

Report finale del W4D4

Obiettivo

Lo scopo è di simulare un architettura client server tra una macchina windows che richiede, tramite web browser una risorsa alla macchina Kali Linux tramite l'hostname `epicode.internal` per poi, intercettare la comunicazione con wireshark evidenziando i MAC address sia di sorgente che di destinazione ed il contenuto della richiesta HTTPS.

Una volta concluso, ripetere il passaggio sostituendo il server HTTPS con un server HTTP.

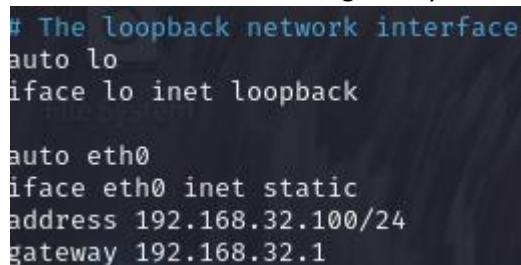
Requisiti

1. Una macchina virtuale Kali Linux con indirizzo IP 192.168.32.100
2. Una macchina Windows 7 con indirizzo IP 192.168.32.101
3. Un server HTTPS attivo
4. Un servizio DNS per la risoluzione di nomi di dominio attivo.

Step 1: assegnazione degli indirizzi IP

Kali Linux

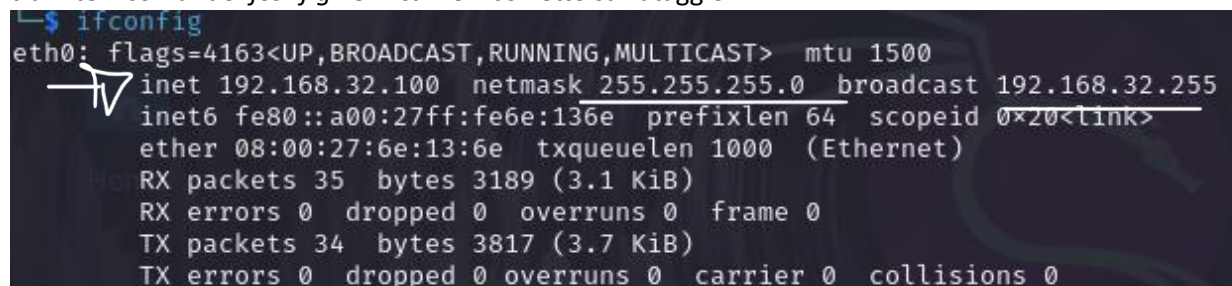
1. Aprire il terminale di Kali e digitare il seguente comando: `sudo nano /etc/network/interfaces` e settare indirizzo IP statico e gateway come segue da foto:



```
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.32.100/24
gateway 192.168.32.1
```

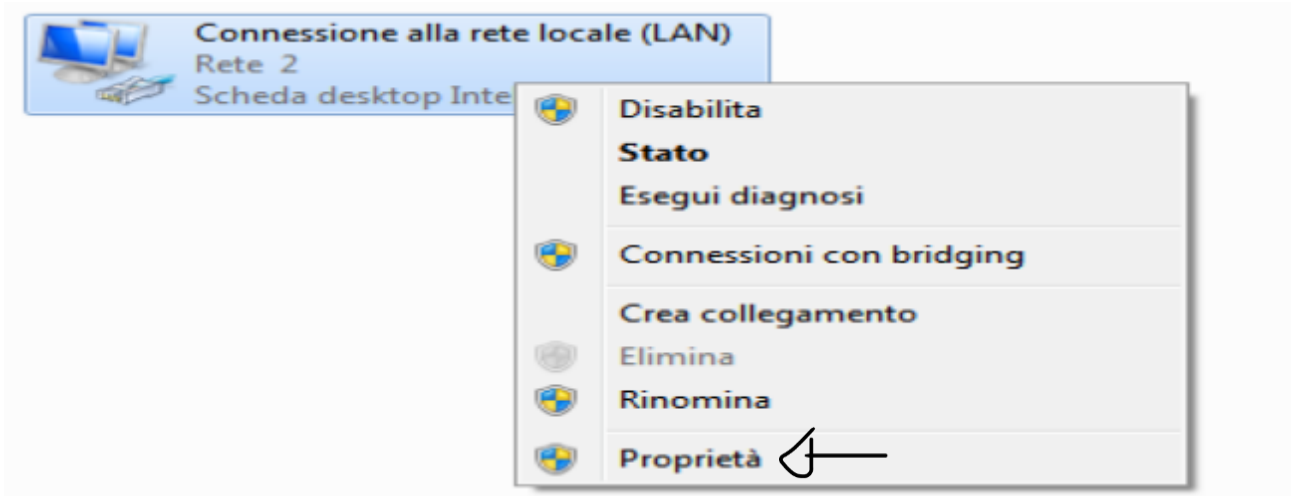
2. Salviamo i settaggi per poi riavviare il tutto con il comando `sudo systemctl restart networking` e, tramite il comando `ifconfig` verificarne il corretto salvataggio.



```
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.32.100 netmask 255.255.255.0 broadcast 192.168.32.255
    inet6 fe80::a00:27ff:fe6e:136e prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6e:13:6e txqueuelen 1000 (Ethernet)
    RX packets 35 bytes 3189 (3.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 34 bytes 3817 (3.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Windows

1. Aprire la schermata del “centro connessioni di rete e condivisione” per andare nelle proprietà della nostra scheda di rete come da immagine



2. Una volta dentro le proprietà aprire la sezione “protocollo internet versione 4 (TCP/IPv4)” ed impostare indirizzo IP e DNS come illustrato

☐ Ottieni automaticamente un indirizzo IP

☒ Utilizza il seguente indirizzo IP:

Indirizzo IP:	192 . 168 . 32 . 101
Subnet mask:	255 . 255 . 255 . 0
Gateway predefinito:	192 . 168 . 32 . 1

☐ Ottieni indirizzo server DNS automaticamente

☒ Utilizza i seguenti indirizzi server DNS:

Server DNS preferito:	192 . 168 . 32 . 100
Server DNS alternativo:	. . .

Salvare e dare un reboot al sistema.

Step 2: Settare i protocolli HTTP/HTTPS

Torniamo sulla macchina Kali per attivare i protocolli HTTP e HTTPS. Per questo passaggio è necessario andare a lavorare con inetsim, ovvero un software progettato per la simulazione di servizi internet all'interno di laboratori. È usato per analisi di rete e test di malware permettendoci di osservare i comportamenti di programmi sospetti senza esporre una rete reale a rischi.

Per accedervi aprire il terminale di Kali e digitare la seguente stringa: `sudo nano /etc/inetsim/inetsim.conf`

Una volta inserita la password (comunemente “Kali” se si ha scaricato la macchina dal sito ufficiale) ci comparirà una pagina immensa, ma a noi interessano poche righe.

Cominciamo abilitando i protocolli http e https togliendo il # davanti alle righe che sta a simboleggiare un commento.

```
#start_service dns
start_service http
start_service https
#start_service smtp
#start_service smtps
```

Scendendo di qualche riga andiamo nel “paragrafo” del `service_bind_address` per andare a togliere il # dall’ultima riga che riporta `service_bind_address` e settarlo a 0.0.0.0

Il service bind address è un’opzione di configurazione che specifica all’indirizzo IP quale servizi dell’inetsim simulati devono essere associati e, lo impostiamo a 0.0.0.0 affinché inetsim ascolti su tutti gli indirizzi IPv4 disponibili sulla macchina locali.

```
#####  
# service_bind_address  
#  
# IP address to bind services to  
#  
# Syntax: service_bind_address <IP address>  
#  
# Default: 127.0.0.1  
#  
service_bind_address 0.0.0.0
```

Step 3: Assegnazione del DNS

Per questo passaggio basterebbe settare il DNS sempre da Kali Linux tramite inetsim ma, per un bug della versione che non ne permette il corretto funzionamento andremo a vedere come attivarlo configurando anche la sezione host.

Settaggio DNS Kali

Siccome inetsim da problemi apriamo il terminale e digitiamo `sudo nano /etc/hosts` per aggiungere nell’ultima riga la stringa “192.168.32.100 epicode.internal”.

Questo serve a creare una mappatura statica tra il nome di dominio e un indirizzo IP, simulando così il servizio di DNS locale funzionante.

```
GNU nano 8.2  
127.0.0.1    localhost  
127.0.1.1    kali  
::1          localhost ip6-localhost ip6-loopback  
ff02::1      ip6-allnodes  
ff02::2      ip6-allrouters  
192.168.32.100 epicode.internal
```

La riga in giallo è la stringa da aggiungere

Settaggio DNS

Windows

Oltre che impostare il DNS dalle impostazioni come visto nello step 1, avendo i problemi con inetsim di Linux dobbiamo anche qui andare a sovrascrivere il file hosts del sistema.

Per fare ciò prima dobbiamo trovare il file da modificare con i privilegi da amministratore seguendo il seguente percorso - `C:\Windows\System32\drivers\etc\hosts`

Aprire il file denominato “hosts” con blocco note ed aggiungere come ultima riga “192.168.32.100 epicode.internal”.

```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com           # source server
#       38.25.63.10       x.acme.com               # x client host
#
# localhost name resolution is handled within DNS itself.
#       127.0.0.1         localhost
#       ::1               localhost
192.168.32.100 epicode.internal
```

Evidenziata c'è la stringa da aggiungere

Salvare il file, e riavviare il sistema per una maggiore sicurezza che abbia preso la modifica.

Step 4: Verifica e Ping

Per verificare che tutto funzioni, aprire il terminale di Kali Linux e digitare il comando *Sudo inetsim* per avviare la simulazione del server.

Andare su Windows, aprire il CMD e provare a pingare il dominio "Epicode.internal"

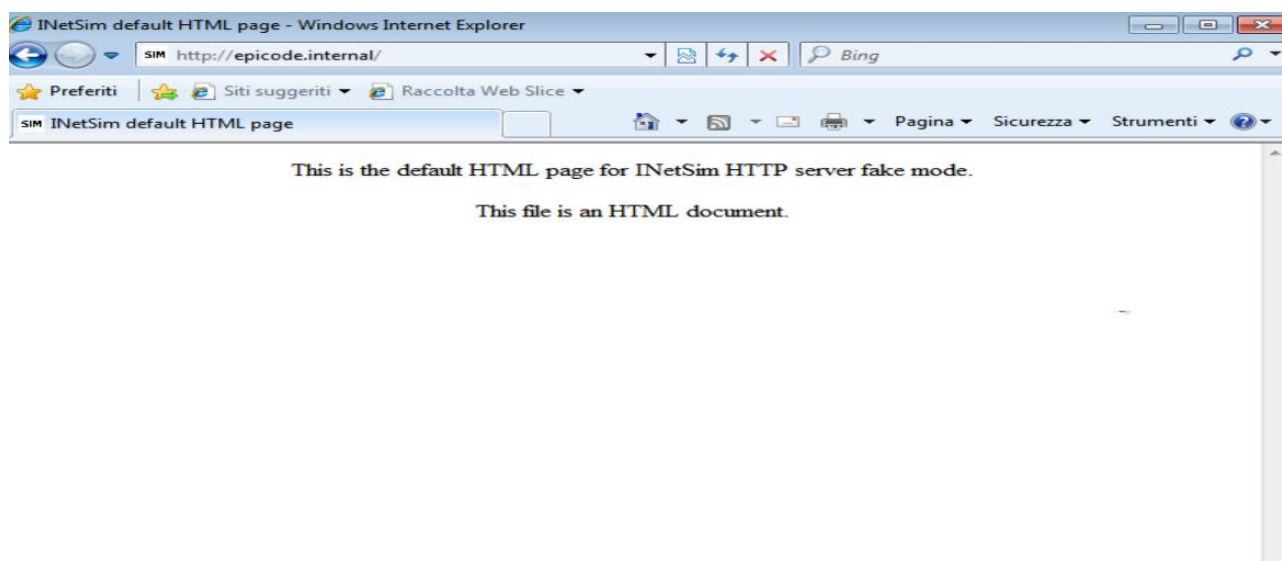
Se il ping va a buon fine allora il sistema funziona. Per avere un'ultima conferma aprire un motore di ricerca e cercare nella barra di ricerca "epicode.internal"

```
C:\Users\Administrator>ping epicode.internal

Esecuzione di Ping epicode.internal [192.168.32.100] con 32 byte di dati:
Risposta da 192.168.32.100: byte=32 durata=1ms TTL=64
Risposta da 192.168.32.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.32.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.32.100: byte=32 durata<1ms TTL=64

Statistiche Ping per 192.168.32.100:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 0ms, Massimo = 1ms, Medio = 0ms
```

Ping da windows a epicode.internal andato a buon fine



Accesso tramite Internet Explorer a Epicode.internal andato a buon fine

Step 5: Sniffing dei pacchetti con Wireshark

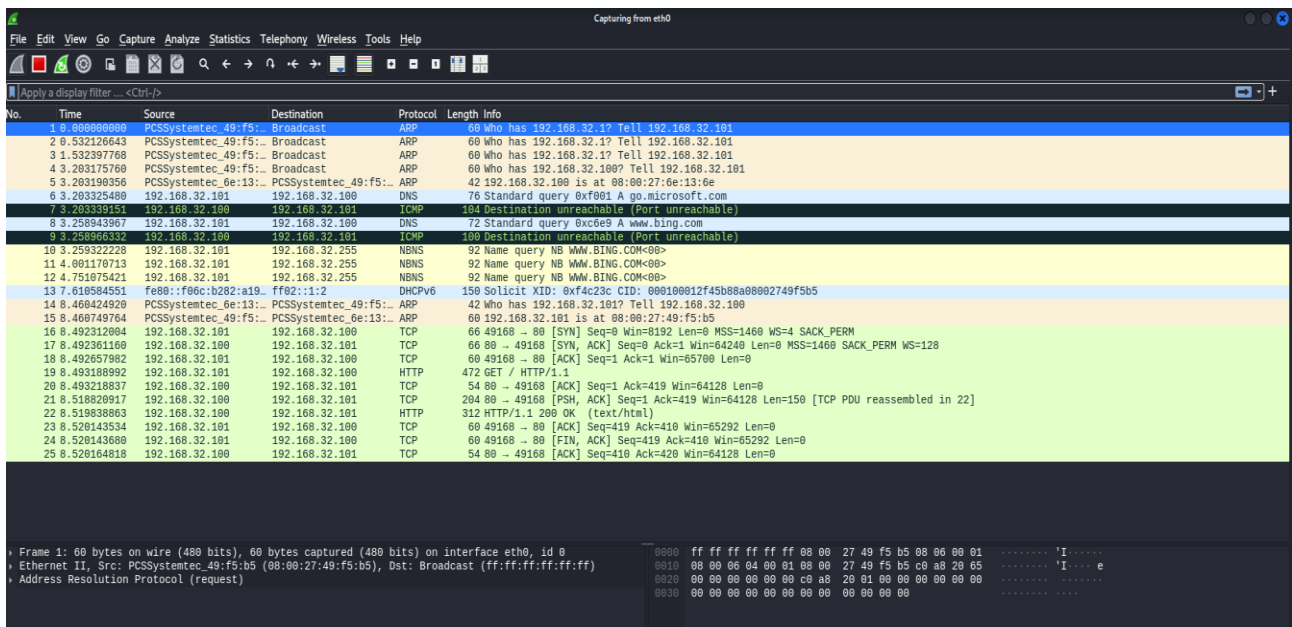
Analisi dei pacchetti tramite accesso HTTP

Apriamo Wireshark, il nostro programma dedicato al monitoraggio ed allo sniffing della rete. Una volta dentro al programma selezionare la rete che desideriamo monitorare, nel nostro caso eth0.



Una volta dentro andiamo sulla macchina di windows e cerchiamo da un motore di ricerca il nostro dominio, in questo caso “epicode.internal”. A questo punto il nostro programma comincerà a leggere tutti i pacchetti sia in uscita che in entrata.

Come si può notare nella riga 6, relativa al protocollo DNS, possiamo vedere le nostre macchine in comunicazione con la sorgente della richiesta l’IP 192.168.32.101, ovvero la macchina windows che sta richiedendo l’accesso a epicode.internal e come destinatario l’IP 192.168.32.100, ovvero la macchina Kali che funge da host.



Finestra di wireshark con tutti i pacchetti sia in entrata che in uscita

Essendo in HTTP possiamo subito notare che i pacchetti sono chiari, ovvero senza alcuna cifratura (plaintext).

1	0.000000000	PCSSystemtec_49:f5:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
2	0.974486893	PCSSystemtec_49:f5:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
3	1.973796950	PCSSystemtec_49:f5:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
4	3.233737084	PCSSystemtec_49:f5:...	Broadcast	ARP	60	Who has 192.168.32.100? Tell 192.168.32.101
5	3.233747860	PCSSystemtec_6e:13:...	PCSSystemtec_49:f5:...	ARP	42	192.168.32.100 is at 08:00:27:6e:13:6e
6	3.233828338	192.168.32.101	192.168.32.100	DNS	76	Standard query 0x64ad A go.microsoft.com
7	3.233843643	192.168.32.100	192.168.32.101	ICMP	104	Destination unreachable (Port unreachable)
8	3.274067818	192.168.32.101	192.168.32.100	DNS	72	Standard query 0x4a6f A www.bing.com
9	3.274091573	192.168.32.100	192.168.32.101	ICMP	100	Destination unreachable (Port unreachable)
10	3.274092353	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WWW.BING.COM<00>
11	4.023133183	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WWW.BING.COM<00>
12	4.772922767	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WWW.BING.COM<00>
13	8.280635251	PCSSystemtec_6e:13:...	PCSSystemtec_49:f5:...	ARP	42	Who has 192.168.32.101? Tell 192.168.32.100
14	8.281647114	PCSSystemtec_49:f5:...	PCSSystemtec_6e:13:...	ARP	60	192.168.32.101 is at 08:00:27:49:f5:b5
15	9.025700935	192.168.32.101	192.168.32.100	TCP	66	49175 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
16	9.025746254	192.168.32.100	192.168.32.101	TCP	66	80 → 49175 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
17	9.025850885	192.168.32.101	192.168.32.100	TCP	60	49175 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
18	9.025904846	192.168.32.101	192.168.32.100	HTTP	472	GET / HTTP/1.1
19	9.025911117	192.168.32.100	192.168.32.101	TCP	54	80 → 49175 [ACK] Seq=1 Ack=419 Win=64128 Len=0
20	9.035423894	192.168.32.100	192.168.32.101	TCP	204	80 → 49175 [PSH, ACK] Seq=1 Ack=419 Win=64128 Len=150 [TCP PDU reassembled in 21]
21	9.036641445	192.168.32.100	192.168.32.101	HTTP	312	HTTP/1.1 200 OK (text/html)
22	9.036913688	192.168.32.101	192.168.32.100	TCP	60	49175 → 80 [ACK] Seq=419 Ack=410 Win=65292 Len=0
23	9.036970736	192.168.32.101	192.168.32.100	TCP	60	49175 → 80 [FIN, ACK] Seq=419 Ack=410 Win=65292 Len=0
24	9.036977939	192.168.32.100	192.168.32.101	TCP	54	80 → 49175 [ACK] Seq=410 Ack=420 Win=64128 Len=0

Dalla riga 15 alla riga 17 si può notare la “three-hand-shake” con la seguente comunicazione:

riga 15: seq=0

riga 16: seq=0 – ack=1

Riga 17: seq:1 – ack=1

Inoltre si può chiaramente vedere come studiato in slide, la richiesta GET dell’HTTP verb, che viene utilizzato quando si richiede una risorsa al web.

```
GET / HTTP/1.1\r\n
Accept: application/x-ms-application,
Accept-Language: en-US\r\n
User-Agent: Mozilla/4.0 (compatible;
Accept-Encoding: gzip, deflate\r\n
Host: epicode.internal\r\n
Connection: Keep-Alive\r\n
\r\n
```

Richiesta HTTP reale

GET / HTTP/1.1

Host: *www.google.com*

User-Agent: *Microsoft Edge/1.0*
(*Microsoft Windows*)

Accept: *text/html*

Accept-Language: *it-IT*

Connection: *keep-alive*

<qui va il corpo del messaggio>

Richiesta HTTP da studio per paragonare wireshark alla teoria

Analisi dei pacchetti tramite accesso HTTPS

Come nell'analisi del collegamento HTTP si potrà notare la "three-hand-shake", gli indirizzi IP sia della macchina sorgente (windows) sia della macchina destinataria (Kali) con la differenza che essendo HTTPS il contenuto dei pacchetti risulta cifrato (ciphertext). Questa cifratura è grazie al protocollo TLSv1 che si occupa sia di crittare che decrittare il contenuto dei pacchetti.

```
30 17.256505714 192.168.32.101 192.168.32.100 TLSv1
Transmission Control Protocol, Src Port: 49176, Dst Port: 443, Seq: 1,
Transport Layer Security
  TLSv1 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 124
  Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 120
    Version: TLS 1.0 (0x0301)
    Random: 67e17cb597e146e8e72e0f254e9ca2c961c8f224b1681cf8fb5e957d
      GMT Unix Time: Mar 24, 2025 16:39:33.000000000 CET
      Random Bytes: 97e146e8e72e0f254e9ca2c961c8f224b1681cf8fb5e957d
    Session ID Length: 0
    Cipher Suites Length: 24
    Cipher Suites (12 suites)
```

CONCLUSIONI E CONSIDERAZIONI PERSONALI

Per concludere, è possibile visualizzare non solo gli indirizzi IP delle macchine che comunicano tra di loro, ma anche i MAC address, ovvero l'indirizzo fisico della macchina che si può visualizzare nei pacchetti DNS

```
16 3.289440494 192.168.32.101 192.168.32.100 DNS 72 Standard query 0xf638 A www.bing.com
17 3.289462837 192.168.32.100 192.168.32.101 ICMP 100 Destination unreachable (Port unreachable)
18 3.289650432 192.168.32.101 192.168.32.255 NBNS 92 Name query NB WWW.BING.COM<00>

Frame 16: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface eth0, id 0
Ethernet II, Src: PCSSystemtec_49:f5:b5 (08:00:27:49:f5:b5), Dst: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Destination: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Source: PCSSystemtec_49:f5:b5 (08:00:27:49:f5:b5)
Type: IPv4 (0x0800)
[Stream index: 5]
Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
```

Visualizzazione degli indirizzi IP e MAC address

Come da immagine si vede il MAC address della macchina sorgente 08:00:27:49:f5:b5 (windows) e il MAC address del destinatario 08:00:27:6e:13:6e (Kali).

Si può delineare che nonostante entrambi gli indirizzi (IP e MAC address) siano visibili in tutti e due i protocolli (HTTPS e HTTP) nel primo caso avremo una maggiore sicurezza con una crittografia mentre nel secondo caso avremo tutto in chiaro.

PS rivolto al professore:

per una onesta chiarezza ho dovuto chieder aiuto a ChatGPT 4.5 per trovare una soluzione al malfunzionamento di inetsim che mi ha suggerito in primo luogo l'utilizzo di software come apache e DNSmasq ma, siccome lei ha detto che vi era un modo per farlo anche senza l'accesso a internet ho fatto il possibile per riuscirci e, dopo un giorno a sbatterci la testa, sempre con l'aiuto di GPT ho settato il file "hosts" sia su Kali che su Windows e per l'HTTPS ho creato un certificato denominato epicode.internal convertito in .pem per farlo leggere a inetsim. Altrimenti non ci sarei mai arrivato di mio.

Sperando che la mia onestà non mi penalizzi, le garantisco che il resto lo svolto in mia completa autonomia.