# International Women's Day (March 8)

- International Women's Day (March 8) is a global day celebrating the social, economic, cultural, and political **achievements of women**.
- The day also marks a **call to action** for accelerating women's equality.
- IWD has occurred for well over a century, with the first IWD gathering in 1911 supported by over a million people. Today, IWD belongs to all groups collectively everywhere. IWD is not country, group or organization specific.


https://www.internationalwomensday.com

# Tomorrow 8 March 2025

- Collectively, we can Accelerate Action for **gender equality**.
- At the current rate of progress, it will take until 2158, which is roughly five generations from now, to reach full gender parity, according to data from the World Economic Forum.
- Focusing on the need to Accelerate Action emphasizes the importance of taking swift and decisive steps to achieve **gender equality**.
- As individuals, we can all take steps in our daily lives to **positively impact women's advancement**.
- We can call out stereotypes, challenge discrimination, question bias, celebrate women's success, and so much more. Additionally, sharing our knowledge and encouragement with others is key.


# Practical Network Defense

*Master's degree in Cybersecurity 2024-25*

# Network traffic monitoring and Wireshark lab

*Angelo Spognardi*
*spognardi@di.uniroma1.it*
*Dipartimento di Informatica*
*Sapienza Università di Roma*

# Layering Concepts

- The communication between the hosts in the network is organized in tasks, each assigned to a **layer**
- Each layer:
  - offers a service (a host of facilities) to the "Users" in the layer above
  - exploits the services offered the layer below
- The task of a level involves the exchange of messages that follow a set of rules defined by a **protocol**.
- Example:
  - Layer (N - 1) provides an insecure service in which data can overheard by unauthorized persons.
  - Protocol of level N specifies that messages sent via (N - 1)-service are encrypted with symmetric encryption.
  - Layer N offers a secure, confidential service.

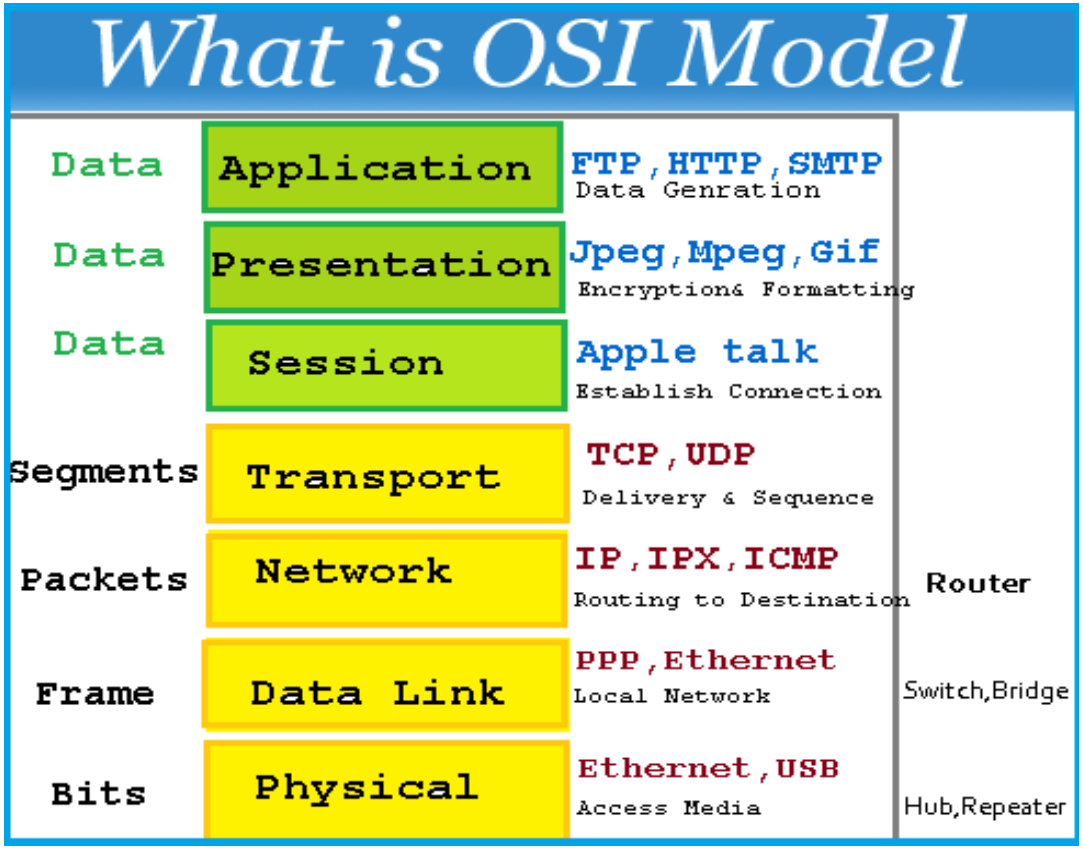
# Encapsulation/decapsulation

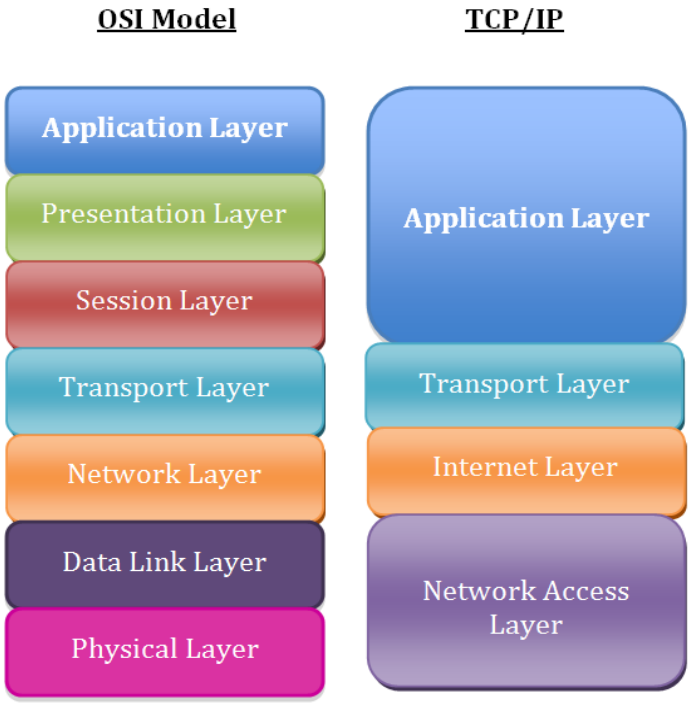
- The data to be transferred from the application layer to application layer over a network.
- Each layer adds some protocol information and provides data to the layer below.
- The physical layer (bottom) sends data over the physical medium to the destination.
- The physical layer in the destination sends the data up the "stack".
- Each protocol in the destination reads the appropriate protocol information and forwards the data to the layer above.

# 2 layered architectures

- ISO/OSI model: based on a reference model with 7 layer.
- TCP/IP model: created by the IETF, based on a reference model with 4 layers.
  - The lower TCP/IP layer is often split in 2 layers.
- Common idea: **packet switched network**

# Architecture comparison

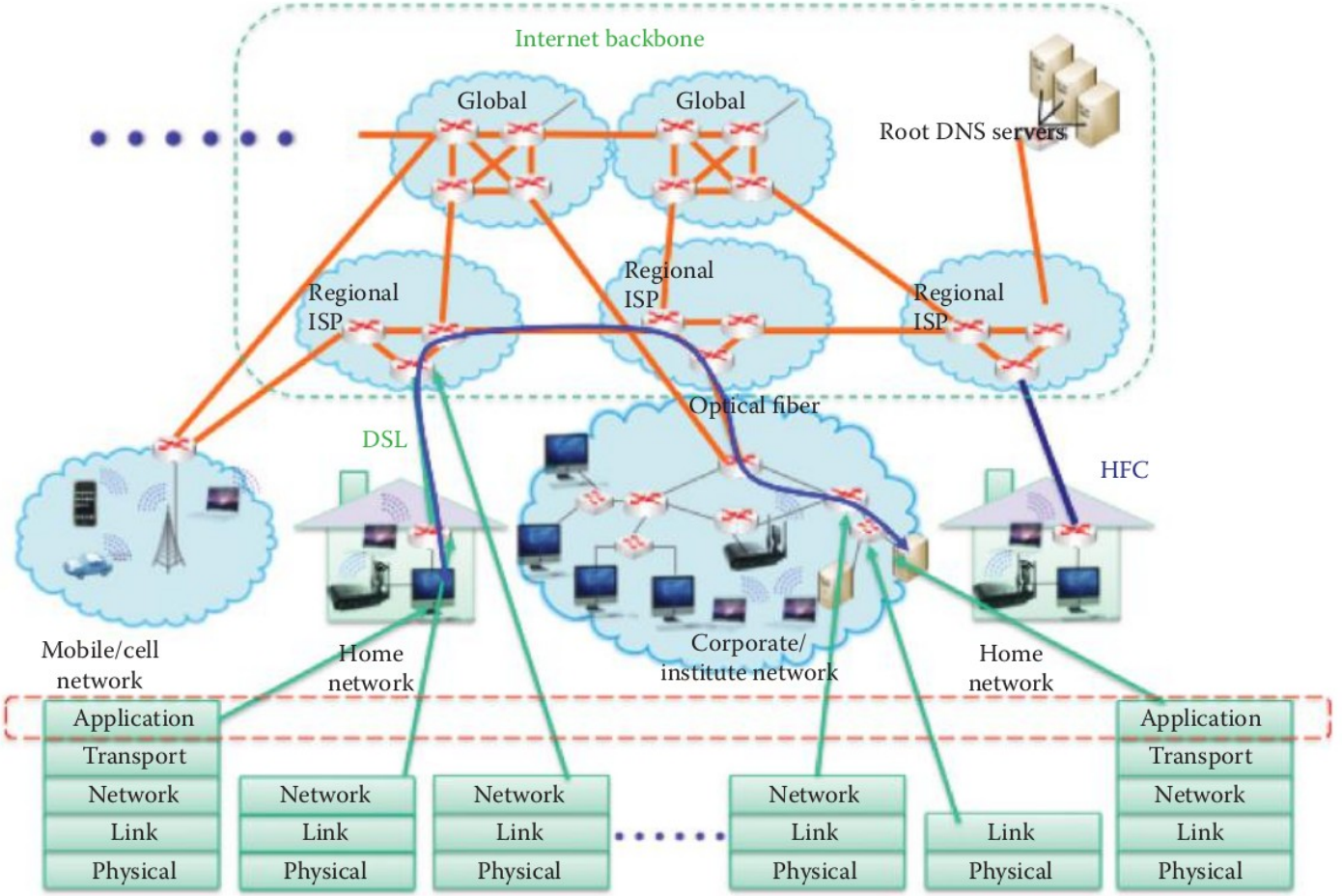



# TCP / IP model

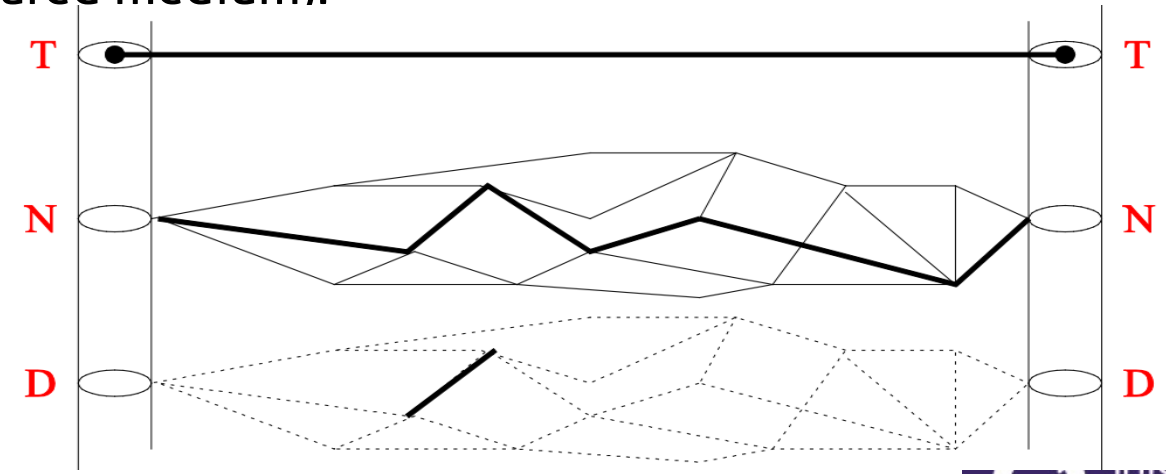
- Application layer: Corresponds to the top three layers of the OSI model.
  - Protocols: SMTP (sending e-mail), HTTP (web), FTP (file transfer), and others
- Transport layer: Equivalent to Layer 4 (Transport) of the OSI model
  - Protocols: TCP, UDP
- Internet: Equivalent to layer 3 (network) of the OSI model.
  - Protocols: IP, ICMP, IPSec
- Datalink: Equivalent to layer 2 (data link) of the OSI model.
  - Protocols: Ethernet, WiFi, ARP, etc.
- Physical layer: Equivalent to Layer 1 (Physical) of the OSI model.
  - NOTE: Datalink + physical layers are known as Network access layer.

# Client-server communication example



# Layer ideal representation

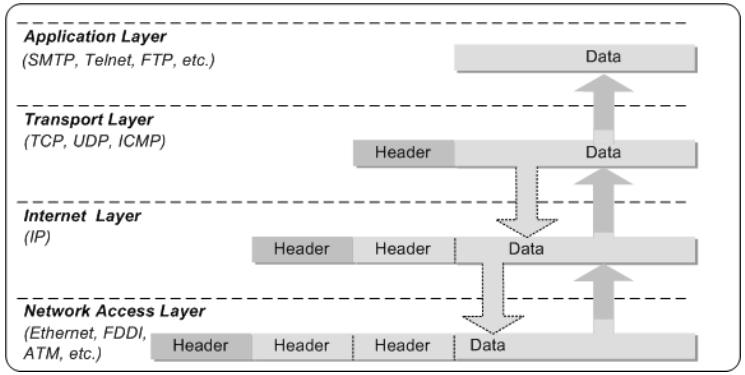
- **Transport**: the illusion of direct end-to-end connection between processes in arbitrary systems.
- **Network**: transferring data between arbitrary nodes.
- **Data Link**: transferring data between directly connected systems (via direct cable or shared medium).

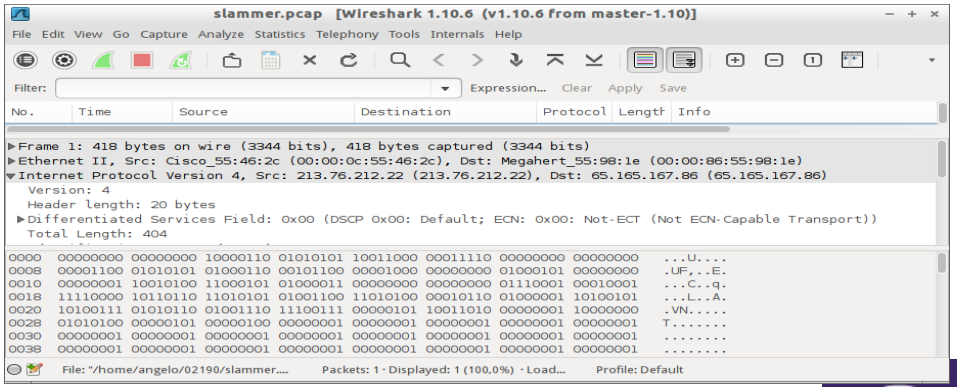


# Addresses in the architectures

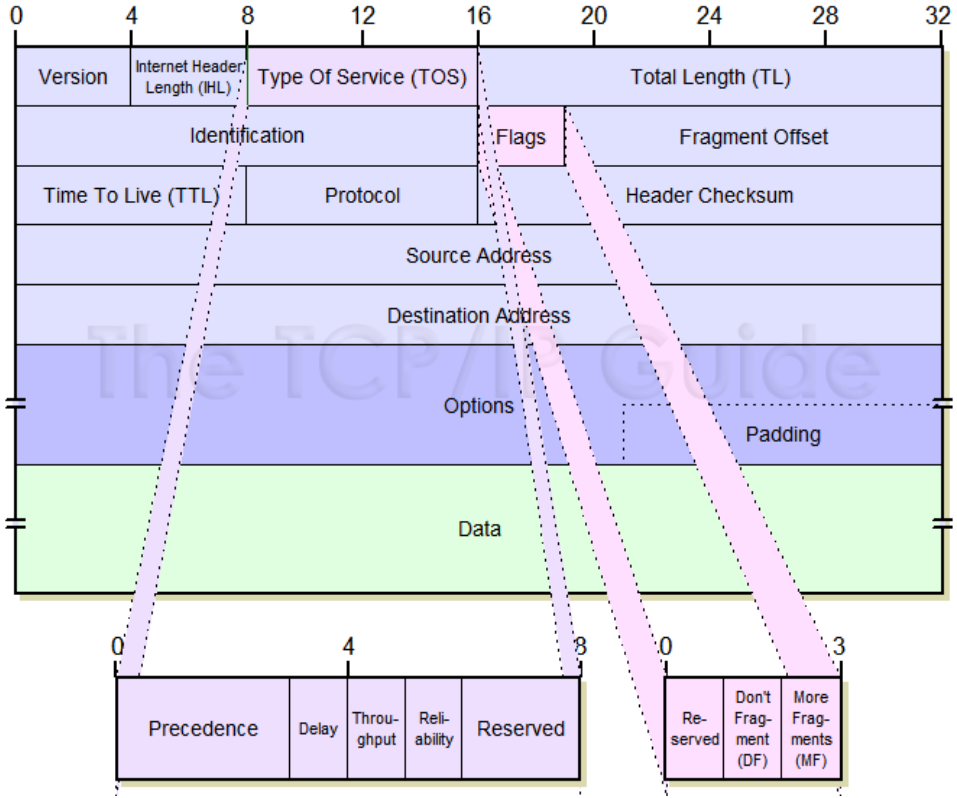
- Each layer has a type of address:
  - Application layer: Internet name, eg. www.*sapienza.it*
  - Transport layer: Port number, in the range [0..65535] that identifies the client or server. For example 80 for HTTP server.
  - Internet layer: IP address that identifies a network card, for example 151.100.17.4
  - Datalink layer: MAC address, also identifies a network cards, for example 49:bd:d2:c7:56:2a

# Encapsulation in TCP/IP

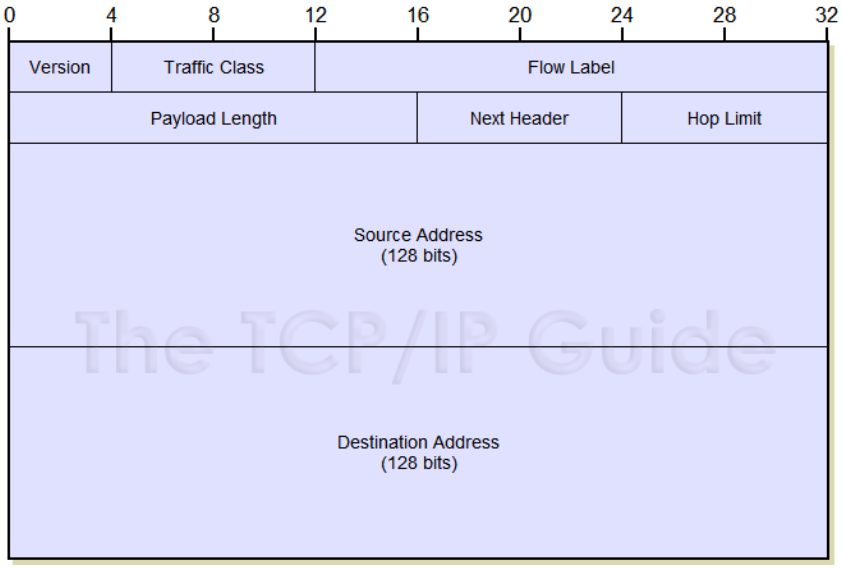


# IP packets




http://www.tcpipguide.com/free/t_IPDatagramGeneralFormat.htm

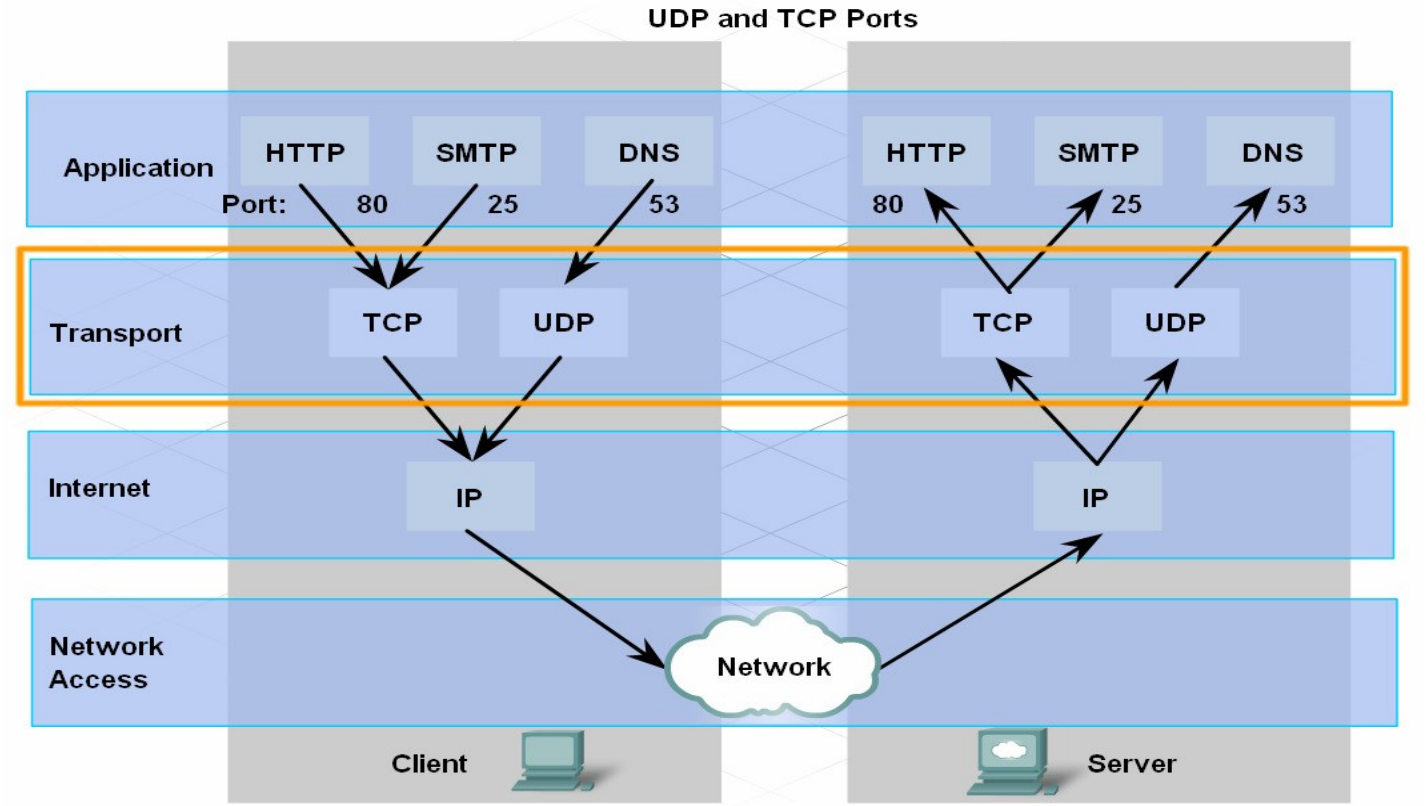



http://www.tcpipguide.com/free/t_IPv6DatagramMainHeaderFormat.htm

# Ports

- Range [0..65535]
- Source port: randomly chosen by the OS
- Destination port determines the required service (application)
  - Assigned Ports [0..1023] are said "well-known ports" and used by servers for standard Internet applications:
    - 25: SMTP (sending mail)
    - 80: HTTP (web)
    - 143: IMAP (pick-up of mail)
  - Ports [1024..49151] can be registered with Internet Application Naming Authority (IANA)
  - Ports [49152..65535] ephemeral ports

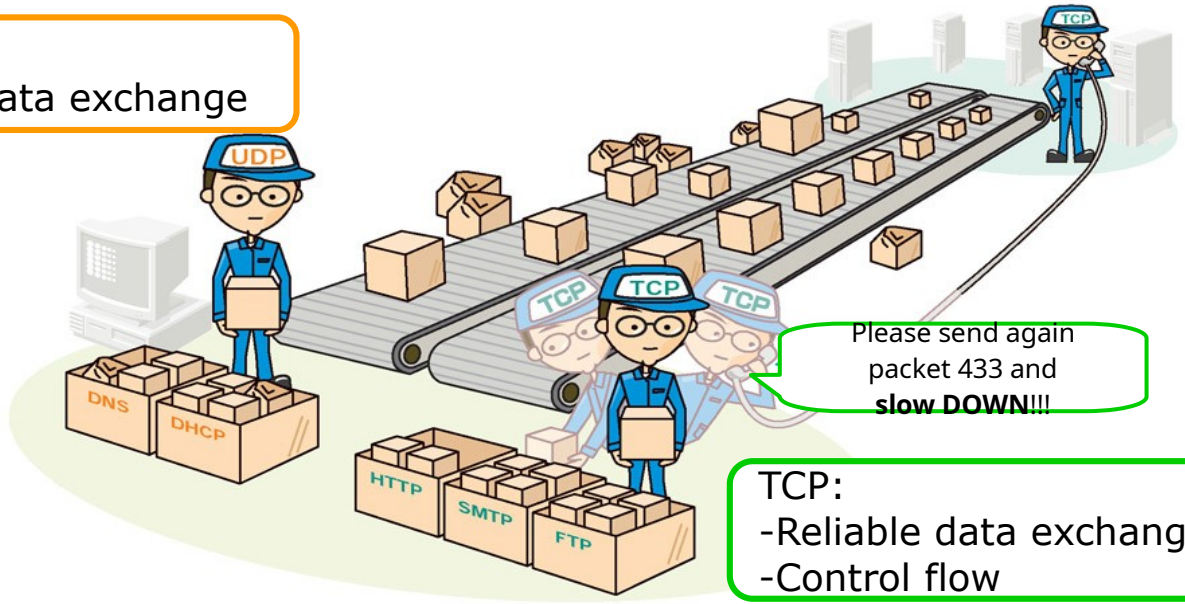
# Transport layer: TCP and UDP



# TCP vs UDP

- Connection vs Connection-less

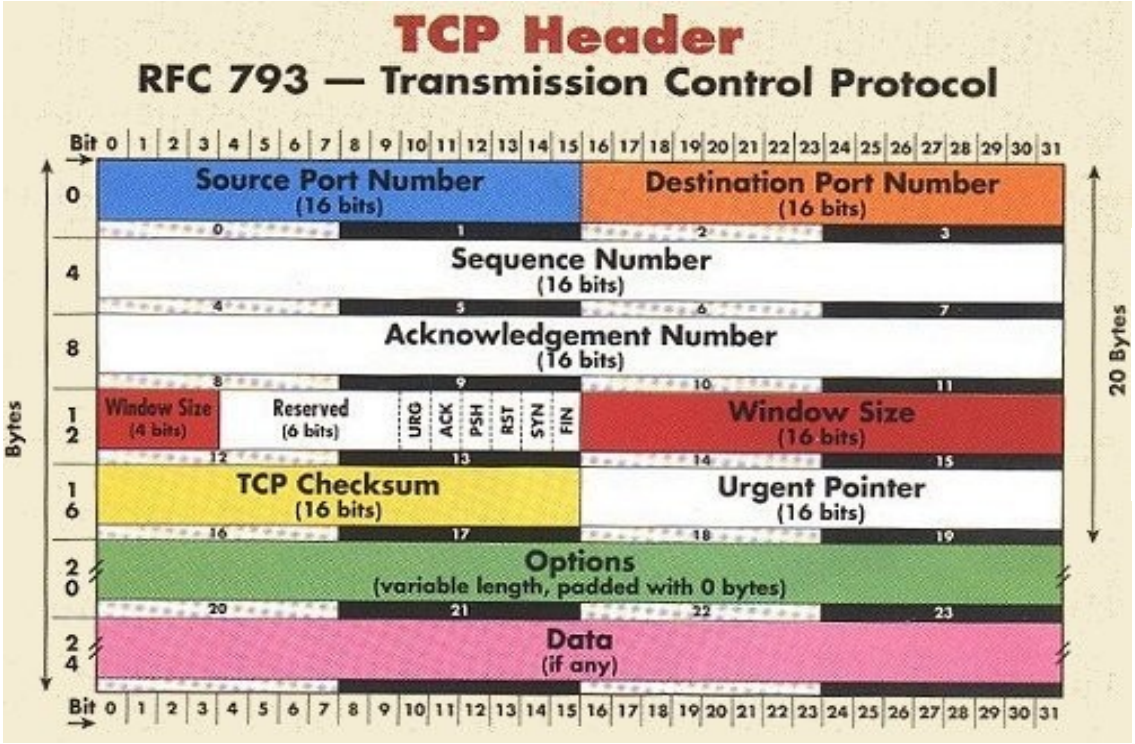
UDP:
-No control on data exchange

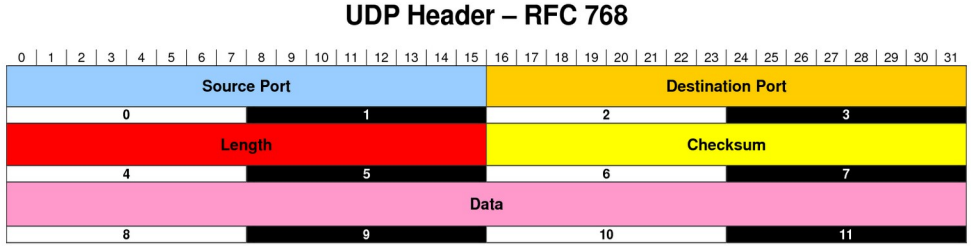



TCP:
-Reliable data exchange
-Control flow

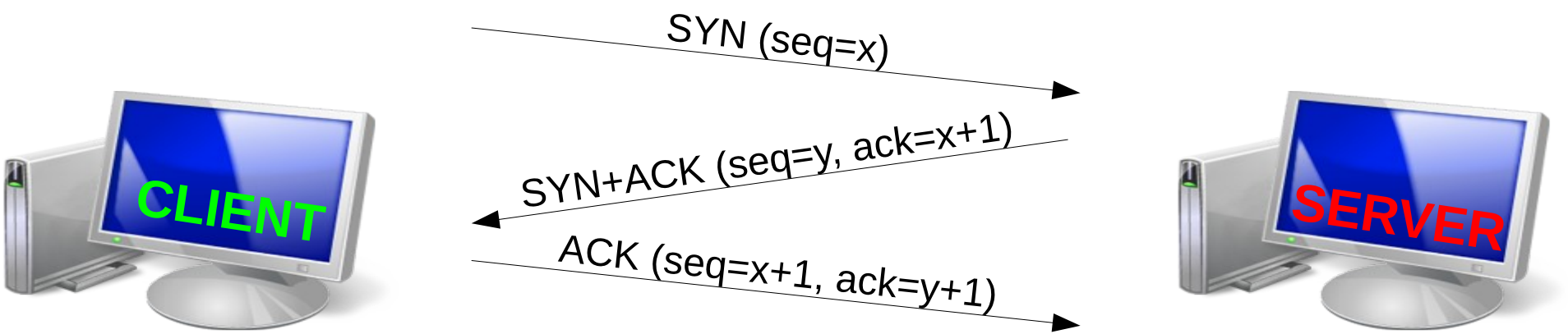
http://itpro.nikkeibp.co.jp/article/lecture/20070305/263897/

# TCP header vs UDP header



# TCP connection handshake



# Services relying on TCP

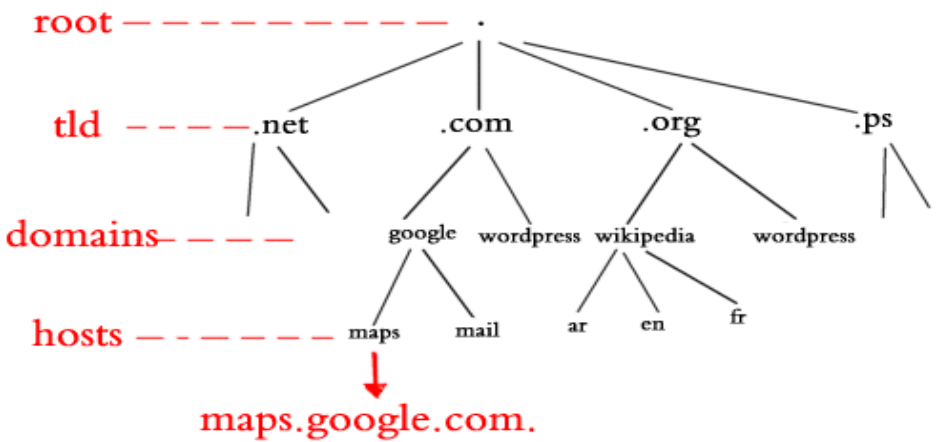
- FTP on port 20 and 21
- SSH on port 22
- Telnet on port 23
- SMTP on port 25
- HTTP on port 80
- IMAP on port 143
- SSL on port 443

# Services relying on UDP

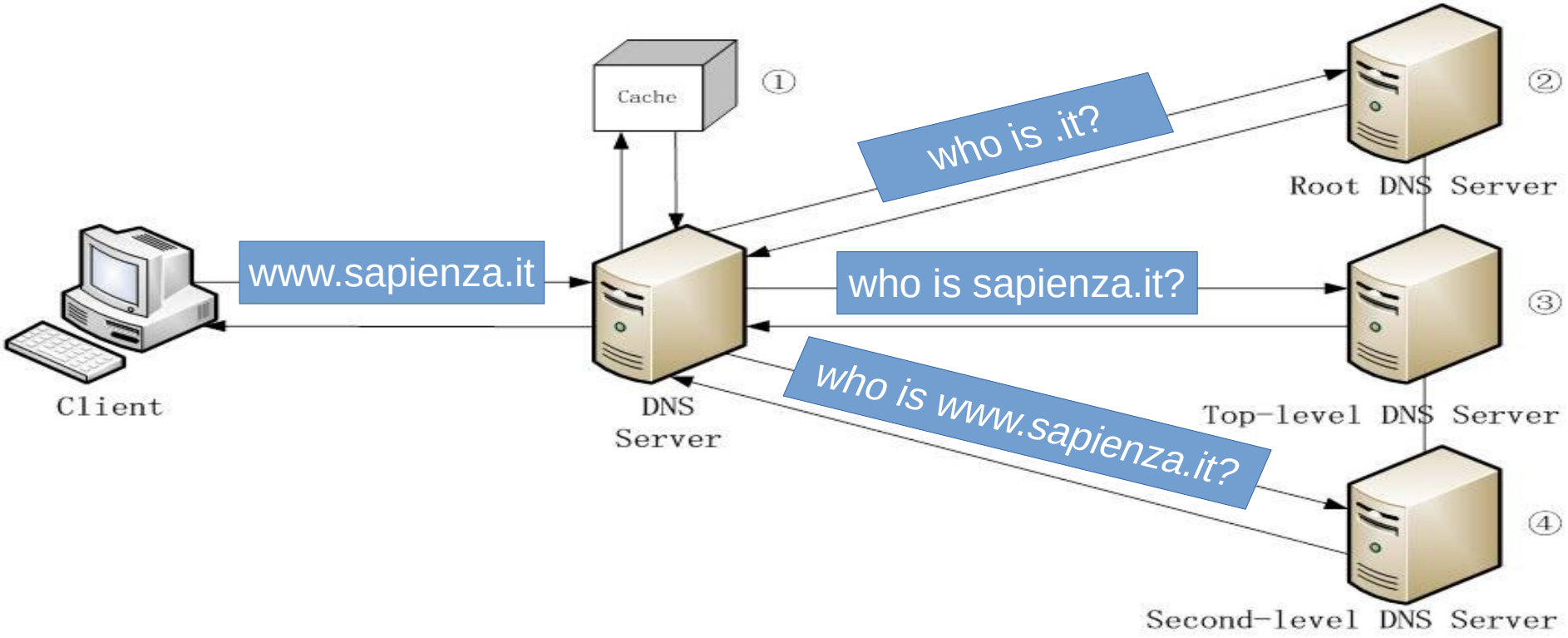
- DNS on port 53
- DHCP on ports 67 and 68
- TFTP on port 69
- SNMP on port 161
- RIP on port 520

# DNS

- A service to get the IP address from an human-friendly domain name, like www.sapienza.it
- Hierarchy of entities responsible for domain names



# DNS query example



# Dive into packets

# Capture packets

- Packets flow in the network, to capture them use a network traffic dump tool, like:
  - **dumpcap**
  - **wireshark/tshark** (https://www.wireshark.org/docs/)
  - **tcpdump**
- All based on the *pcap* (*winpcap* in Windows) library
- All of them can visualize and save the captured data
- Wireshark and tcpdump can also **analyze** *(decode)* the captured packets

# Wireshark

- Data from a network interface are "dissected" in frames, segments, and packets, understanding where they begin and end
- Then, they are interpreted and visualized in the context of the recognized protocol
- **Promiscuous mode** (also called monitor mode) is required to capture packets not intended for the capturing host
- Best suited for
  - Looking for the root cause of a known problem
  - Searching for a certain protocol or stream between devices
  - Analyzing specific timing, protocol flags, or bits on the wire
  - Following a conversation between two devices
- It shouldn't be the first tool thought of early on in discovering a problem, but solving a problem...

# Logic of wireshark

- Frames are collected from the interface and passed to several, consecutive, “dissectors”, one for each layer
- Frames pass from bottom layer to upper layer
- Protocols can be detected in two ways:
  - directly, if a frame (e.g. Ethernet) has the field that states which protocol it is encapsulating
  - indirectly, with tables of protocol/port combinations and heuristics
    - Usually working, troubles when protocols are used in nonstandard ports

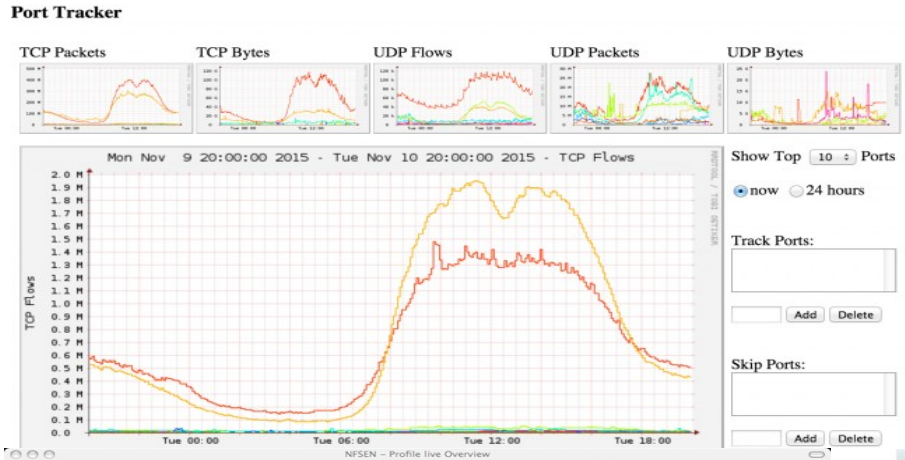
# Alternative way to capture traffic info

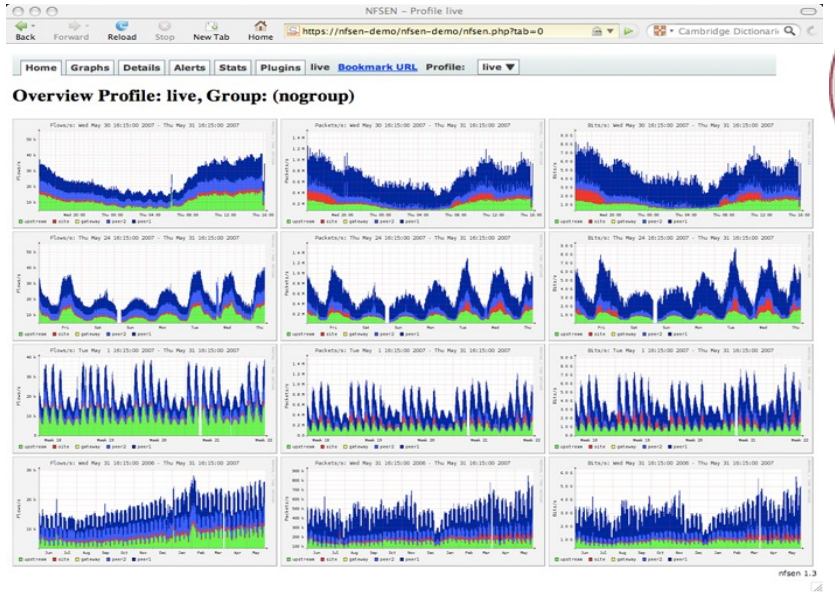
- Traffic represented as “connections”
- Netflow
  - For statistics and monitoring
  - Netflow v9 https://www.ietf.org/rfc/rfc3954.txt
- Zeek (formerly known as Bro)
  - Framework for traffic inspection and monitoring
  - Scripting engine to enable immediate processing

# Netflow

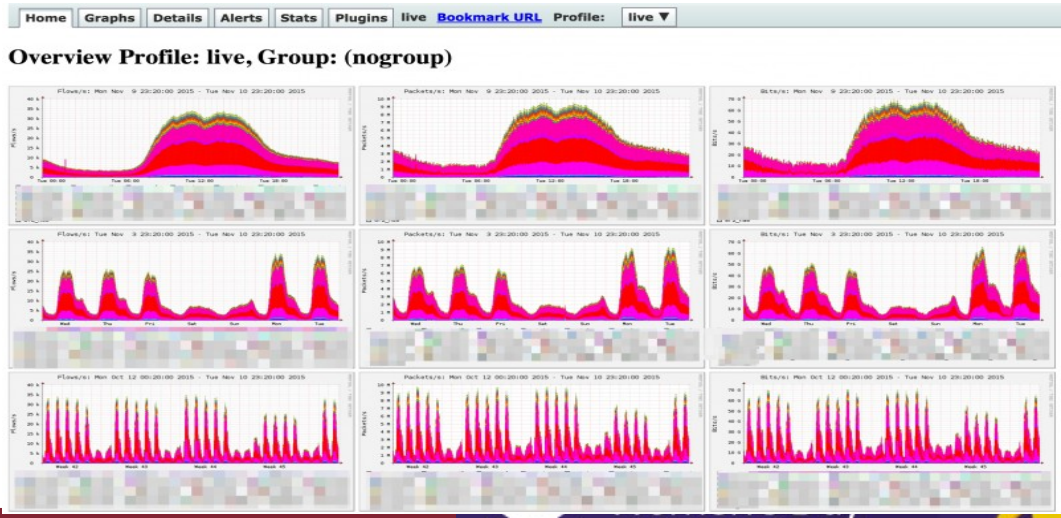
- Suite of tools:
  - nfcapd
    - Capture and save netflows
  - nfdump
    - Analyze netflow files (tcpdump-stlye)
  - nfsen
    - Graphical tool to access captured netflows
      - It uses nfdump as back end

# Nfsen








# Summary

- Packets: made of stacked layers
  - Each layer has its own role in the communication
- Encapsulation is the rule
  - Protocol encapsulates other protocols as their data
- How to capture packets?
  - It depends on the final goal: monitoring, statistics, debugging, security
- Wireshark: dissecting packets
  - Very powerful tool to explore and dive into packets

# Wireshark activity

# Using wireshark

- Capturing is way too easy… Too many packets!
  - https://wiki.wireshark.org/CaptureSetup/CapturePrivileges
- To survive, use filters!
  - They allow to only focus on requested packets or certain activity by network devices
- Two kinds of filters: **display filters** and **capture filters**
  - Capture filters to limit the amount of network data that goes into processing and is getting saved
  - Display filters to inspect only the packets you want to analyze once the data has been processed

# Capture filters – wireshark/tcpdump

- Limit the traffic captured and, optionally, analyzed
  - Packets not captured are lost…
- Berkeley Packet Filter (BPF) syntax (man pcap-filter)

  protocol direction type

  - Protocol: ether, tcp, udp, ip, ip6, arp
  - Direction: src, dst
  - Type: host, port, net, portrange
  - Other primitives: less, greater, gateway, broadcast
  - Combinations with operators: and (&&), or (||), not (!)

# Display filters – wireshark

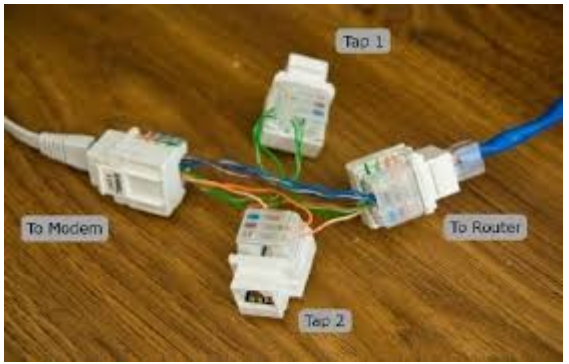
- Display only captured packets matching the filters
  - Packets are not discarded or lost
- Easy but refined syntax: only packets evaluating true are displayed
  - Comparison operators
  - Filters use types (strings where numbers are required return errors)
  - Common logical operators
- Filters can be built interacting with the packets

# Logic of wireshark

- Frames are collected from the interface and passed to several, consecutive, “dissectors”, one for each layer
- Frames pass from bottom layer to upper layer
- Protocols can be detected in two ways:
  - directly, if a frame (e.g. Ethernet) has the field that states which protocol it is encapsulating
  - indirectly, with tables of protocol/port combinations and heuristics
    - Usually working, troubles when protocols are used in nonstandard ports

# How to capture network traffic

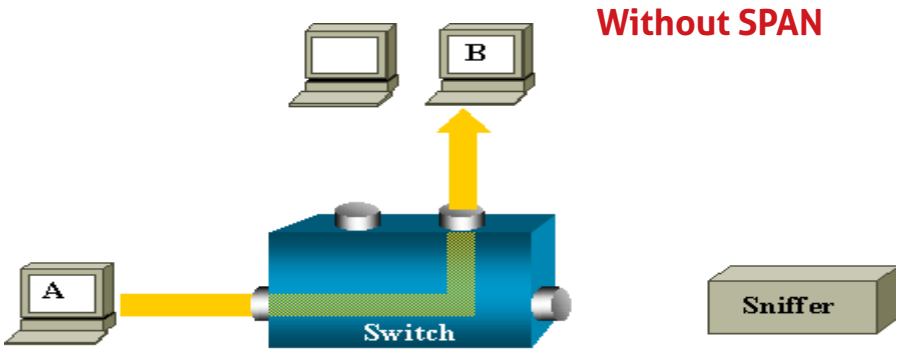
- Promiscuous mode
  - Limitations?
  - Remember the difference between hubs and switches!
- Physical tap
- Port mirroring on a managed switch
- More "aggressive" approaches:
  - ARP cache poisoning
  - MAC flooding
  - DHCP redirection
  - Redirection and interception with ICMP
- NOTICE: on virtualized environments and SDN, this can be easier or harder

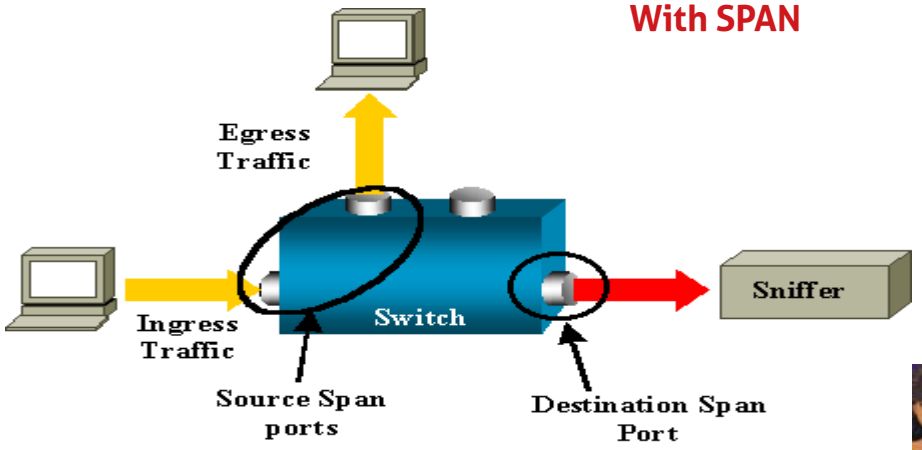




# Port mirroring

- Switched Port Analyzer (SPAN) or Roving Analysis Port (RAP)



# Less conventional approaches for sniffing

- ARP cache poisoning (or spoofing)
  - Unsolicited ARP replies to steal IP addresses (ettercap, cain&abel)
- MAC flooding
  - Fill the CAM of the switch to make it acting as a hub (macof)
- DHCP redirection
  - Rogue DHCP server: it exhausts the IP addresses of the pool
  - Then pretends to be the default gateway of the network with the new DHCP requests (Gobbler, DHCPstarv, Yersinia)
- Redirection and interception with ICMP
  - ICMP type 5 (redirect) used to indicate a better route (ettercap)

# How to prevent packet capture

- Dynamic address inspection
  - Implemented in switches: Dynamic Address Resolution Inspection (DAI) validates ARP packets
  - IP-to-MAC address binding inspection, drop invalid packets
- DHCP snooping
  - Implemented in switches: distinguishes between trusted and untrusted ports and uses a database of IP-to-MAC
  - Ports that show rogue activity can also be automatically placed in a disabled state

# Additional setup

- Configure the GeoIP resolver
  - https://wiki.wireshark.org/HowToUseGeoIP
  - Sign and download the GeoLite2 MaxMind free database(s)
  - Unzip the files in a directory
- In wireshark:
  - Edit→Preferences→Name Resolution
  - Select MaxMind database directories
- Now you can use filters like

ip.geoip.country eq “China”

# Activity 1: pnd-labs/lab1/ex4

- Download the package https://github.com/vitome/pnd-labs.git
- Run tcpdump and save the captured traffic in an output file
  - Best option: into the `/hosthome/` directory
- Use the browser for connecting to the webserver in pnd-labs/lab1/ex4/pc1
  - Browse page **`ba.php`**

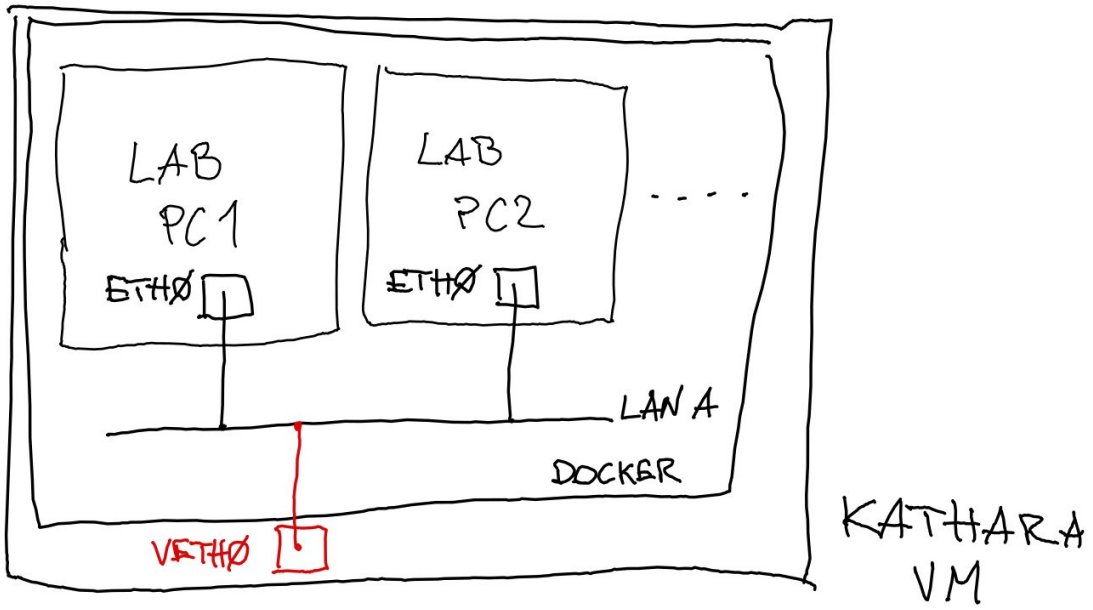
user=angelo psw=angsp

- Stop tcpdump
- Repeat the procedure with another outfile
  - Browse page **`da.php`**

user=angelo psw=angsp

- Use wireshark to analyze and compare the captured files

# Connect the host with the internal lab



# The script pnd-labs/connect-lab.sh

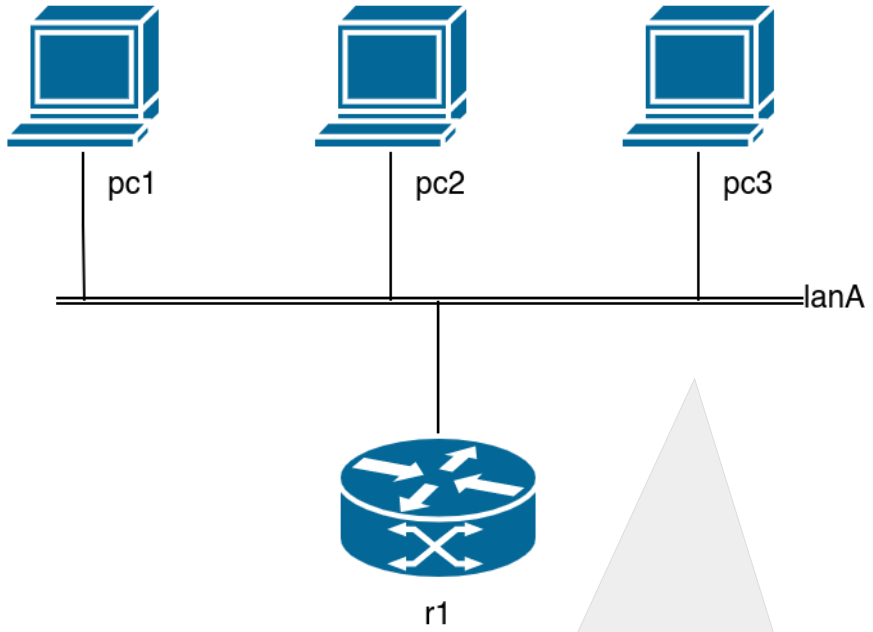
- We can add an interface to connect the internal lans
  - Useful for **interacting** with the lab machines and **capturing traffic**
- Usage: `./connect-lab.sh <IP>/<mask> [<lan>]`
- What it does:
  - adds a virtual interface (pair veth0@veth1):
    - `ip link add dev veth0 type veth peer name veth1`
  - connects one veth end to the virtual bridge:
    - `ip link set master br0 dev veth1`
  - assigns an IP address to the other end (not enslaved):
    - `ip addr add x.x.x.x/y dev veth0`
  - enables both the ends of the virtual interface
    - `ip link set veth0 up`
    - `ip link set veth1 up`

# Activity 2

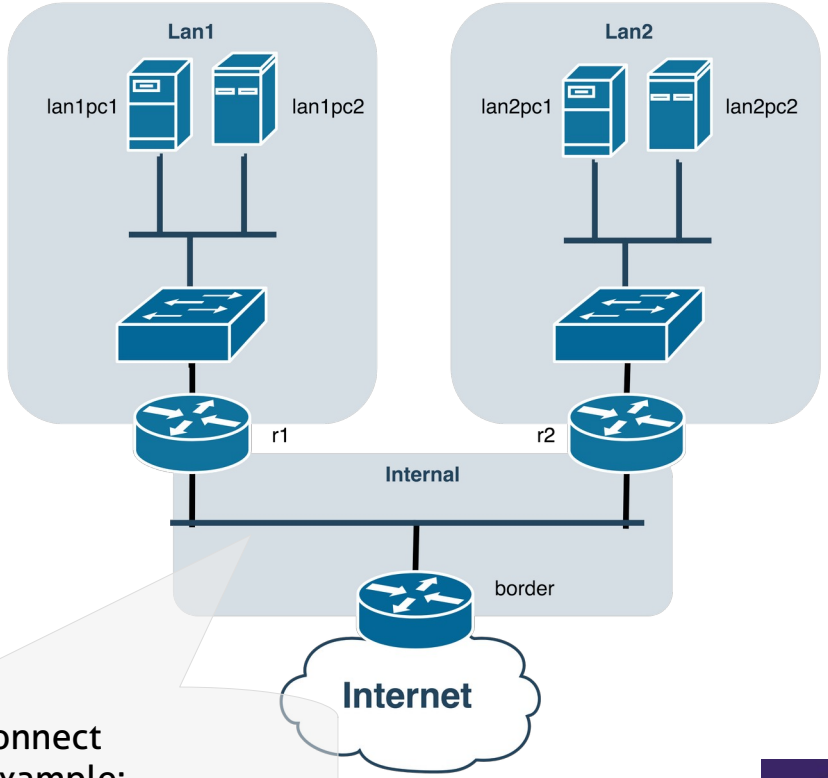
- Run tcpdump and save the captured traffic in an output file
- Connect via ftp to an open ftp server
  - e.g.: test.rebex.net (demo:password)
- Stop tcpdump
- Repeat the procedure with another outfile, connecting with sftp to:
  - test.rebex.net (demo:password)
- Use wireshark to analyze and compare the captured files
- Use wireshark FILTERS of ftp to look for user/password

# Activity 3: pnd-labs/lab1/ex2

- Capture the DHCP exchange of pnd-labs/lab1/ex2
- Run tcpdump and save the captured traffic in an output file
- Then use wireshark from the host machine to explore the captured traffic



# Activity 4




use connect-lab.sh to connect
the host to the lans, for example:

```
./connect-lab.sh 192.168.100.29 internal
```

# Activity 4: pnd-labs/lab1/ex3

- Capture the traffic exchange of pnd-labs/lab1/ex3 between the hosts of the two different lans from different positions
  - lan1, lan2 and internal (between r1 and r2)
- Use tcpdump to save the captured traffic in an output files into the /hosthome/ directory
- Then use wireshark from the host machine to explore the captured traffic
- Pay attention to the layering approach and how packets change when moving from one network to the other

# Activity 5

- Try to solve with wireshark the CTF of Hack3rCon 3 conference (2012)
- http://sickbits.net/other/hc3.pcap-04.cap
  - https://drive.google.com/file/d/1ANd0t_U7Ya8R1fppcHhi51WYq9FjltM6/view?usp=drive_web&authuser=0

# References

- Wireshark for Security Professionals: Using Wireshark and the Metasploit Framework
  - Bullok, Parker, Wiley ed.
- The Network Security Test Lab: A Step-by-Step Guide
  - Gregg, Wiley e.

# That’s all for today

- **Questions?**
- See you next lecture!
- References:
  - http://www.tcpipguide.com/free/t_IPDatagramGeneralFormat.htm
  - https://www.wiley.com/en-us/Wireshark+for+Security+Professionals%3A+Using+Wireshark+and+the+Metasploit+Framework-p-9781118918210