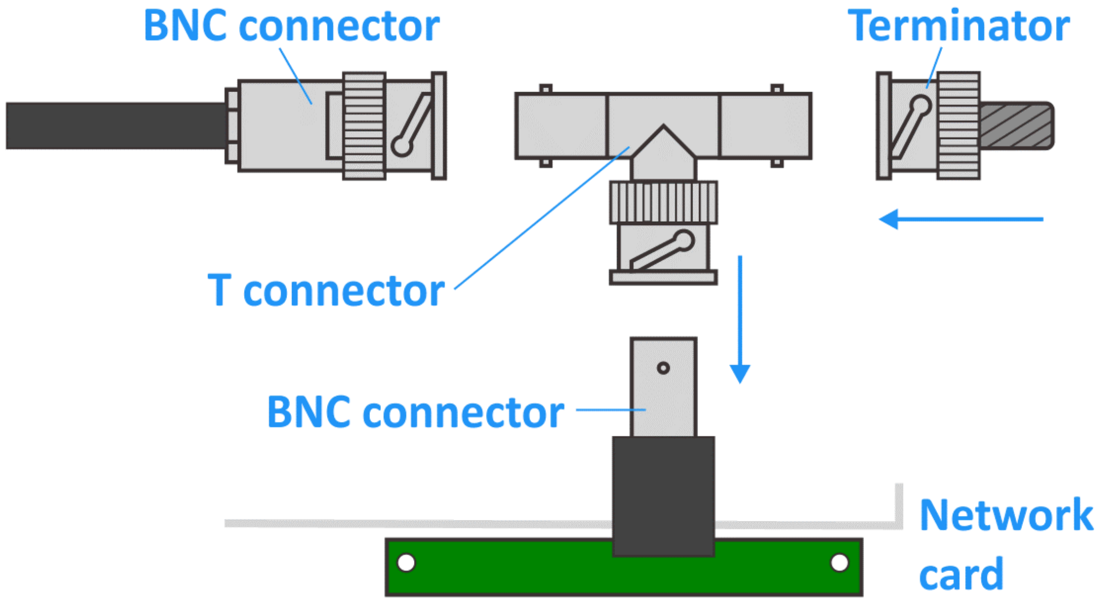
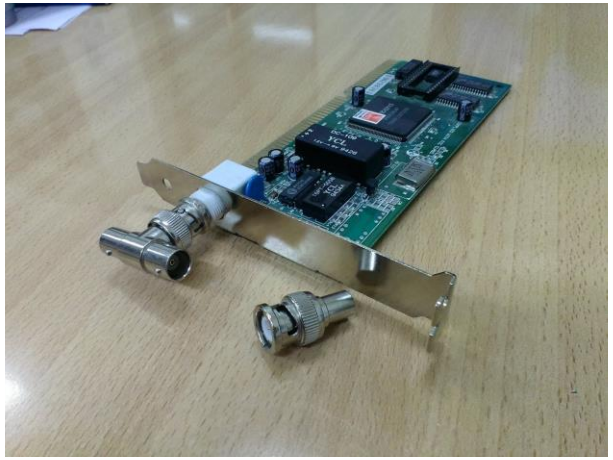
Packet sniffing lab

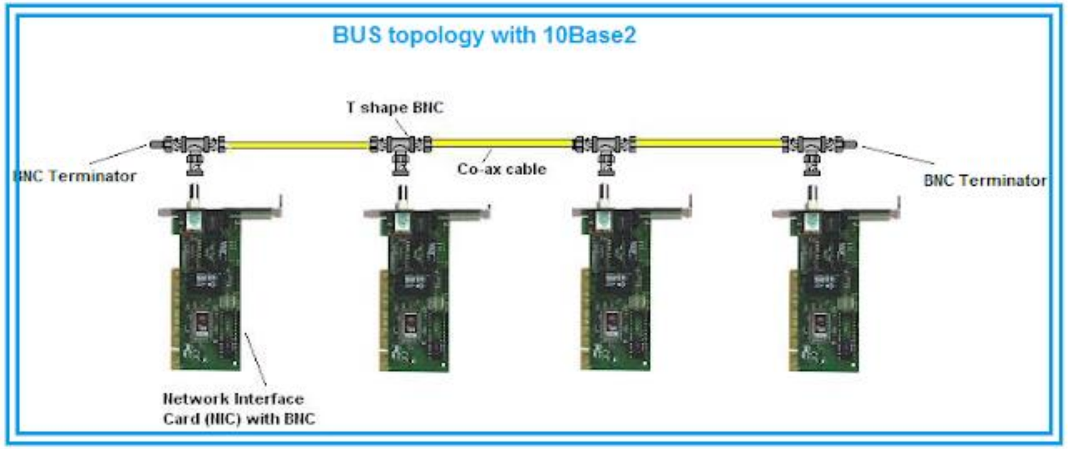
CalPolyPomona

LAN with T connectors (e.g., Eth 10Base2) – old times

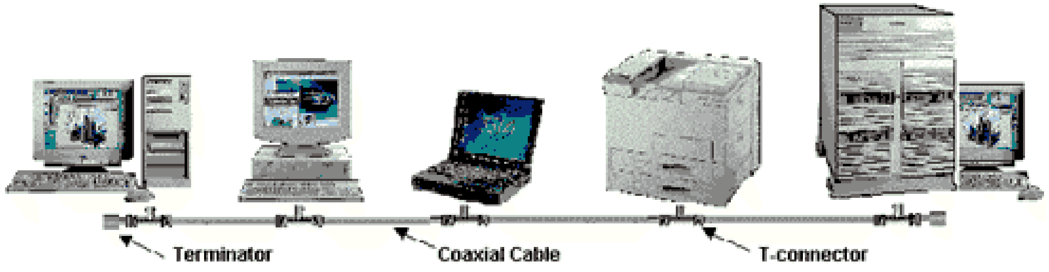
The parts of the bus topology





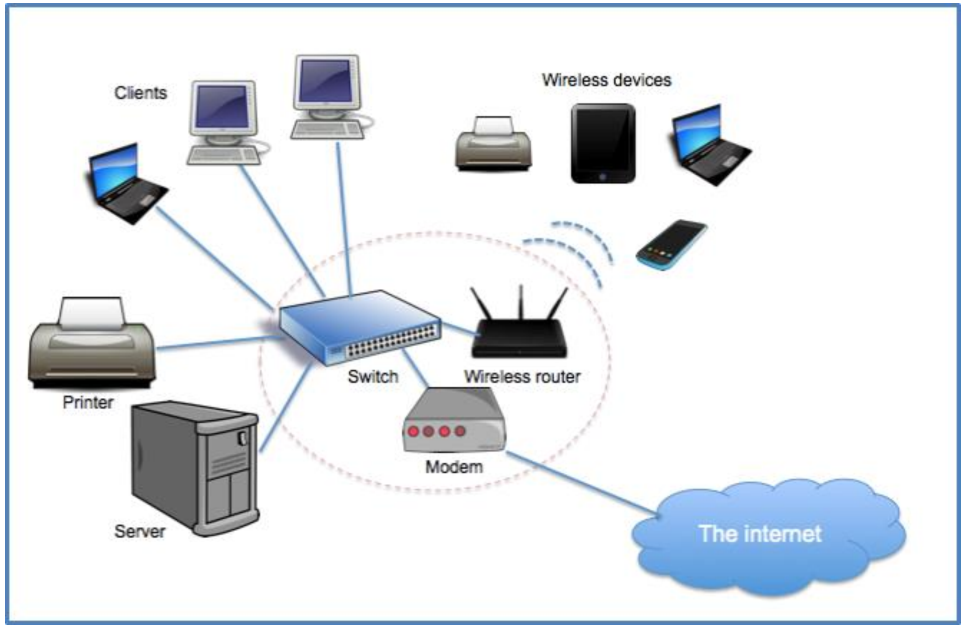

BUS topology with 10Base2
T shape BNC
BNC Terminator
Co-ax cable
BNC Terminator
Network Interface Card (NIC) with BNC

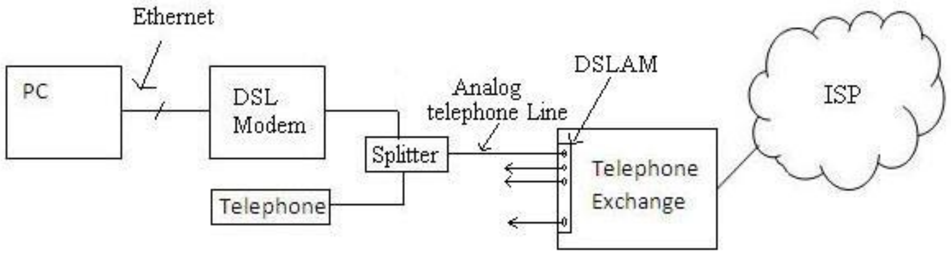
Each signal is transmitted and reaches all machines physically



LAN with RJ45 or air (Eth 802.3 and 802.11abg)

RJ45





Device that connect nodes part of the same LAN

NIC: network interface cards

Access Point (AP)

hub

switch

Router

Wireless router = AP + Router

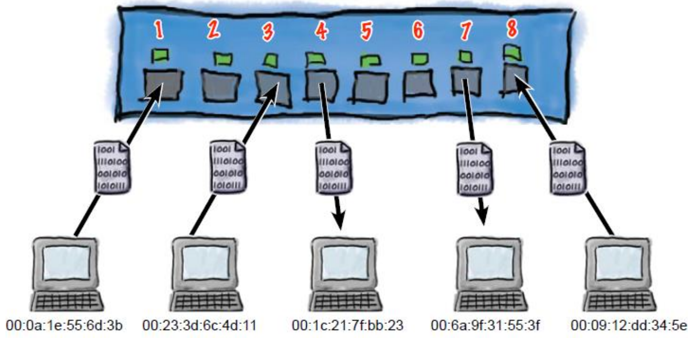
Routers: Connect machines on different Network domains (e.g., LANs)

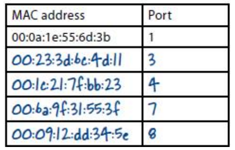
modem

Send and receive an electric signal MoDem = Modulator/ Demodulator on a cables (e.g., to reach the Internet Service Provider station from your office/home LAN)

LAN connectors RJ45 and Ethernet connection

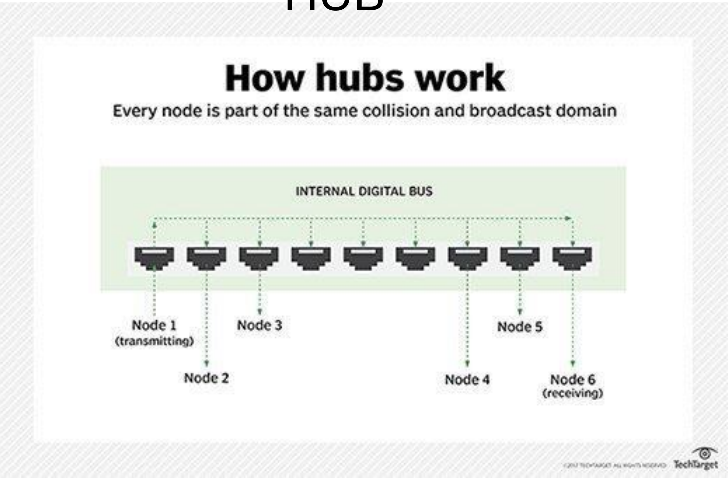
SWITCH





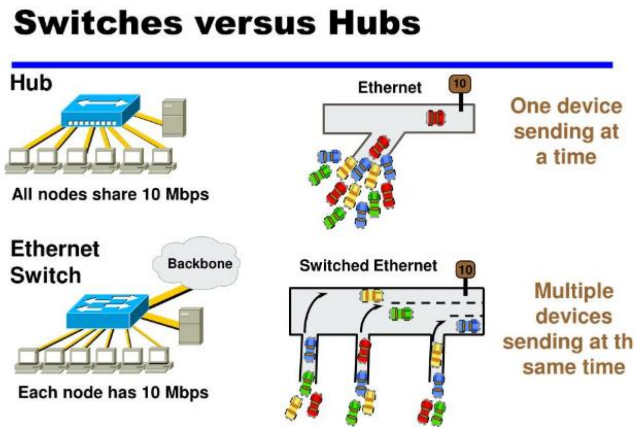
MAC address	Port
00:0a:1e:55:6d:3b	1
00:23:3d:6c:4d:11	3
00:1c:21:7f:bb:23	4
00:6a:9f:31:55:3f	7
00:09:12:dd:34:5e	8

HUB

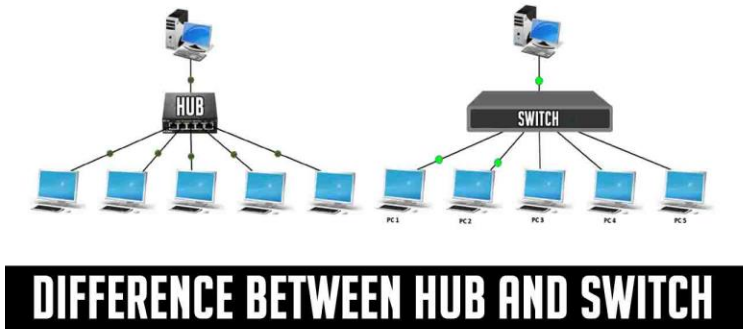


How hubs work

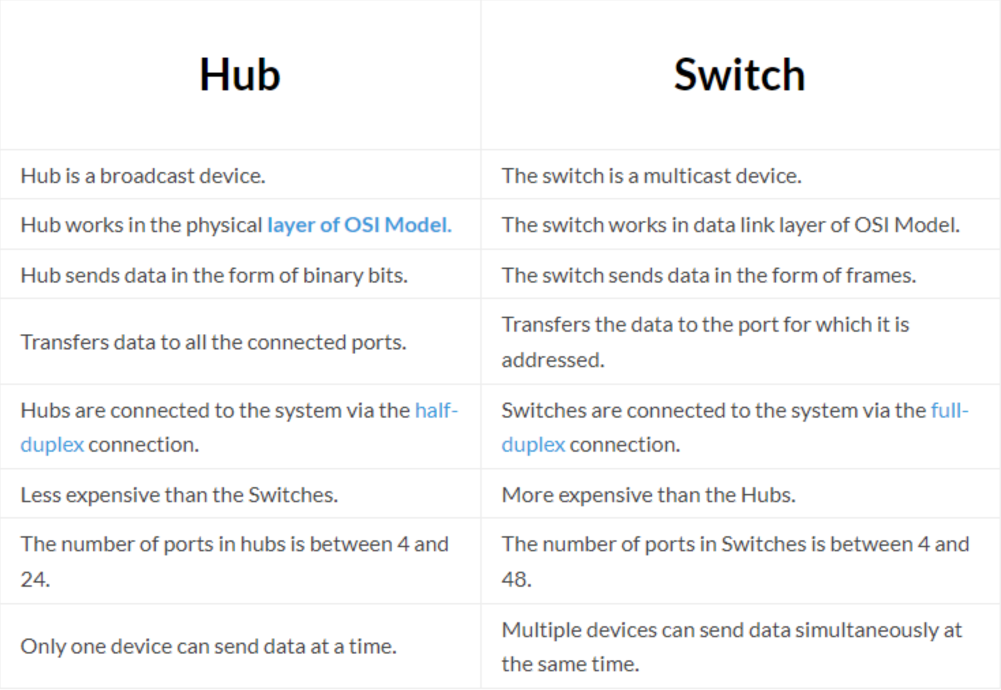
Every node is part of the same collision and broadcast domain



Switches versus Hubs

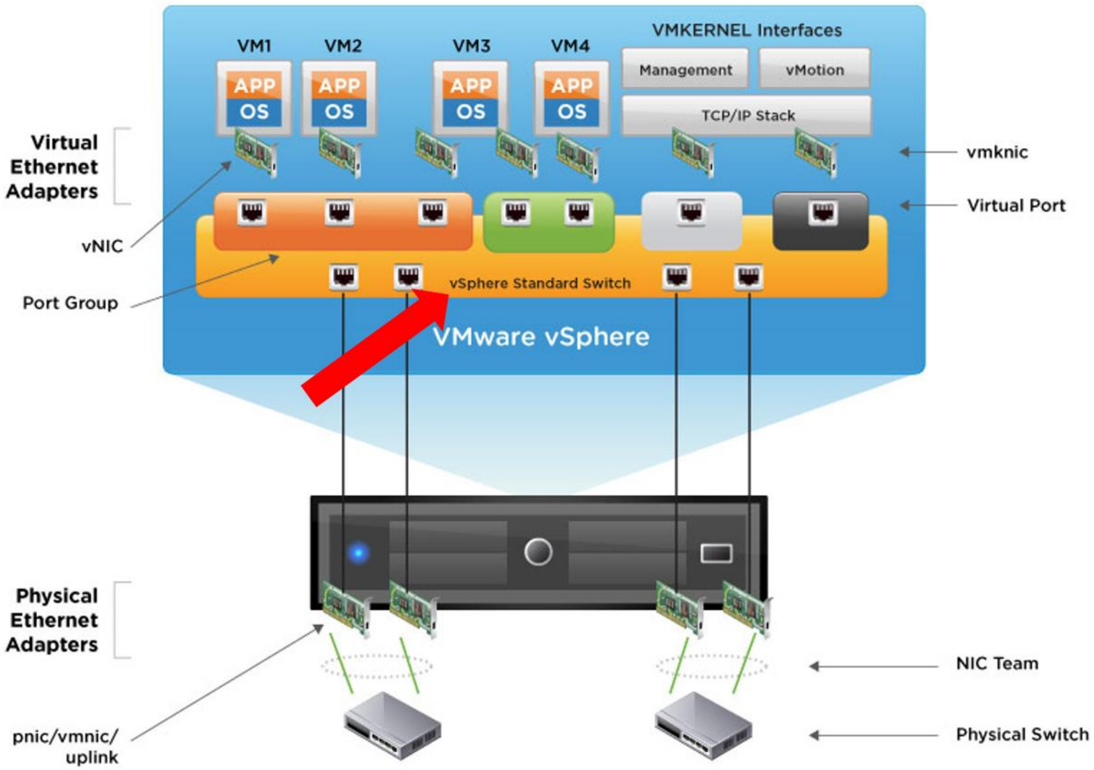


DIFFERENCE BETWEEN HUB AND SWITCH



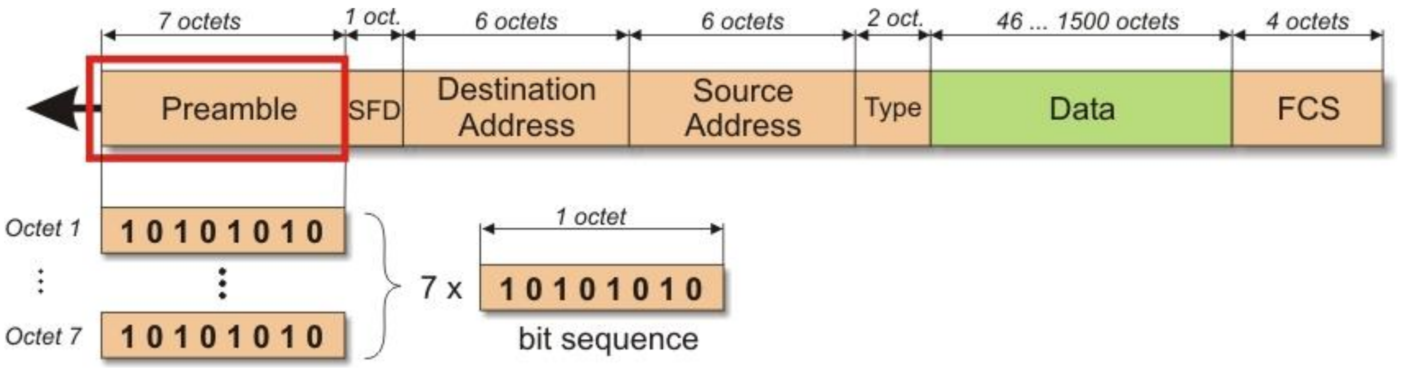
Hub	Switch
Hub is a broadcast device.	The switch is a multicast device.
Hub works in the physical layer of OSI Model.	The switch works in data link layer of OSI Model.
Hub sends data in the form of binary bits.	The switch sends data in the form of frames.
Transfers data to all the connected ports.	Transfers the data to the port for which it is addressed.
Hubs are connected to the system via the half-duplex connection.	Switches are connected to the system via the full-duplex connection.
Less expensive than the Switches.	More expensive than the Hubs.
The number of ports in hubs is between 4 and 24.	The number of ports in Switches is between 4 and 48.
Only one device can send data at a time.	Multiple devices can send data simultaneously at the same time.

Virtual LANs with virtualization (VMware)



- Works in a similar way of physical devices but the software “recreates” emulated network components as seen before (Ethernet cards, Switches and routers)

Ethernet Frame



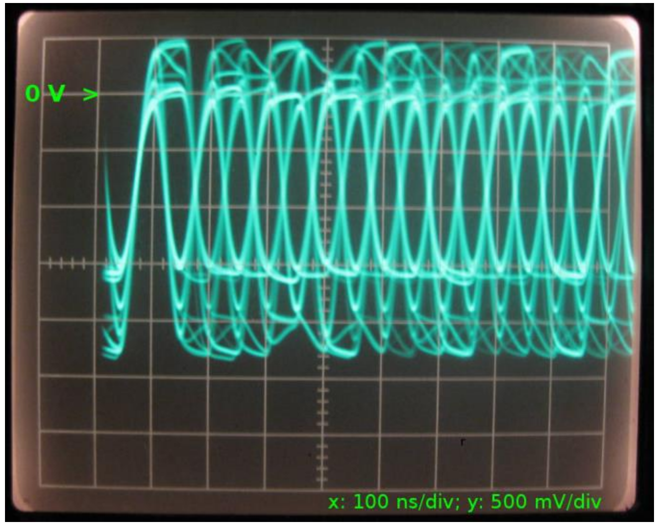
7 octets	1 oct.	6 octets	6 octets	2 oct.	46 ... 1500 octets	4 octets
Preamble	SFD	Destination Address	Source Address	Type	Data	FCS

Octet 1 1 0 1 0 1 0 1 0

⋮

Octet 7 1 0 1 0 1 0 1 0

7 x 1 octet 1 0 1 0 1 0 1 0 bit sequence



PREAMBLE – 8 BYTES

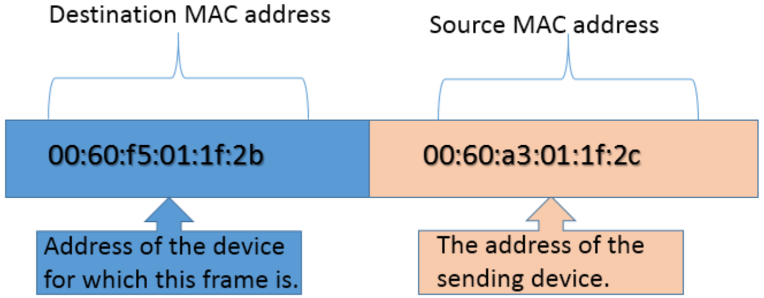
10101010 ... 10101010 (ETHERNET 2)

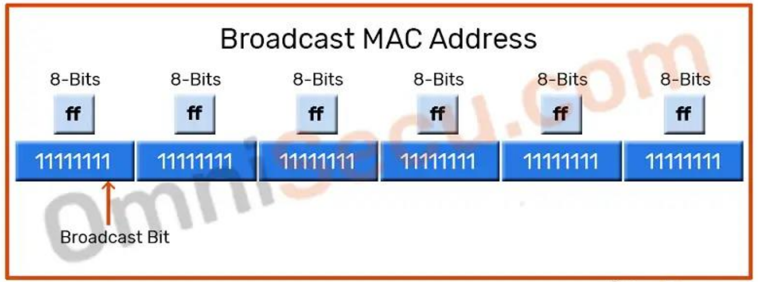
10101010 ... 10101011 (802.3)

Preamble changes from 802.3 (wired Ethernet) to WiFi 802.11abg, but the purpose is the same: each network card catches the preamble signal to understand when the ethernet frame starts

Destination and source address

- Once the frame starts, the first thing arriving to the network card is the destination (physical) address, also known as **destination MAC address**


Destination MAC address
Source MAC address
00:60:f5:01:1f:2b
00:60:a3:01:1f:2c
Address of the device for which this frame is.
The address of the sending device.



©OmniSecu.com

Default mode of operation for an Ethernet card ***(Unicast mode)***

```
If (destination_MAC_address == my_MAC_address)
            receive_all_frame
else
            skip_frame
```

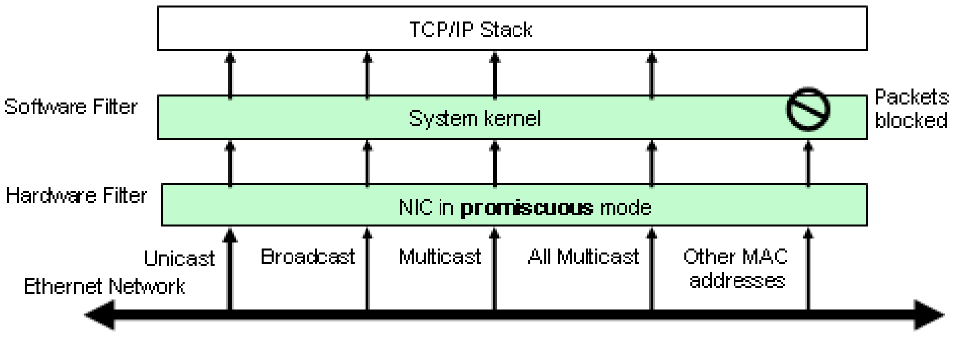
Network Sniffing

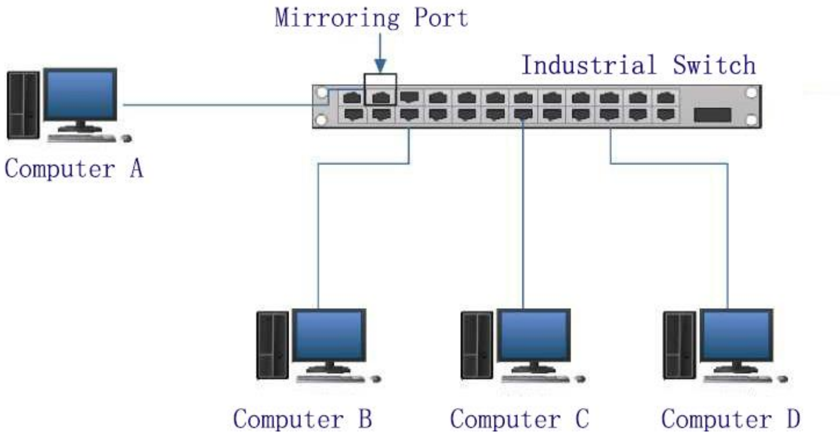
- Adversaries may sniff network traffic to capture information about an environment, including authentication material passed over the network. Network sniffing refers to using the network interface on a system to monitor or capture information sent over a wired or wireless connection. An adversary may place a network interface into promiscuous mode to passively access data in transit over the network, or use span ports to capture a larger amount of data.
- Data captured via this technique may include user credentials, especially those sent over an insecure, unencrypted protocol. Techniques for name service resolution poisoning, such as LLMNR/NBT-NS Poisoning and SMB Relay, can also be used to capture credentials to websites, proxies, and internal systems by redirecting traffic to an adversary.
- Network sniffing may also reveal configuration details, such as running services, version numbers, and other network characteristics (e.g. IP addresses, hostnames, VLAN IDs) necessary for subsequent Lateral Movement and/or Defense Evasion activities.
- In cloud-based environments, adversaries may still be able to use traffic mirroring services to sniff network traffic from virtual machines. For example, AWS Traffic Mirroring, GCP Packet Mirroring, and Azure vTap allow users to define specified instances to collect traffic from and specified targets to send collected traffic to.[1][2][3] Often, much of this traffic will be in cleartext due to the use of TLS termination at the load balancer level to reduce the strain of encrypting and decrypting traffic.[4][5] The adversary can then use exfiltration techniques such as Transfer Data to Cloud Account in order to access the sniffed traffic.[4]

https://attack.mitre.org/techniques/T1040/

How do we capture all the traffic on a LAN?

1) Set your network card in “Promiscuous mode”
   - It makes sure your NIC doesn’t skip frames not intended for the interface MAC address, .e., observe anything “on the wire”
   - Note: wireshark should set it for you, if you use that as a sniffing tool

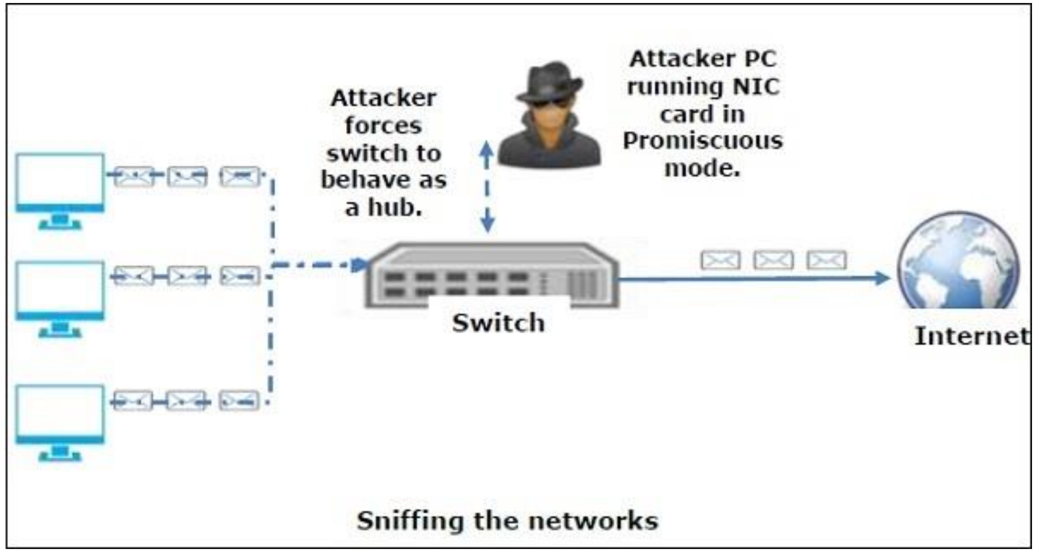


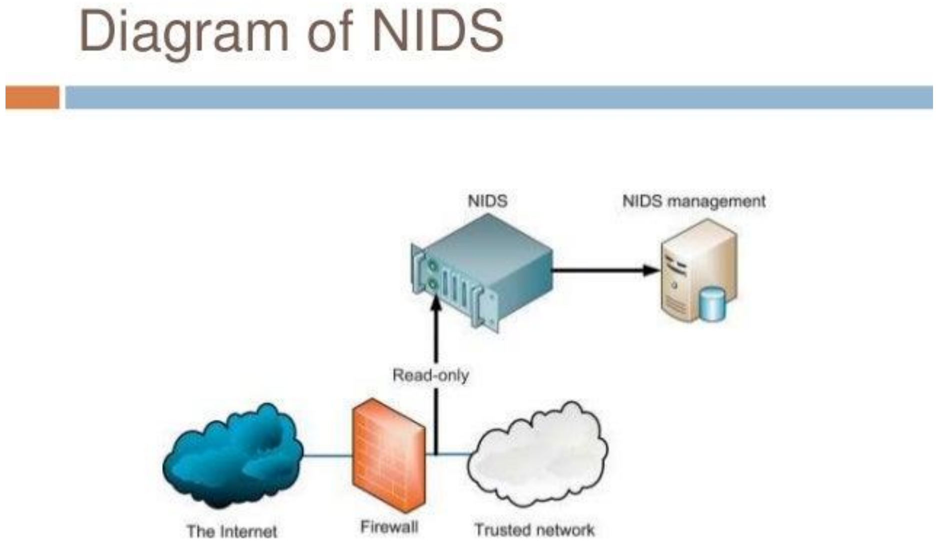


2) Connect your computer to a **HUB** or to the **mirror port** of a network switch
   - Network switches implement port segmentation which doesn’t allow your machine to receive all network traffic, i.e., packets not for your MAC address; that’s why they have a **mirror port**
   - Switch vendors tell you which port is for mirroring, or you can set it from the switch configuration interface

Network Sniffing: malicious vs good intent

Sniffing is a security solution for data-in-transit


Attacker forces switch to behave as a hub.
Attacker PC running NIC card in Promiscuous mode.
Switch
Internet
Sniffing the networks



Network-based Intrusion Detection System
N-IDS

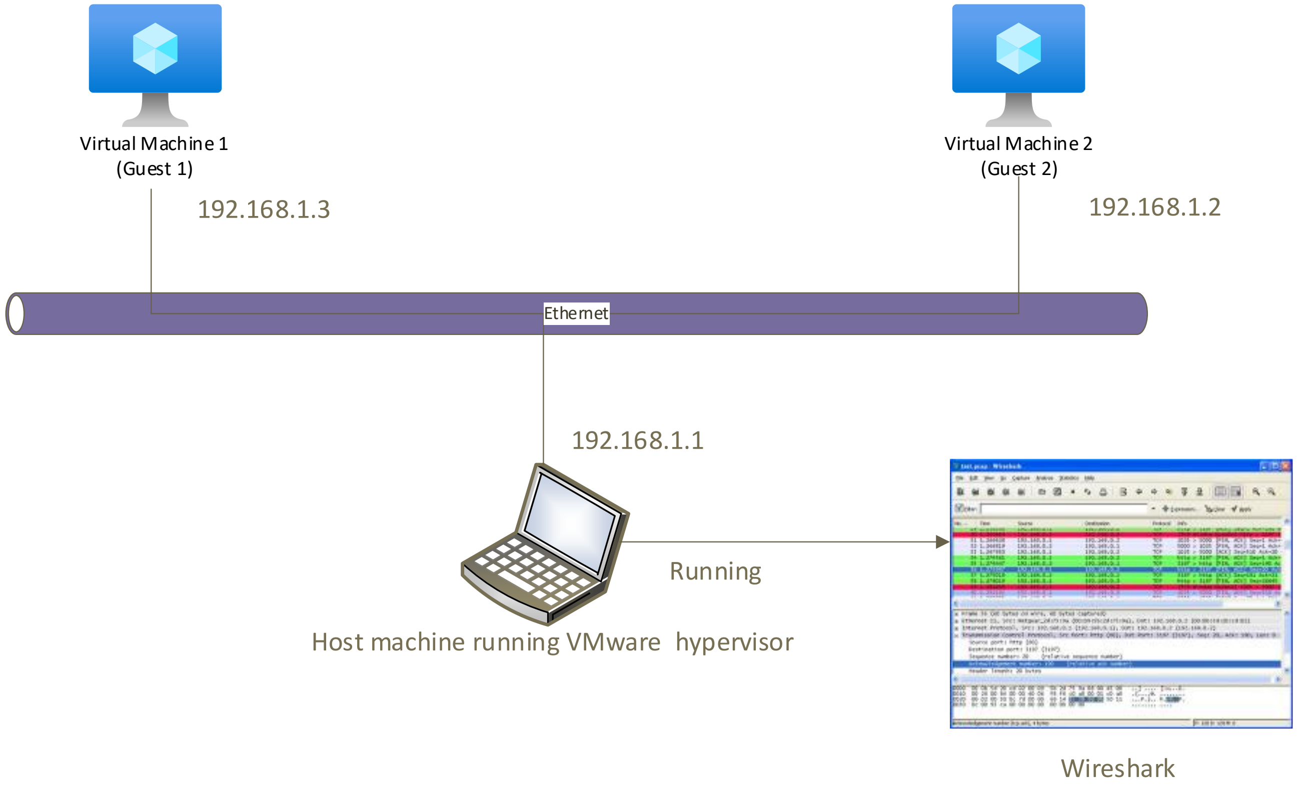
Sniffing based attack, ways to obtain (alternatives):

1. The attacker physically connect to a hub
2. The attacker connects to the managing port of the switch and removes the segmentation features, transforming it into a hub
3. The attacker access a host which is able to see all network traffic (e.g., a Network Intrusion Detection System, right picture here)

LAN sniffing in VMware

Set up

- Live…
- Note: IP addresses might be different



Virtual Machine 1 (Guest 1)

192.168.1.3

Virtual Machine 2 (Guest 2)

192.168.1.2

Ethernet

192.168.1.1

Running

Host machine running VMware hypervisor

Wireshark

Example of protocol stack

FIGURE 1: INTERNET PROTOCOL STACK

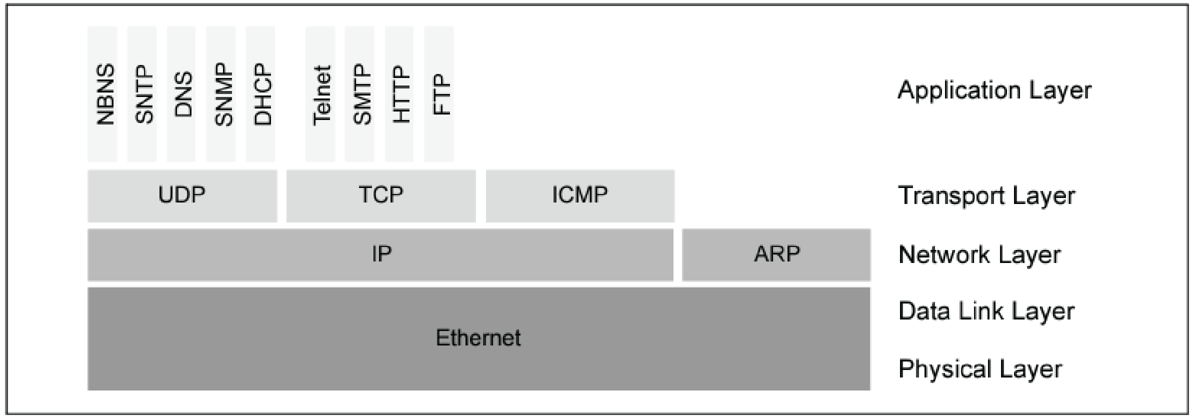
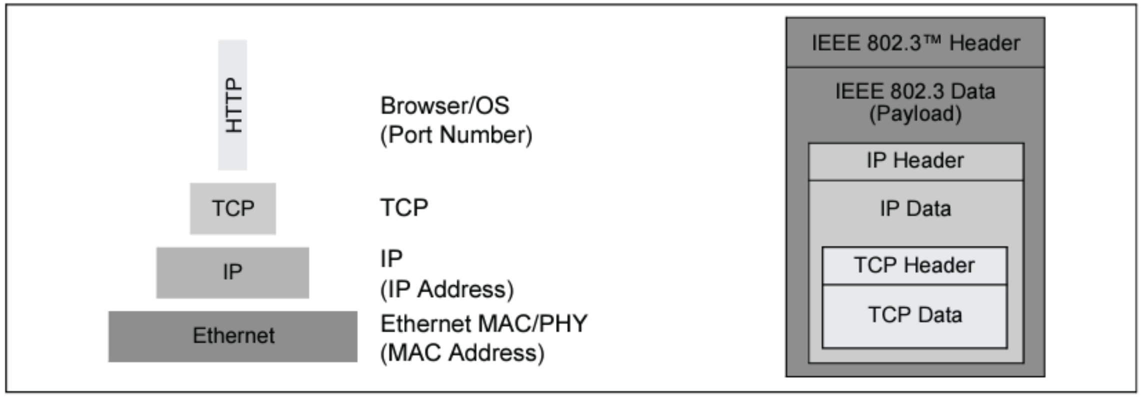


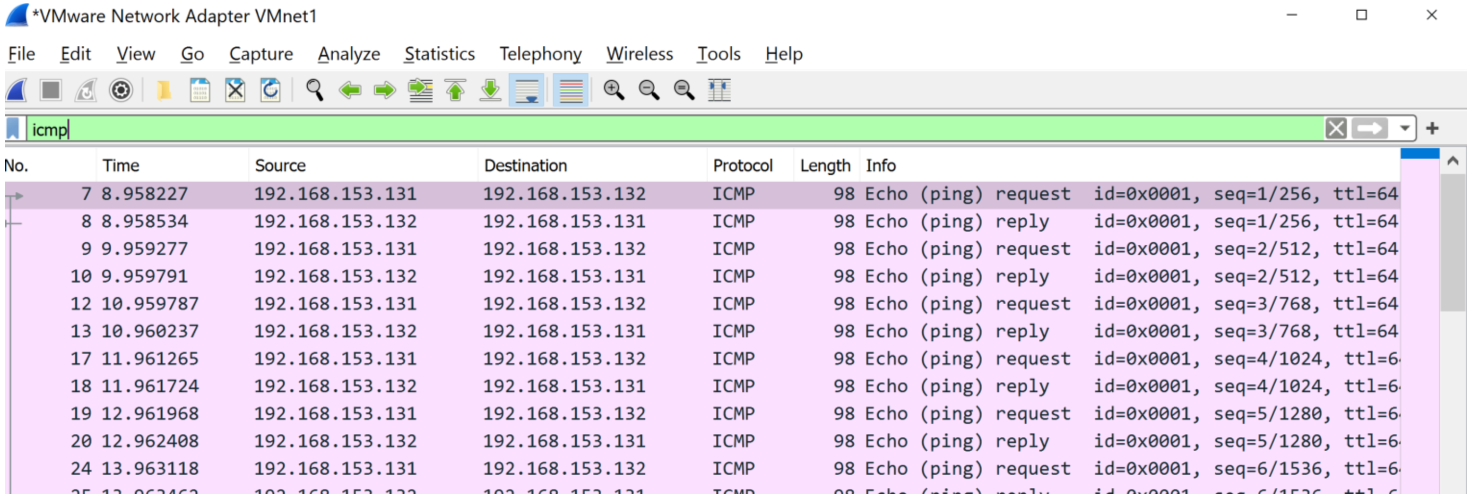
FIGURE 2: DATA ENCAPSULATION EXAMPLE



Exercise 1 – sniff a ping pong

1. Run Wireshark on Host machine
2. Ping VM1 with VM2
3. Observe Ping between VM1 and VM2
    - ICMP Echo Request and Reply
    - Apply display filter: icmp
4. Inspect fields

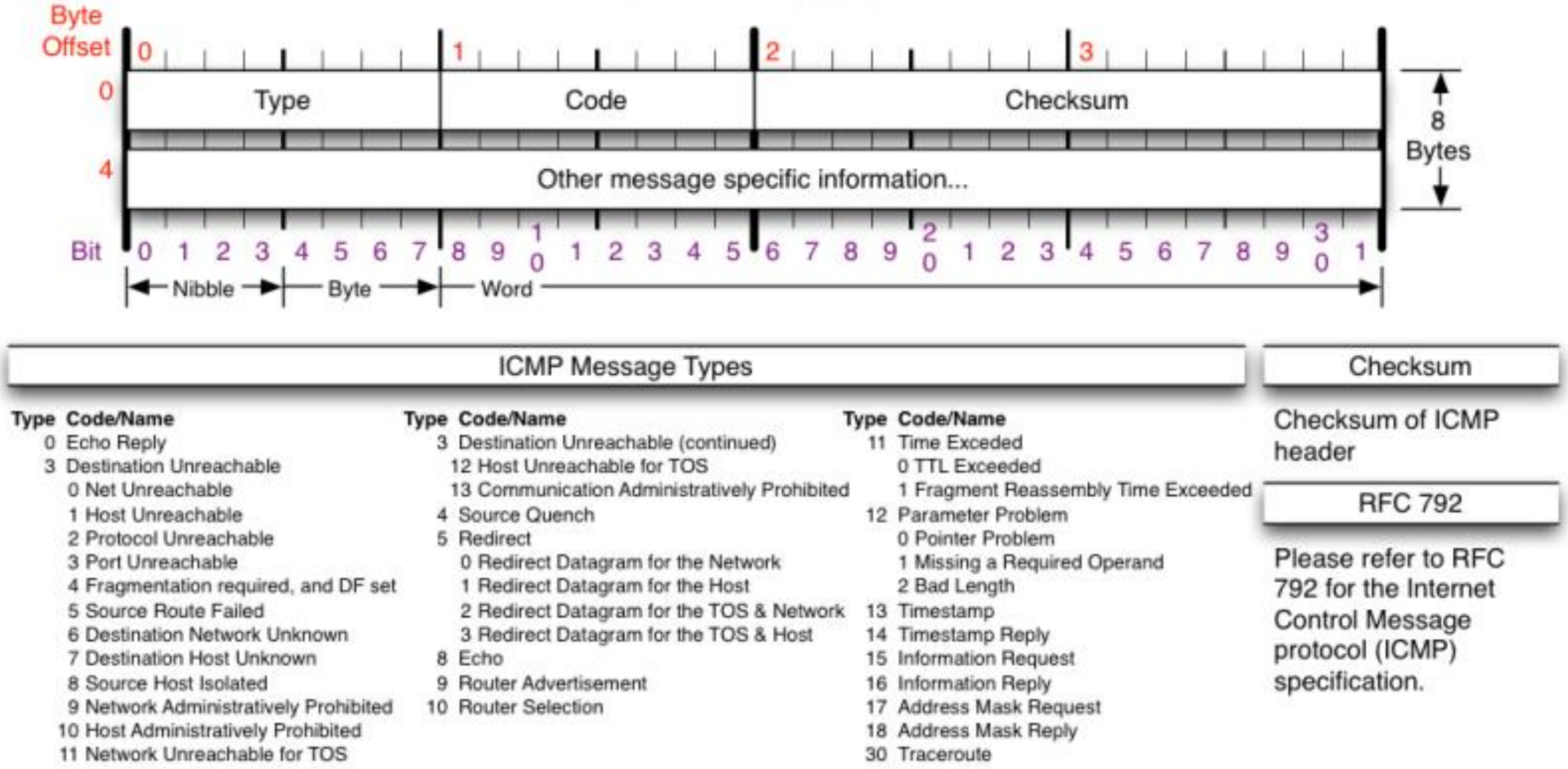
https://wiki.wireshark.org/Internet_Control_Message_Protocol



icmp

No.	Time	Source	Destination	Protocol	Length	Info
7	8.958227	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=1/256, ttl=64
8	8.958534	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256, ttl=64
9	9.959277	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=2/512, ttl=64
10	9.959791	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=2/512, ttl=64
12	10.959787	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=3/768, ttl=64
13	10.960237	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=3/768, ttl=64
17	11.961265	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=4/1024, ttl=6
18	11.961724	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=4/1024, ttl=6
19	12.961968	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=6
20	12.962408	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=5/1280, ttl=6
24	13.963118	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=6/1536, ttl=6
25	13.963462	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=6/1536, ttl=6

ICMP (ping, pong and others)



ICMP Message Types

Type	Code/Name
0	Echo Reply
3	Destination Unreachable
	0 Net Unreachable
	1 Host Unreachable
	2 Protocol Unreachable
	3 Port Unreachable
	4 Fragmentation required, and DF set
	5 Source Route Failed
	6 Destination Network Unknown
	7 Destination Host Unknown
	8 Source Host Isolated
	9 Network Administratively Prohibited
	10 Host Administratively Prohibited
	11 Network Unreachable for TOS
3	Destination Unreachable (continued)
	12 Host Unreachable for TOS
	13 Communication Administratively Prohibited
4	Source Quench
5	Redirect
	0 Redirect Datagram for the Network
	1 Redirect Datagram for the Host
	2 Redirect Datagram for the TOS & Network
	3 Redirect Datagram for the TOS & Host
8	Echo
9	Router Advertisement
10	Router Selection
11	Time Exceded
	0 TTL Exceeded
	1 Fragment Reassembly Time Exceeded
12	Parameter Problem
	0 Pointer Problem
	1 Missing a Required Operand
	2 Bad Length
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply
17	Address Mask Request
18	Address Mask Reply
30	Traceroute

Checksum

Checksum of ICMP header

RFC 792

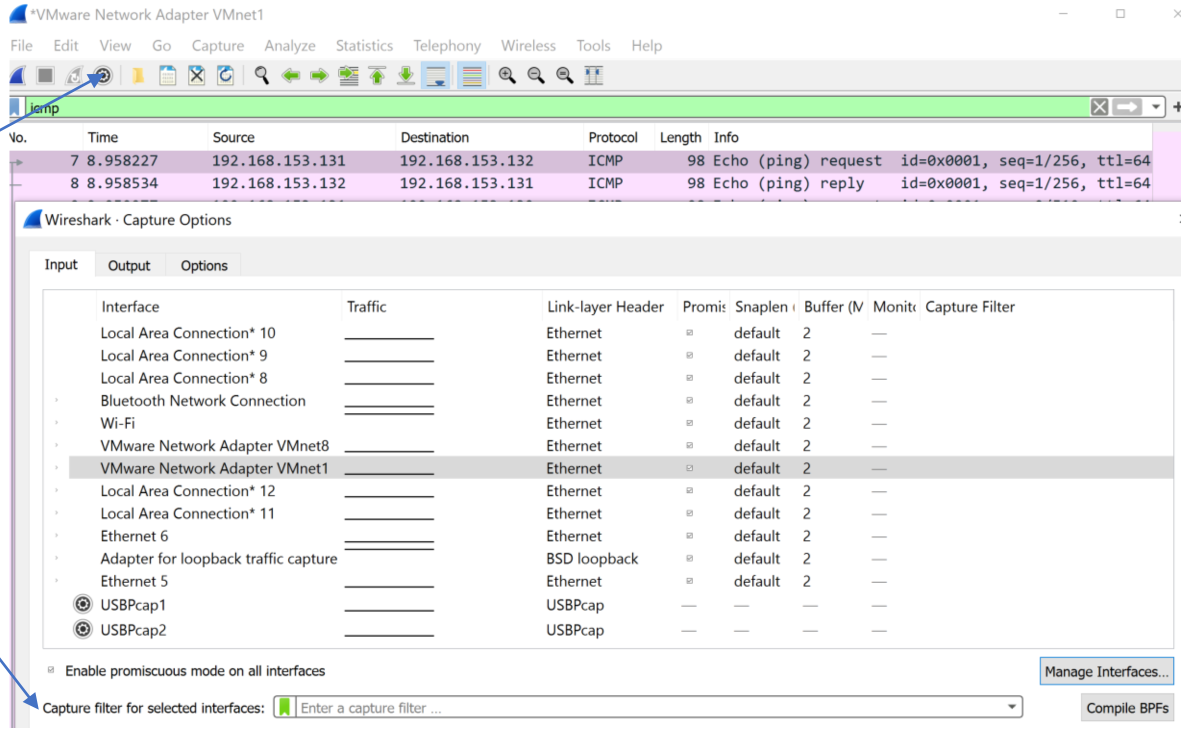
Please refer to RFC 792 for the Internet Control Message protocol (ICMP) specification.

Observation: Capture filter Vs Display filter

- **Capture filter** is a filter that drops data not matching the filter rule
- **Display filter** is a filter that simply shows some packets matching the filter in the visual interface. Other packets might still be captured

Display filter set up

Capture filter set up


*VMware Network Adapter VMnet1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Wireshark · Capture Options
Input Output Options
Manage Interfaces...
Compile BPFs

icmp

No.	Time	Source	Destination	Protocol	Length	Info
7	8.958227	192.168.153.131	192.168.153.132	ICMP	98	Echo (ping) request id=0x0001, seq=1/256, ttl=64
8	8.958534	192.168.153.132	192.168.153.131	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256, ttl=64

Interface	Traffic	Link-layer Header	Promis	Snaplen	Buffer (M	Monit	Capture Filter
Local Area Connection* 10		Ethernet	☑	default	2	—	
Local Area Connection* 9		Ethernet	☑	default	2	—	
Local Area Connection* 8		Ethernet	☑	default	2	—	
Bluetooth Network Connection		Ethernet	☑	default	2	—	
Wi-Fi		Ethernet	☑	default	2	—	
VMware Network Adapter VMnet8		Ethernet	☑	default	2	—	
VMware Network Adapter VMnet1		Ethernet	☑	default	2	—	
Local Area Connection* 12		Ethernet	☑	default	2	—	
Local Area Connection* 11		Ethernet	☑	default	2	—	
Ethernet 6		Ethernet	☑	default	2	—	
Adapter for loopback traffic capture		BSD loopback	☑	default	2	—	
Ethernet 5		Ethernet	☑	default	2	—	
USBPcap1		USBPcap	—	—	—	—	
USBPcap2		USBPcap	—	—	—	—	

☑ Enable promiscuous mode on all interfaces

Capture filter for selected interfaces: Enter a capture filter ...

Exercise 2 – sniff a netcat

1. Establish a kept-alive client-server communication between VM1 and VM2 using netcat
   1. Server:

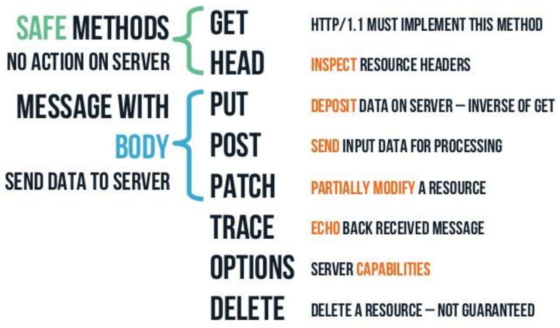
      nc –l –p 2000 -k
   2. Client :

      nc *server_IP_address* 2000
2. Execute a network sniffing with wireshark from host machine
3. Send some messages from Netcat client towards Netcat server
4. Stop the capture
5. Analyze the network caputres:
   - Visualize src and dst IP addresses from the packet capture on the host machine
   - Visualize what is the network transport protocol
   - Visualize if the destination port on the server is corresponding to the destination port on the captured packets (sniffed packets)
   - Visualize the content of your messages sent from client and server in Wireshark

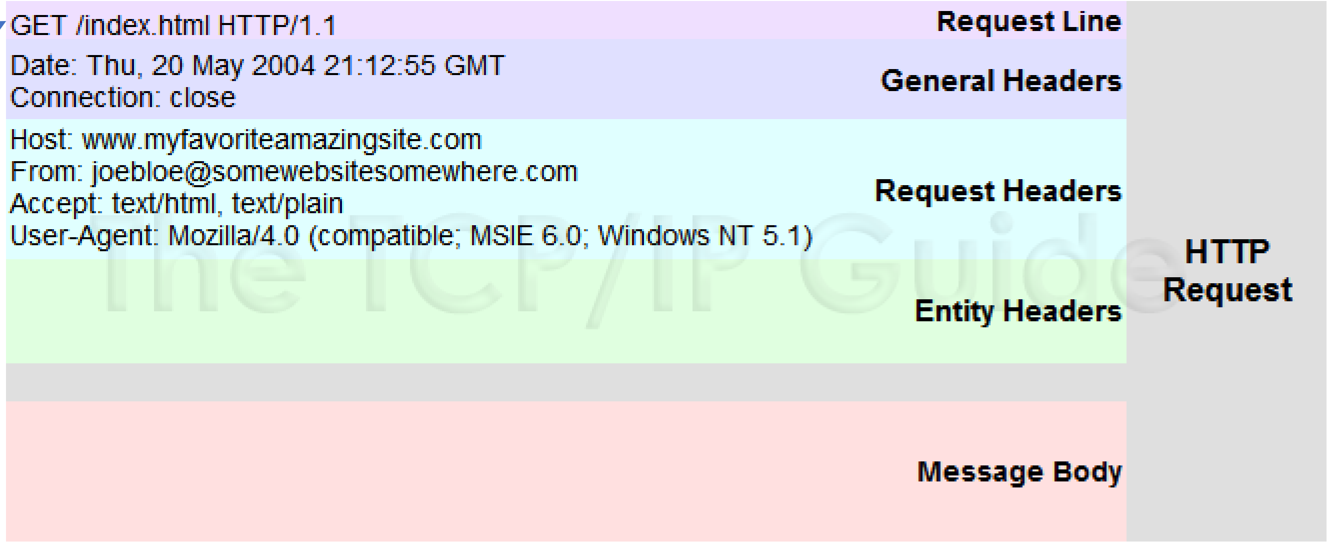
Exercise 3 – sniff an http communication

- Install apache webserver on one of the VMs
  - sudo apt -y install apache2
- Connect from the browser to http://IP_address_web_server
- Change the content of the webserver to a random page:
  - echo '<!doctype html><html><body><h1>Hello World!</h1></body></html>' | sudo tee /var/www/html/index.html
- Execute a network capture from Wireshark and observe the communication pattern
  - Identify the HTTP packet with "Hello World!"

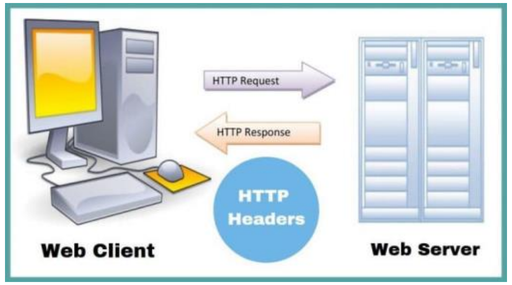
HTTP messages

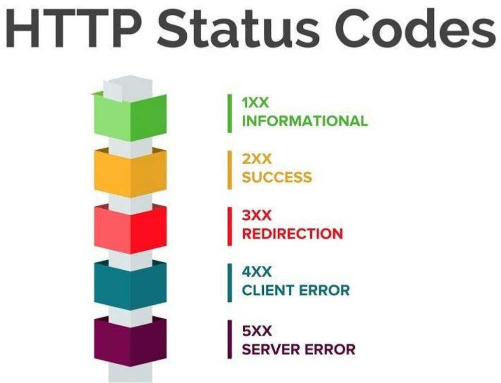


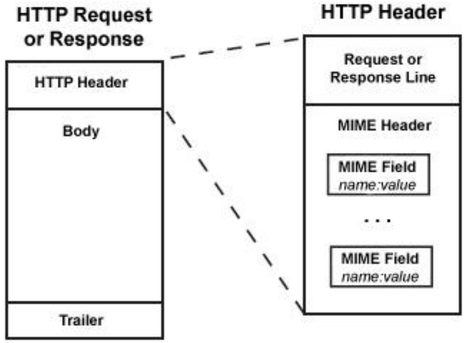
SAFE METHODS NO ACTION ON SERVER	GET	HTTP/1.1 MUST IMPLEMENT THIS METHOD
	HEAD	INSPECT RESOURCE HEADERS
MESSAGE WITH BODY SEND DATA TO SERVER	PUT	DEPOSIT DATA ON SERVER – INVERSE OF GET
	POST	SEND INPUT DATA FOR PROCESSING
	PATCH	PARTIALLY MODIFY A RESOURCE
	TRACE	ECHO BACK RECEIVED MESSAGE
	OPTIONS	SERVER CAPABILITIES
	DELETE	DELETE A RESOURCE – NOT GUARANTEED

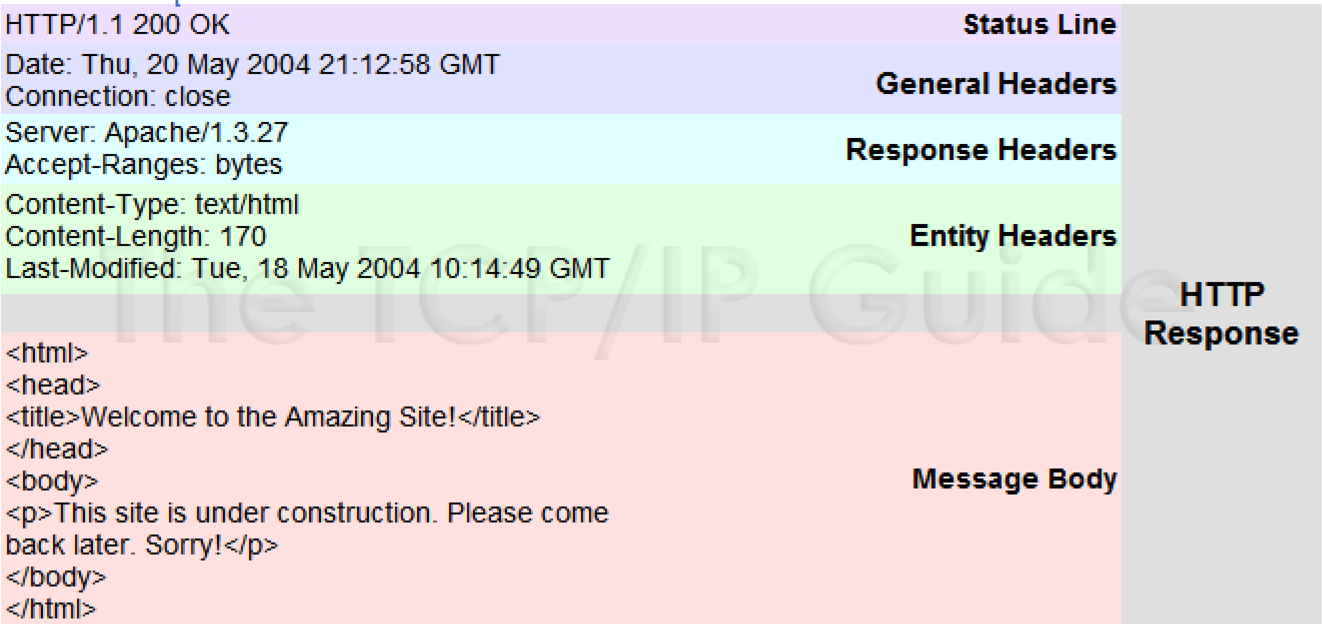


GET /index.html HTTP/1.1	Request Line	HTTP Request
Date: Thu, 20 May 2004 21:12:55 GMT Connection: close	General Headers	
Host: www.myfavoriteamazingsite.com From: joebloe@somewebsitesomewhere.com Accept: text/html, text/plain User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)	Request Headers	
	Entity Headers	
	Message Body	



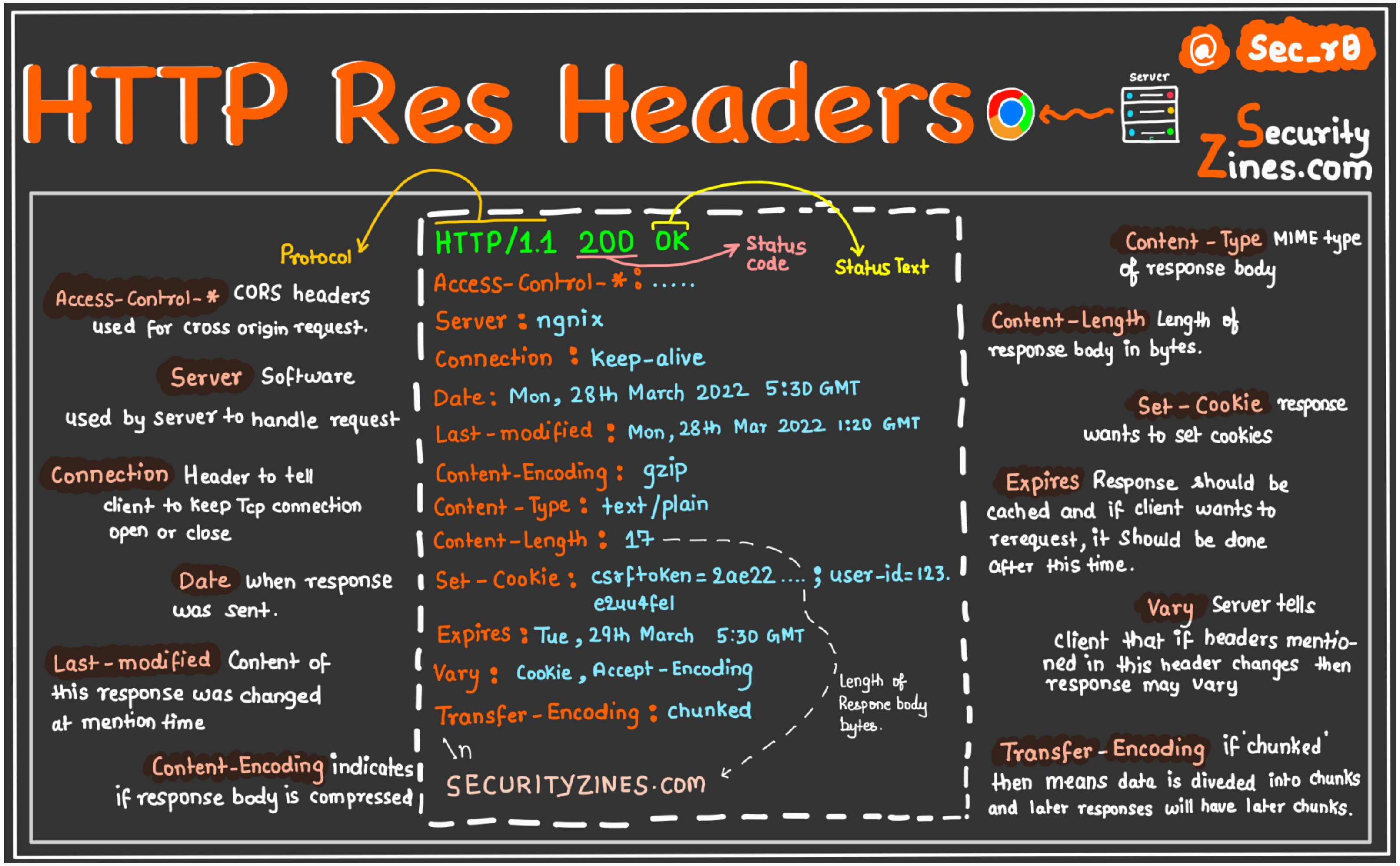

HTTP Status Codes





HTTP/1.1 200 OK	Status Line	HTTP Response
Date: Thu, 20 May 2004 21:12:58 GMT Connection: close	General Headers	
Server: Apache/1.3.27 Accept-Ranges: bytes	Response Headers	
Content-Type: text/html Content-Length: 170 Last-Modified: Tue, 18 May 2004 10:14:49 GMT	Entity Headers	
<html> <head> <title>Welcome to the Amazing Site!</title> </head> <body> <p>This site is under construction. Please come back later. Sorry!</p> </body> </html>	Message Body	

More header fields



@ Sec_r0

HTTP Res Headers

SecurityZines.com

Protocol

Access-Control-* CORS headers used for cross origin request.

Server Software used by server to handle request

Connection Header to tell client to keep Tcp connection open or close

Date when response was sent.

Last-modified Content of this response was changed at mention time

Content-Encoding indicates if response body is compressed

```
HTTP/1.1 200 OK
Access-Control-* : .....
Server : ngnix
Connection : Keep-alive
Date : Mon, 28th March 2022 5:30 GMT
Last-modified : Mon, 28th Mar 2022 1:20 GMT
Content-Encoding : gzip
Content-Type : text/plain
Content-Length : 17
Set-Cookie : csrftoken=2ae22.... ; user-id=123.
             e2uu4fel
Expires : Tue, 29th March 5:30 GMT
Vary : Cookie, Accept-Encoding
Transfer-Encoding : chunked
\n
SECURITYZINES.COM
```

Status code

Status Text

Length of Respone body bytes.

Content-Type MIME type of response body

Content-Length Length of response body in bytes.

Set-Cookie response wants to set cookies

Expires Response should be cached and if client wants to rerequest, it should be done after this time.

Vary Server tells client that if headers mentioned in this header changes then response may vary

Transfer-Encoding if 'chunked' then means data is diveded into chunks and later responses will have later chunks.

https://securityzines.com/flyers/httpres.html

Exercise 4 – sniff using tcpdump

- A command line tool that uses same library of wireshark (libpcap or winpcap)
  - Remember to be sudoer
  - Cheatsheet: https://cdn.comparitech.com/wp-content/uploads/2019/06/tcpdump-cheat-sheet-1.jpg.webp
- Check available interfaces and their names:
  - tcpdump –D
- Command line for sniffing on any interfaces (or specify one) and stop after 5 packets:
  - tcpdump –i any –c5
- disable name resolution by using the option -n and port resolution with –nn:
  - tcpdump –i any –c5 –n –nn
- Filtering packets (e.g., only icmp packets):
  - sudo tcpdump -i any -c5 icmp
- Quite mode (less packet details):
  - sudo tcpdump -i any –q
- Capture http packets and also translate in ASCII format:
  - sudo tcpdump -i any -A port 80
- Save the capture on a file (pcap format) or read it:
  - sudo tcpdump -w capture.pcap
  - sudo tcpdump -r capture.pcap

Exercise 5 – build a sniffer in Python Scapy

- https://scapy.readthedocs.io/en/latest/introduction.html
- Install python3 (if not installed)
- Install scapy module:
  - sudo apt-get install python3-scapy
- First step, create a text file with py extinction
  - touch sniffer.py
- Second, make it executable (for all users, add executable permission on the file):
  - chmod a+x sniffer.py
  - Not necessary but makes life easier to run the script without calling python
- Then, we edit and try it (& will leave the gedit process in background, so you can continue use the current shell to test the python code):
  - gedit sniffer.py &

Exercise 6

- Write a Scapy program that:
  - captures only HTTP data and ICMP packets towards one of your virtual machines
  - saves the pcap files in a local folder where is the sniffer running
  - Bonus: capture and save pcap files from your host machine (e.g., Windows). The capture packets are related only to one of your guest virtual machines
    - For the bonus, you need to have python running on your host