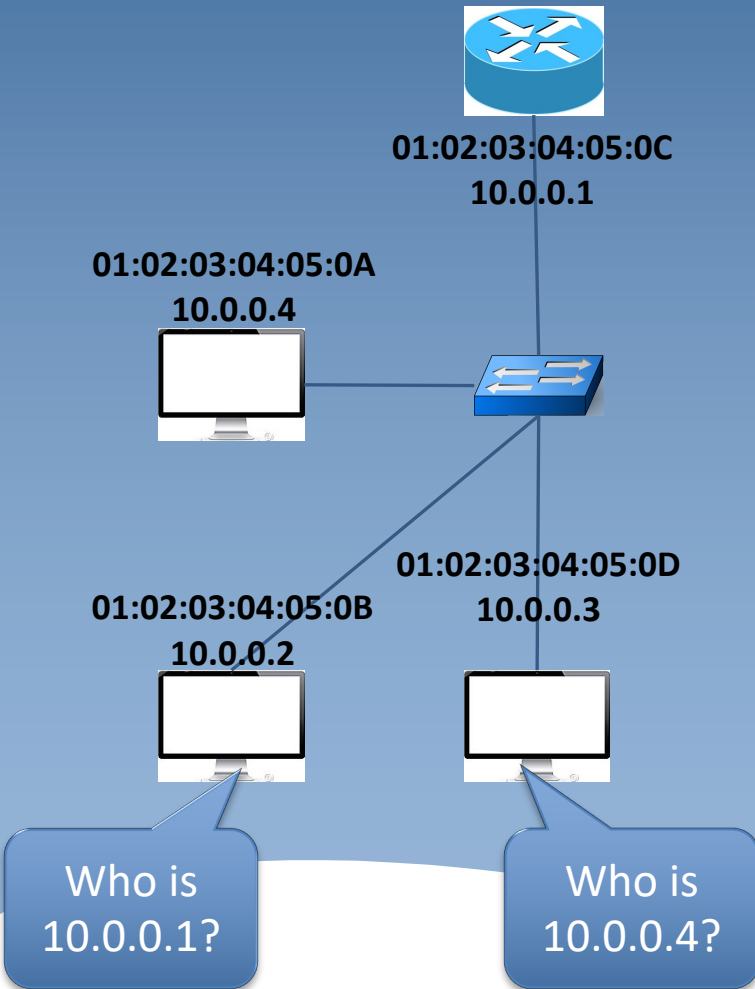
# The longest prefix matching

EXAMPLE ROUTING TABLE

Entry No.	Address (Binary)	Output Port
1	101XX	<i>A</i>
2	0110X	<i>B</i>
3	011XX	<i>C</i>
4	10011	<i>D</i>



# Layer 2 Addressing and Switching

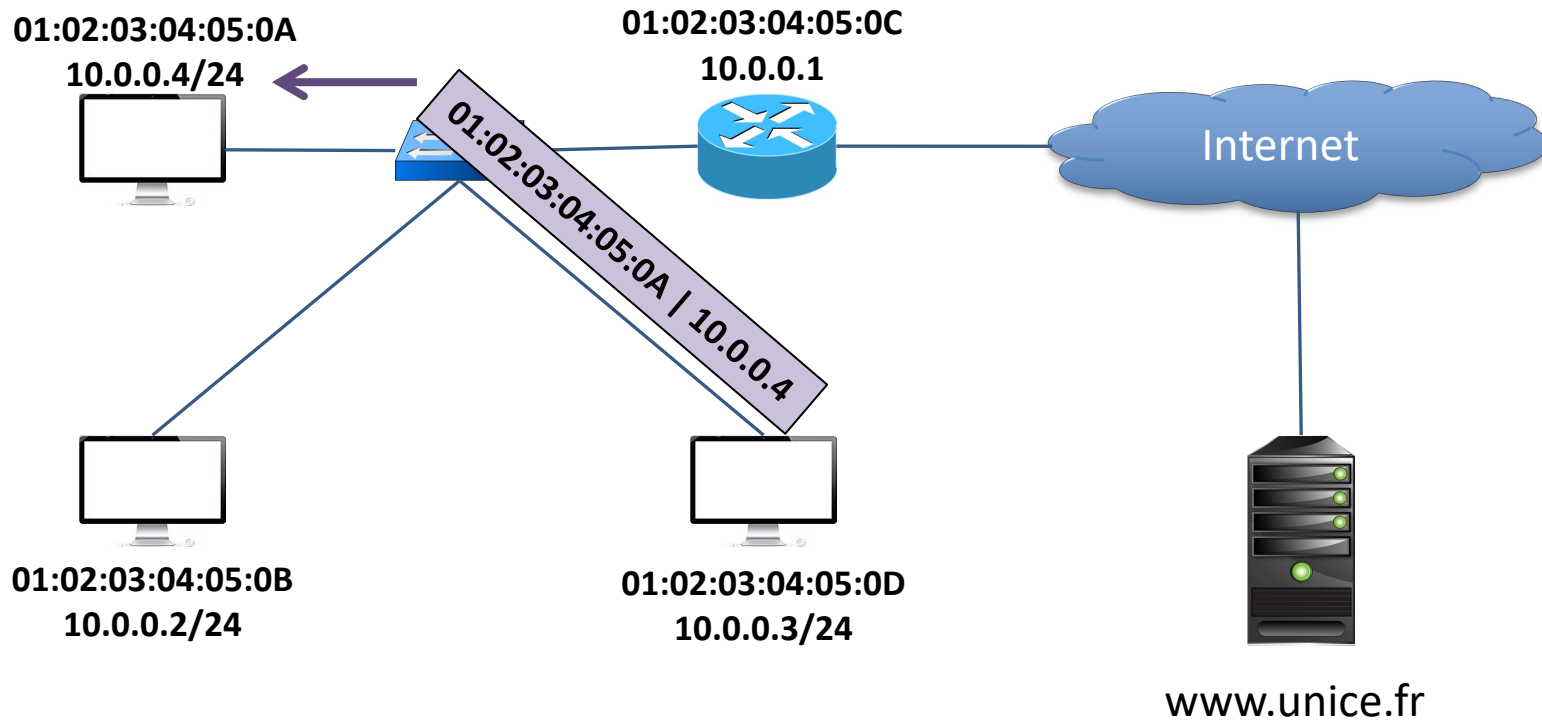


## Intra LAN Communication

Intra LAN communication needs a MAC address to know which device is configured with a given IP address

A MAC address is composed of 6 bytes and represented in hexadecimal with ":" or "-" between two bytes

# Example 1: 10.0.0.3 → 10.0.0.4

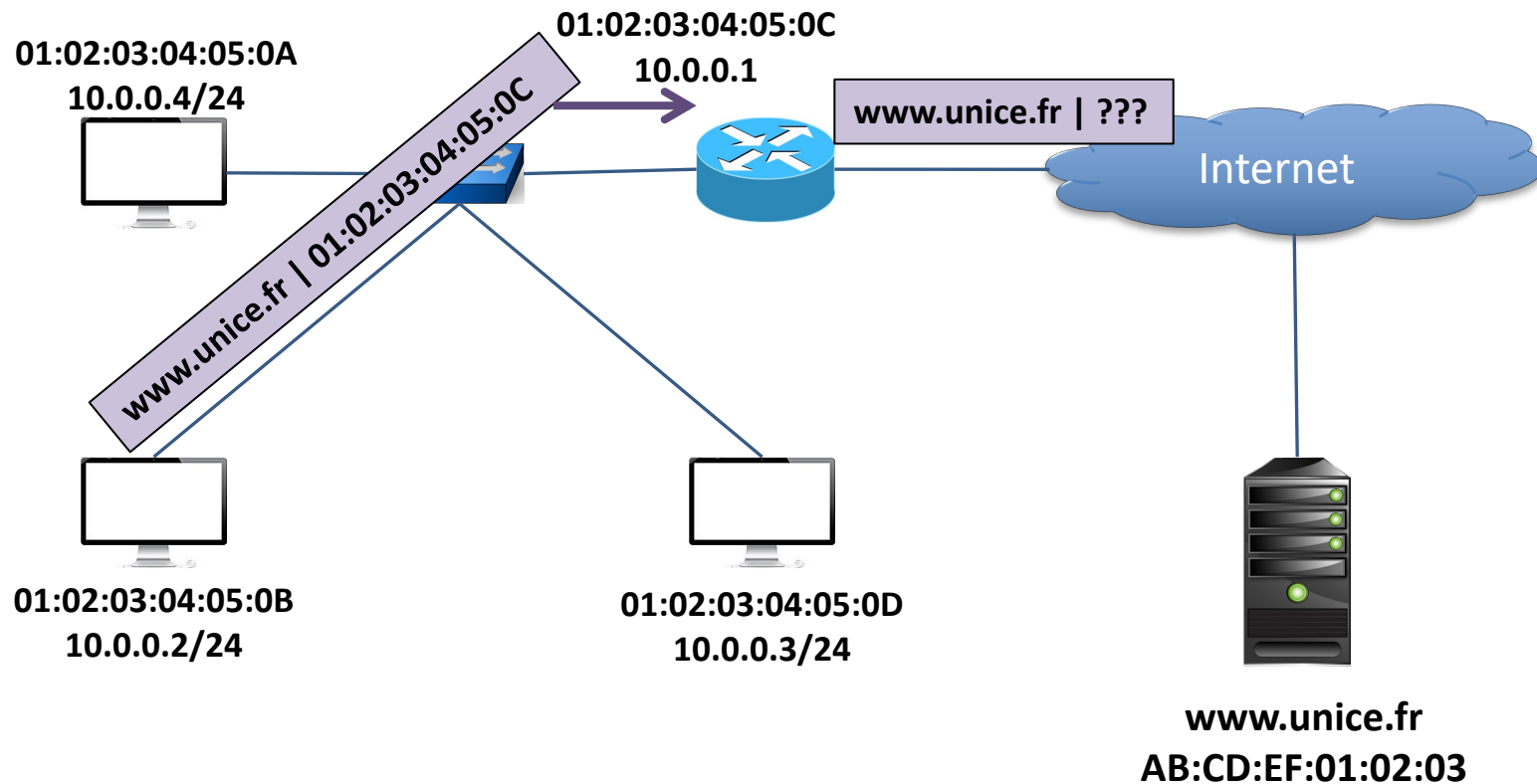




# Overview of Layer 2 devices

- Switches ports do not have IP addresses
- How does a switch do to forward packets?
  - Switches have a forwarding database (FDB)
  - The forwarding database associates the port number with the MAC address of the host plugged in that port
  - Switches learn the MAC addresses when the host connected to it sends a packet
  - If the MAC destination address is already in the FDB, the switch forward the packet in the right port. Otherwise, the packet is sent through all the ports, except the one where the packet comes from.
  - Packets with the broadcast MAC address are sent through all the ports except the one where the packet comes from.

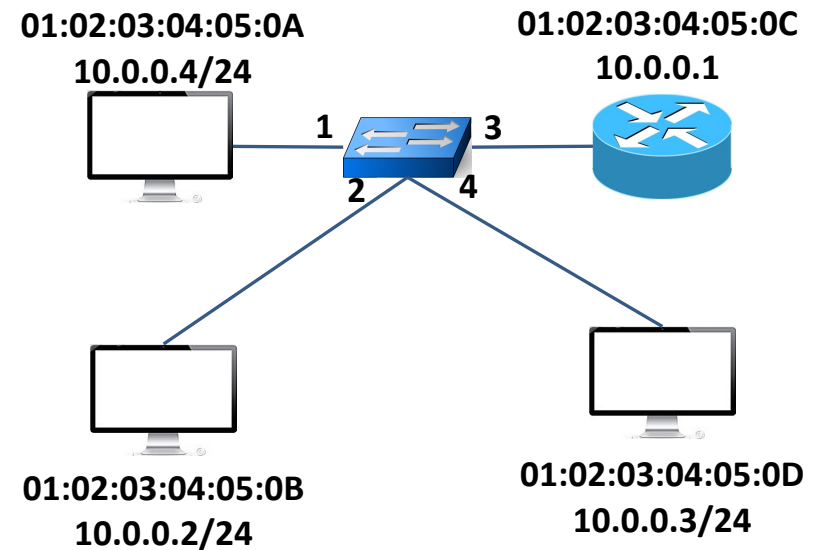
# Example 2: 10.0.0.2 → www.unice.fr



# Forwarding table in a switch

- Expected forwarding database (FDB) of the switch

@MAC	#Port
01:02:03:04:05:0A	1
01:02:03:04:05:0B	2
01:02:03:04:05:0C	3
01:02:03:04:05:0D	4



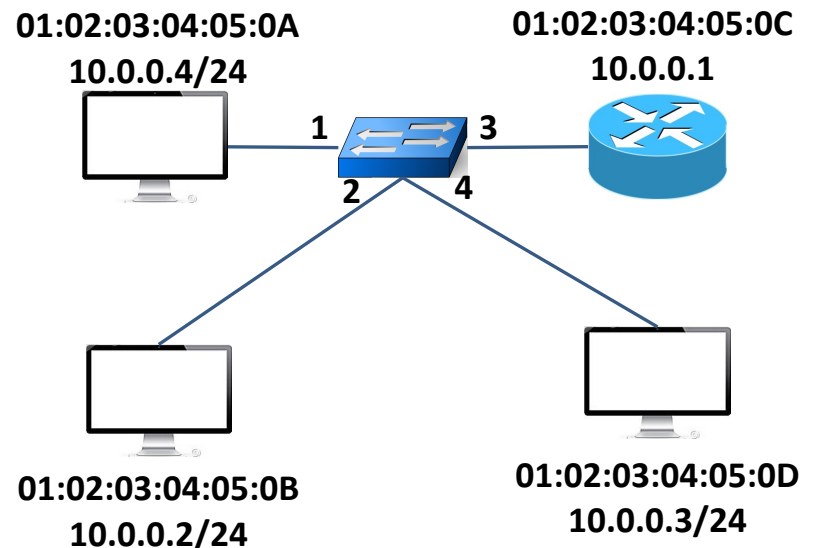
# The Address Resolution Protocol (ARP)

- Suppose that you know the IP address of the targeted device inside your LAN, so how do you get its MAC address?
  - This is done by the ARP
  - You don't get the MAC address of a device from another LAN (i.e. on the other side of the gateway)
- When a host needs to know the MAC address of a host inside the LAN, it broadcasts an ARP request
  - In the request, you will find the IP address of the targeted host.
- Every device will take the ARP request. The one that will see its IP address inside must send an ARP reply
  - The ARP reply contains the IP address of the sender and its MAC address
  - Applications can now communicate
- ARP is a Link-Layer protocol

# ARP table

- Expected ARP table at host 10.0.0.4
  - Communication with the router and host 10.0.0.3
  - Add route to 10.0.0.2 “arp -s 10.0.0.2 01:02:03:04:05:0B”
  - Delete route to 10.0.0.5 “arp -d 10.0.0.5”
  - And show the ARP table “arp -n”

@IP	@MAC	dev
10.0.0.3	01:02:03:04:05:0D	eth0
10.0.0.1	01:02:03:04:05:0C	eth0
10.0.0.2	01:02:03:04:05:0B	eth0



# Some special addresses

- To send a single message to the entire LAN, you must send a broadcast messages
  - The IP broadcast address is 255.255.255.255
  - The MAC broadcast address is FF:FF:FF:FF:FF:FF
- The IP loopback address, which allows to communicate two processes in a single machine by mean of the network protocol stack.
  - Packets with this destination never leaves the host
  - By default, 127.0.0.1
  - But defined to be 127.0.0.0/8 by RFC6890