

Compito W1D4 – Pratica

Corso: Cybersecurity Analyst – Epicode

Studente: Daniele Taormina

Data: 25 settembre 2025

Introduzione:

Per questo esercizio ho installato Cisco Packet Tracer sia sul mio Mac che su Kali Linux in macchina virtuale. Ho preferito completare l' esercizio su Kali.

La procedura è stata un po' diversa da quella delle slide a causa di un problema di una libreria mancante, ma smanettando un pò sono riuscito a trovare una soluzione ed infine ad installarlo. (se è necessario posso fare anche una consegna su come ho risolto il problema).

Obiettivo dell'esercizio

Mettere in comunicazione Laptop0, Laptop1 e PC0 con Switch0 ed eseguire un ping in modalità simulazione per osservare il broadcast a livello 2.

Esecuzione:

Usando il comando: **`sudo apt update && sudo apt upgrade -y`**

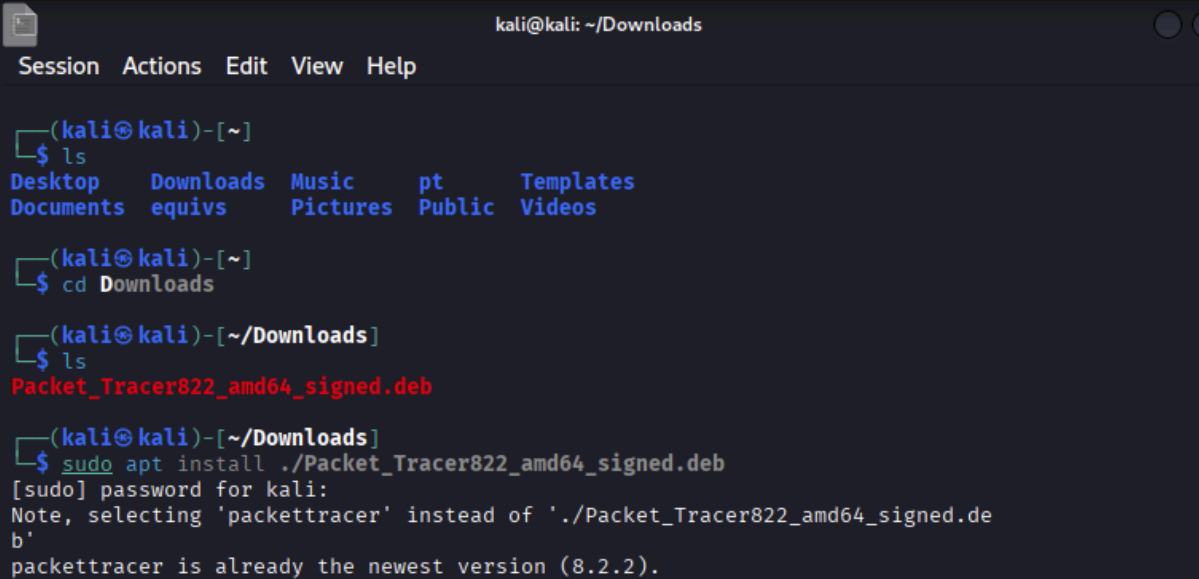
A screenshot of a terminal window titled 'kali@kali: ~'. The window has a menu bar with 'Session', 'Actions', 'Edit', 'View', and 'Help'. The terminal prompt is '(kali@kali)-[~]'. The user has entered the command '\$ sudo apt update && sudo apt full-upgrade -y'. The output shows 'Hit:1 http://http.kali.org/kali kali-rolling InRelease' and 'All packages are up to date.'

Ho semplicemente messo in sequenza i comandi con: **`&&`**
Invece ho bypassato la richiesta di conferma dell'upgrade con : **`-y`**

Successivamente dopo aver scaricato il pacchetto:

Packet_Tracer822_amd64_signed.deb

Ho navigato all'interno del folder Downloads per installarlo:



```
kali@kali: ~/Downloads
Session Actions Edit View Help

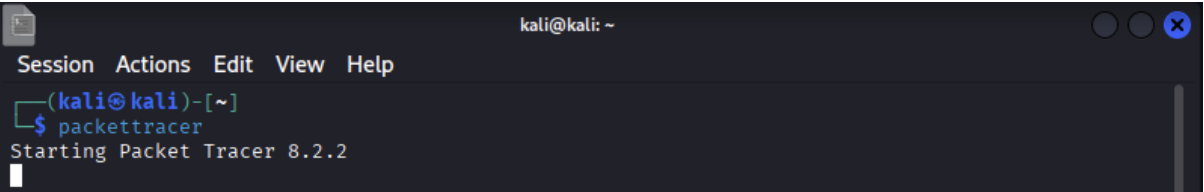
(kali@kali)-[~]
$ ls
Desktop  Downloads  Music  pt  Templates
Documents  equivs  Pictures  Public  Videos

(kali@kali)-[~]
$ cd Downloads

(kali@kali)-[~/Downloads]
$ ls
Packet_Tracer822_amd64_signed.deb

(kali@kali)-[~/Downloads]
$ sudo apt install ./Packet_Tracer822_amd64_signed.deb
[sudo] password for kali:
Note, selecting 'packettracer' instead of './Packet_Tracer822_amd64_signed.de
b'
packettracer is already the newest version (8.2.2).
```

Avvio il software: ***packettracer***



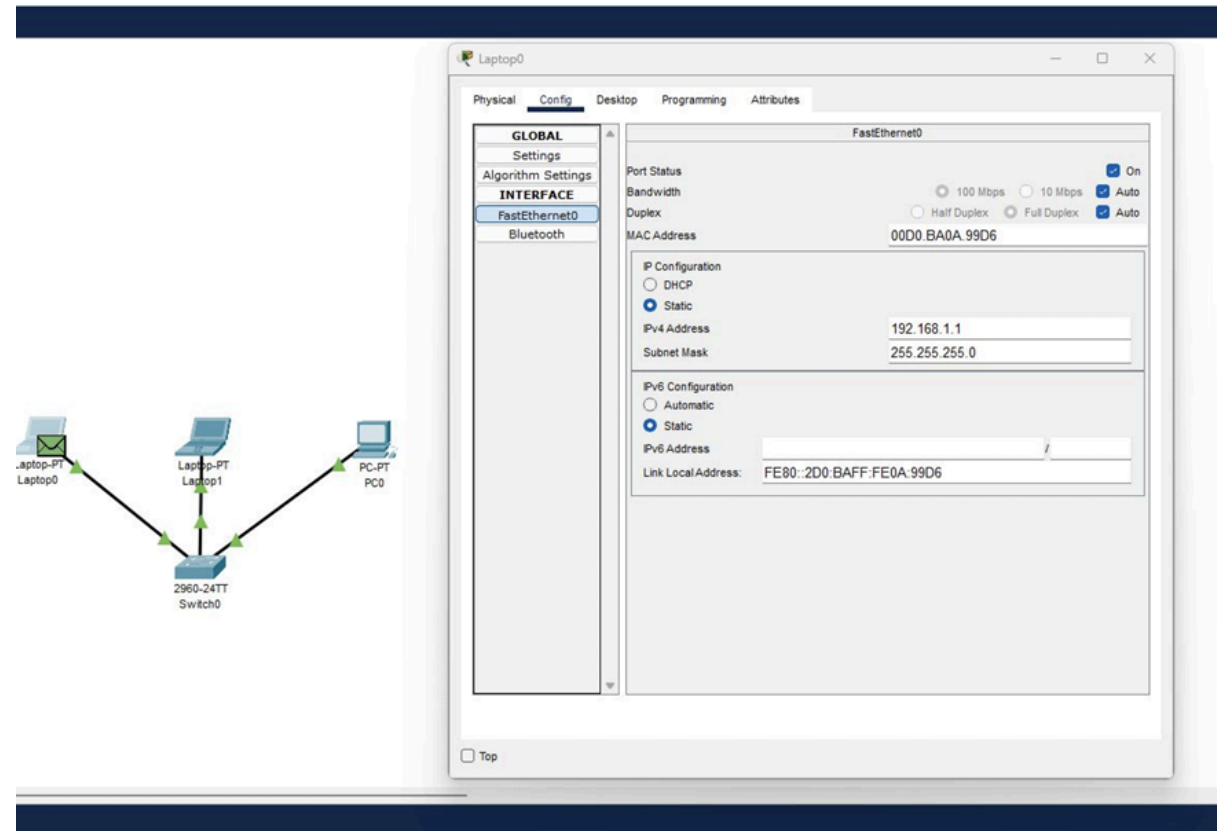
```
kali@kali: ~
Session Actions Edit View Help

(kali@kali)-[~]
$ packettracer
Starting Packet Tracer 8.2.2
█
```

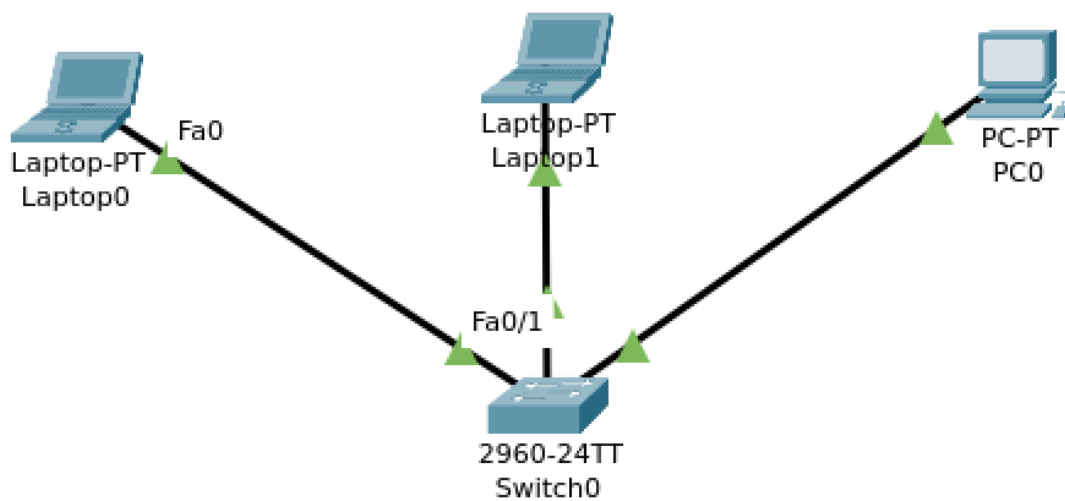
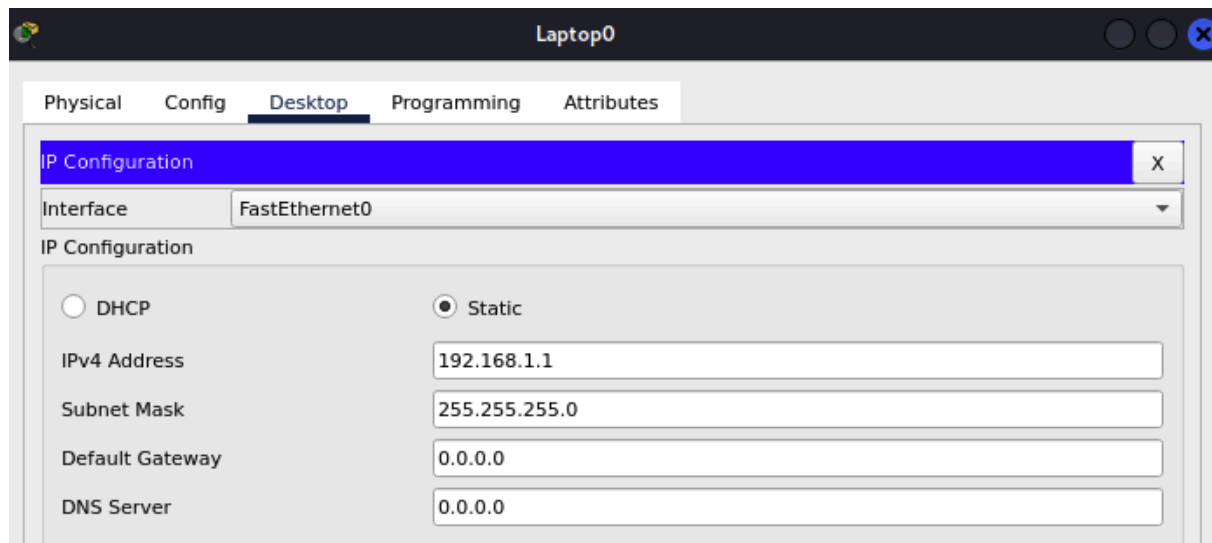
Il programma di per se è molto intuitivo, e seguendo le slide riesco a creare la rete, e a configurare tutti i dispositivi (lo switch non è richiesto). Con il seguente settaggio:

- Laptop0 192.168.1.1 / 255.255.255.0
- Laptop1 192.168.1.2 / 255.255.255.0
- PC0 192.168.1.3 / 255.255.255.0

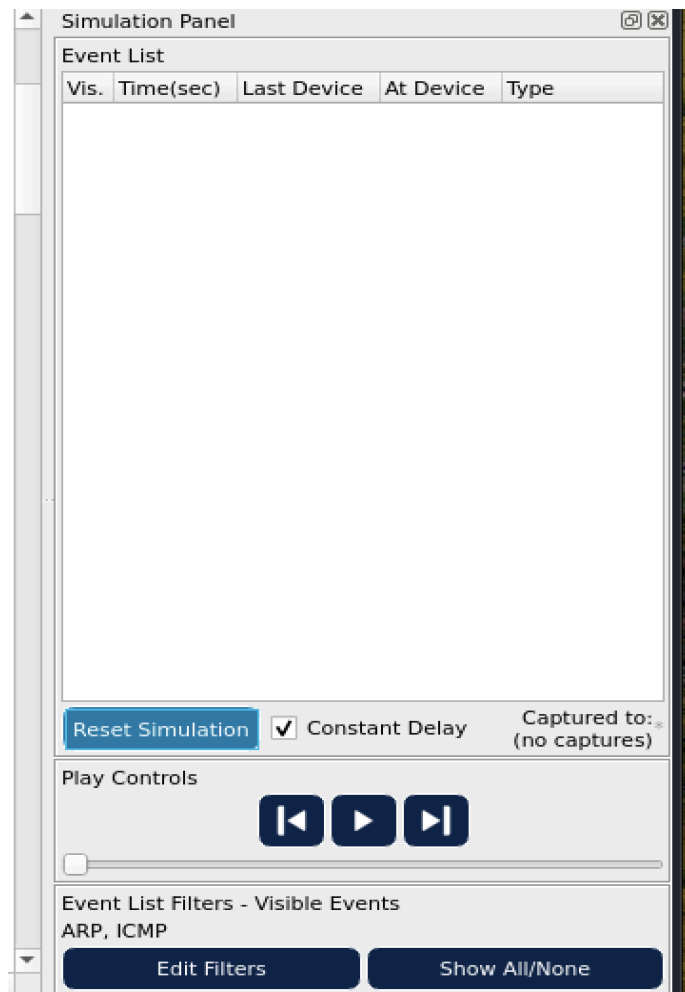
La slide della consegna dell'esercizio mette gli indirizzi da configurare sulla voce (Config)



Invece come ci è stato spiegato nella lezione la configurazione degli Ip e Subnet Mask viene eseguita cliccando sul dispositivo con tasto sinistro del mouse - Desktop - Ip Configuration



Una volta Configurati i dispositivi e cablata la connessione andremo ad aprire il nostro Simulation Panel:



In questo pannello possiamo vedere cosa avviene all'interno di questa **Broadcast**. Per questa consegna anche se non è stato suggerito ho utilizzato dei filtri, dato che l'esercizio richiedeva l'analisi dei primi due livelli, i protocolli che più ci interessano del modello **ISO/OSI**: **ARP** e **ICMP**.

ARP (Address Resolution Protocol):

Serve a tradurre un indirizzo IP in un indirizzo MAC. Per farlo, un dispositivo invia una richiesta in **Broadcast** a tutti i dispositivi collegati allo stesso switch. Solo il dispositivo giusto risponde con il suo MAC, e così il mittente sa a chi inviare i dati nella rete locale.

ICMP (Internet Control Message Protocol):

Serve per il controllo e la diagnostica della rete. Quando un dispositivo invia un ping, usa ICMP per mandare una richiesta ("Echo Request") a un altro host. Se l'host è attivo, risponde con un messaggio ("Echo Reply"), permettendo così di verificare la raggiungibilità e il tempo di risposta.

Qui possiamo vedere come all'interno del pannello vengono registrati gli eventi o comunicazioni tra i dispositivi collegati.

Come si può anche notare in basso ho applicato i 2 filtri in questo modo ho un registro degli eventi coerente allo scopo dell'esercizio.

Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	Laptop1	ICMP
	0.000	--	Laptop1	ARP
	0.001	Laptop1	Switch0	ARP
	0.002	Switch0	Laptop0	ARP
	0.002	Switch0	PC0	ARP
	0.003	PC0	Switch0	ARP
	0.004	Switch0	Laptop1	ARP
	0.004	--	Laptop1	ICMP
	0.005	Laptop1	Switch0	ICMP
	0.006	Switch0	PC0	ICMP
	0.007	PC0	Switch0	ICMP
	0.008	Switch0	Laptop1	ICMP
	1.008	--	Laptop1	ICMP
	1.009	Laptop1	Switch0	ICMP
	1.010	Switch0	PC0	ICMP
	1.011	PC0	Switch0	ICMP
	1.012	Switch0	Laptop1	ICMP
	2.015	--	Laptop1	ICMP
	2.016	Laptop1	Switch0	ICMP

Reset Simulation

☒ Constant Delay

Captured to:
153.039 s

Play Controls

☐

Event List Filters - Visible Events

ARP, ICMP

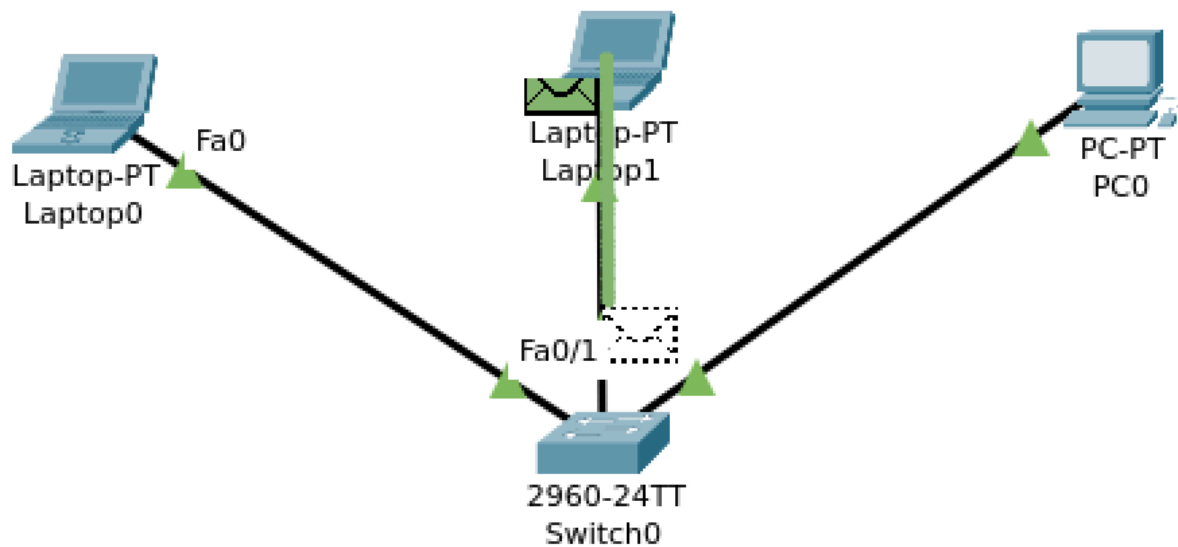
Edit Filters

Show All/None

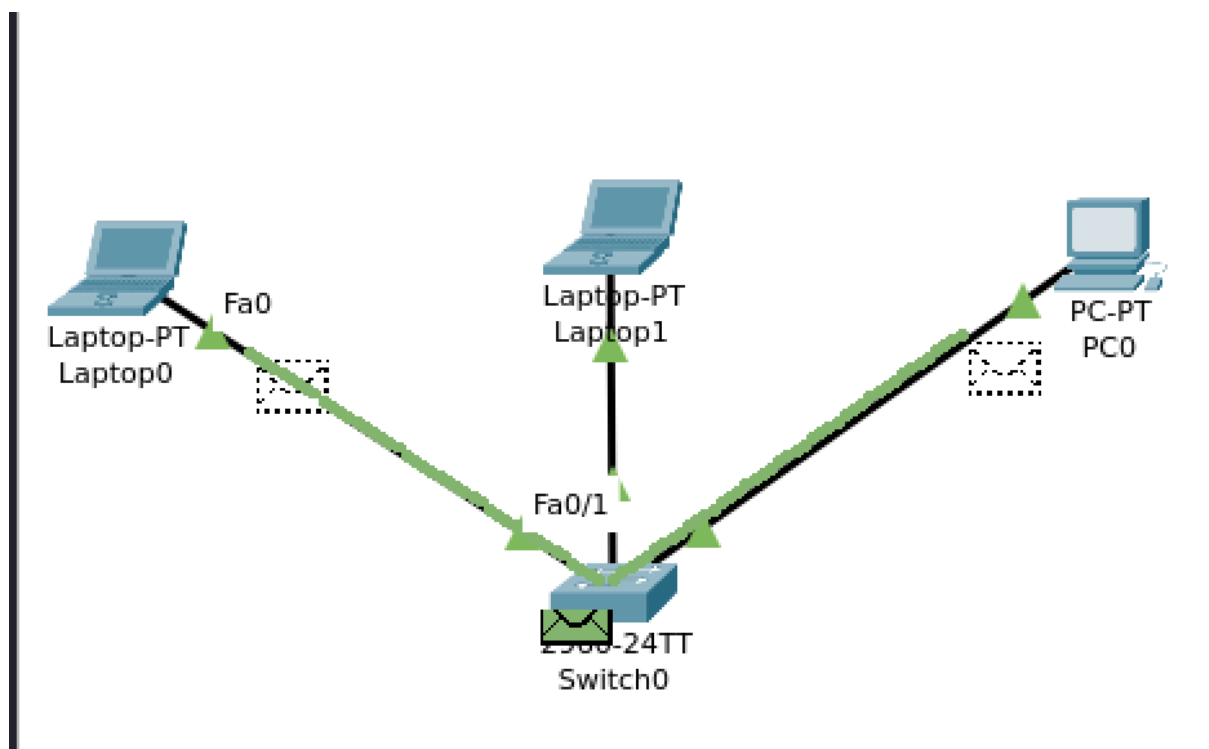
Prima di procedere all'analisi degli eventi, cliccando su Laptop1 troviamo il menù a tendina e clicchiamo su Desktop , poi su **CommandPrompt** dove all'interno possiamo eseguire il comando: **ping 192.168.1.3**

Prima di analizzare il terminale possiamo notare che il protocollo **ARP** cerca il destinatario mandando richieste a tutti i dispositivi in rete.

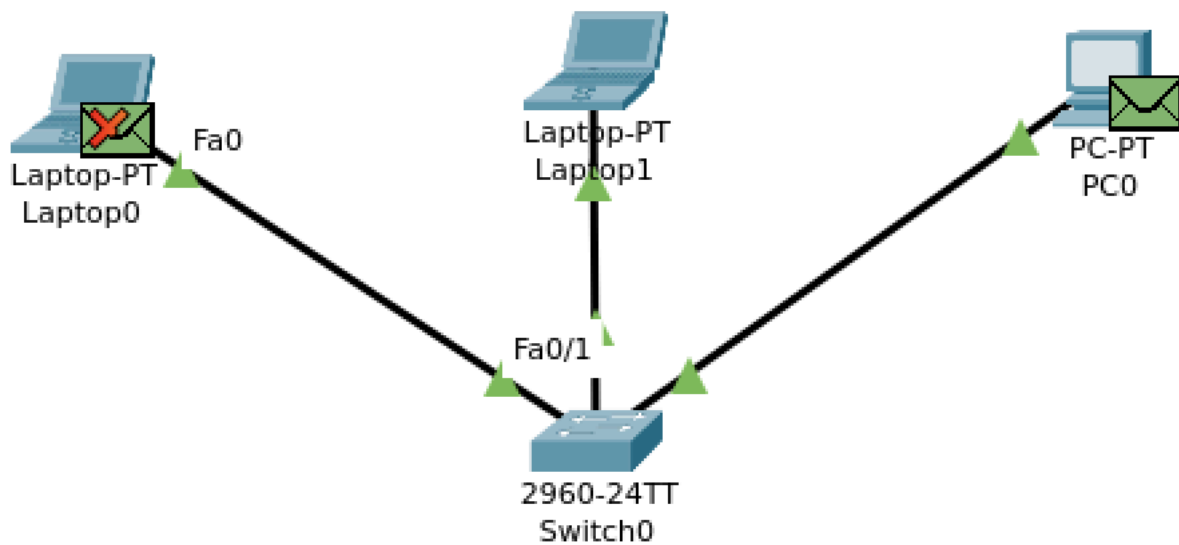
(Ma solo un dispositivo darà la conferma 192.168.1.3 ovvero PC0)



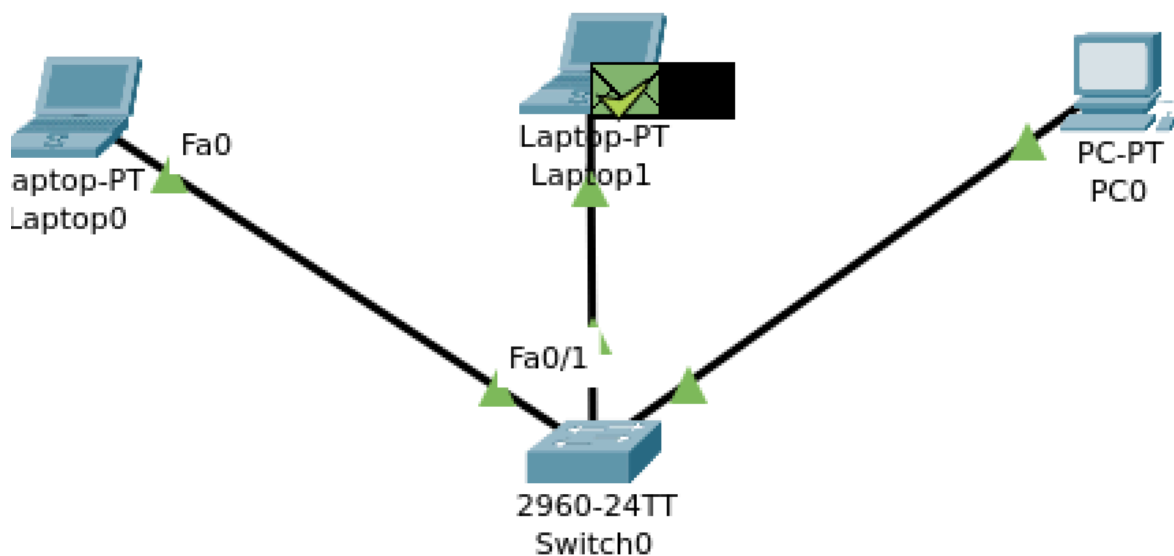
Vediamo come in questa figura **ARP** parte dal nostro Laptop1, e passando per lo Switch0



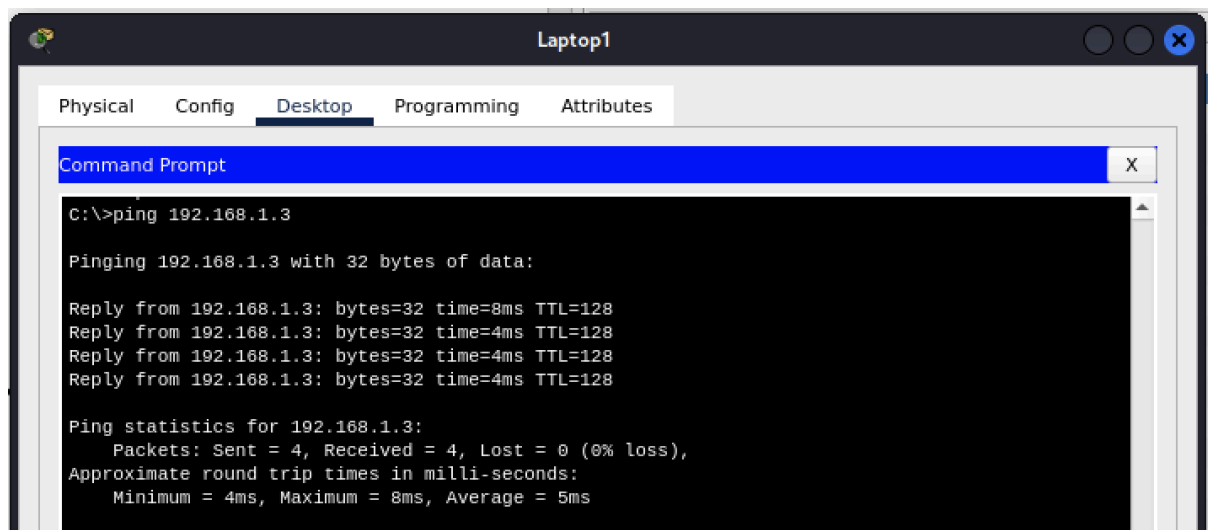
Arriva ai nostri dispositivi



Vediamo come sopra l'icona del nostro dispositivo Laptop0 si forma una X rossa. Questo ovviamente perchè non è il destinatario che abbiamo interpellato con la nostra richiesta **ping**.



In questa immagine **ARP** ha completato il suo lavoro la richiesta è tornata al mittente e adesso Laptop1 sa a chi mandare la richiesta **ping**.



```
C:\>ping 192.168.1.3

Pinging 192.168.1.3 with 32 bytes of data:

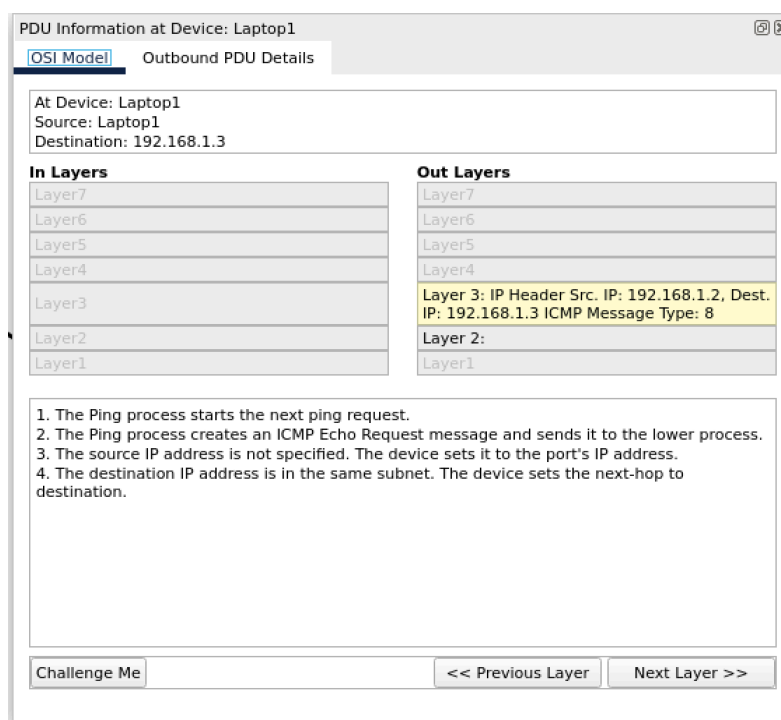
Reply from 192.168.1.3: bytes=32 time=8ms TTL=128
Reply from 192.168.1.3: bytes=32 time=4ms TTL=128
Reply from 192.168.1.3: bytes=32 time=4ms TTL=128
Reply from 192.168.1.3: bytes=32 time=4ms TTL=128

Ping statistics for 192.168.1.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 4ms, Maximum = 8ms, Average = 5ms
```

ping infatti è uno strumento di diagnostica della rete tramite il protocollo **ICMP**, che come abbiamo spiegato prima funziona tramite uno scambio di “messaggi” (“echo request”) (“echo reply”).

Da questo terminale possiamo vedere che tutte le richieste che sono state mandate sono state ricevute con successo e confermate per poi essere rimandate al mittente. Quindi possiamo evincere che la connessione tra i due dispositivi ora è presente.

Se apriamo ogni singolo evento sulla nostra event list si aprirà una finestra come questa. All'interno troviamo i nostri 7 Layers dello stack **ISO/OSI**. Quelli evidenziati sono quelli che ci interessano, infatti all'interno di ogni stringa in basso si apre una lista di informazioni in cui possiamo trovare anche il nome del protocollo e la funzione. Questo in particolare è il primo evento che troviamo, che al suo interno notiamo **header** che delimita Mittente e Destinatario: Laptop1 e Pc0, 192.168.1.2 e 192.168.1.3.



PDU Information at Device: Laptop1

OSI Model Outbound PDU Details









At Device: Laptop1
Source: Laptop1
Destination: 192.168.1.3

In Layers	Out Layers
Layer7	Layer7
Layer6	Layer6
Layer5	Layer5
Layer4	Layer4
Layer3	Layer3: IP Header Src. IP: 192.168.1.2, Dest. IP: 192.168.1.3 ICMP Message Type: 8
Layer2	Layer 2:
Layer1	Layer1

1. The Ping process starts the next ping request.
2. The Ping process creates an ICMP Echo Request message and sends it to the lower process.
3. The source IP address is not specified. The device sets it to the port's IP address.
4. The destination IP address is in the same subnet. The device sets the next-hop to destination.

Challenge Me << Previous Layer Next Layer >>

Se noi analizziamo invece gli eventi **ARP**:

Vis.	Time(sec)	Last Device	At Device	Type
	0.000	--	Laptop1	 ICMP
	0.000	--	Laptop1	 ARP
	0.001	Laptop1	Switch0	 ARP
	0.002	Switch0	Laptop0	 ARP
	0.002	Switch0	PC0	 ARP
	0.003	PC0	Switch0	 ARP
	0.004	Switch0	Laptop1	 ARP

Notiamo come le richieste passano da ogni dispositivo sulle colonne centrali:

Last Device e At Device. Per Poi tornare indietro al mittente. Una volta che Laptop1 riconosce il mittente , Laptop0 viene escluso dal resto degli eventi.

Simulation Panel

Event List

Vis.	Time(sec)	Last Device	At Device	Type
	0.004	--	Laptop1	ICMP
	0.005	Laptop1	Switch0	ICMP
	0.006	Switch0	PC0	ICMP
	0.007	PC0	Switch0	ICMP
	0.008	Switch0	Laptop1	ICMP
	1.008	--	Laptop1	ICMP
	1.009	Laptop1	Switch0	ICMP
	1.010	Switch0	PC0	ICMP
	1.011	PC0	Switch0	ICMP
	1.012	Switch0	Laptop1	ICMP
	2.015	--	Laptop1	ICMP
	2.016	Laptop1	Switch0	ICMP
	2.017	Switch0	PC0	ICMP
	2.018	PC0	Switch0	ICMP
	2.019	Switch0	Laptop1	ICMP
	3.023	--	Laptop1	ICMP
	3.024	Laptop1	Switch0	ICMP
	3.025	Switch0	PC0	ICMP
	3.026	PC0	Switch0	ICMP
	3.027	Switch0	Laptop1	ICMP

Reset Simulation

☒ Constant Delay

Captured to: 153.039 s

Play Controls

⏮

▶

⏭

Event List Filters - Visible Events

ARP, ICMP

Edit Filters

Show All/None

Qui invece vediamo che tutti i tipi di eventi hanno il protocollo **ICMP**

0.004	--	Laptop1	ICMP
0.005	Laptop1	Switch0	ICMP
0.006	Switch0	PC0	ICMP
0.007	PC0	Switch0	ICMP
0.008	Switch0	Laptop1	ICMP

Ogni ciclo è composto da questa serie di passaggi partendo e finendo sempre all Laptop1.

E per ogni ciclo si verifica una stringa come questa sul nostro prompt di comandi :

```
Reply from 192.168.1.3: bytes=32 time=9ms TTL=128
```

Reply from 192.168.1.3: bytes=32 time=9ms TTL=128

bytes=32 → dimensione del payload ricevuto.

time=9ms → RTT (round-trip time): tempo impiegato andata+ritorno (millisecondi).

TTL=128 → Time To Live rimanente nel pacchetto di risposta. Valore 128 è tipico di stack Windows; un TTL alto/uguale al default indica **pochissimi hop** (probabilmente host nella stessa LAN).