

# PROGETTO

## Requisiti e servizi:

- Kali Linux: IP 192.168.32.100
- Windows: IP 192.168.32.101
- HTTPS server: attivo
- Servizio DNS per risoluzione nomi di dominio: attivo

## Traccia:

Simulare, in ambiente di laboratorio virtuale, un'architettura client server in cui un client con indirizzo 192.168.32.101(Windows) richiede tramite web browser una risorsa all'hostname epicode.internal che risponde all'indirizzo 192.168.32.100(Kali).

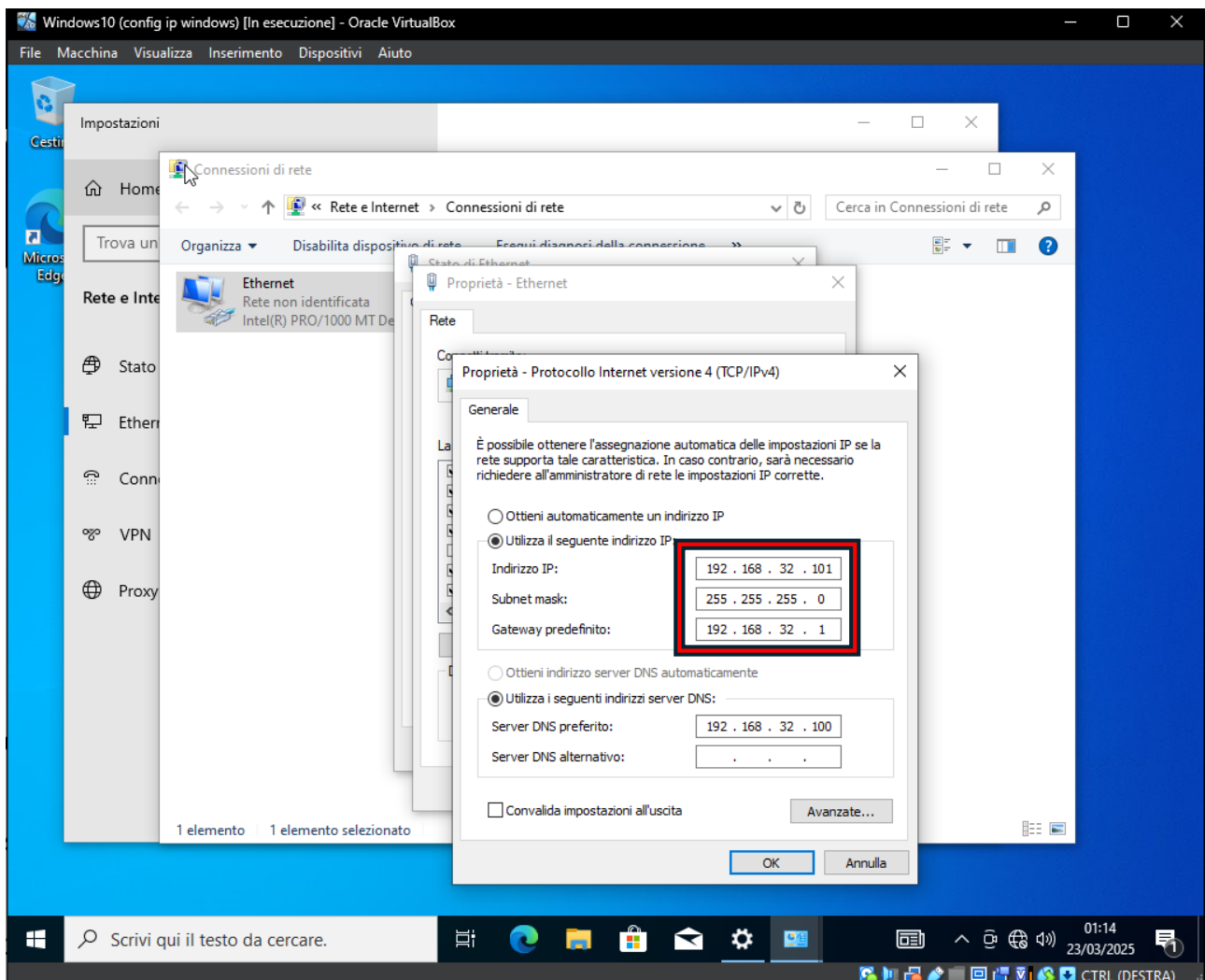
Si intercetti poi la comunicazione con Wireshark, evidenziando i MAC address di sorgente e destinazione ed il contenuto della richiesta HTTPS.

Ripetere l'esercizio, sostituendo il server HTTPS, con un server HTTP. Si intercetti nuovamente il traffico, evidenziando le eventuali differenze tra il traffico appena catturato in HTTP ed il traffico precedente in HTTPS. Spiegare, motivandole, le principali differenze se presenti.

## Svolgimento:

- Per prima cosa, come nei precedenti esercizi avvio le due macchine virtuali create con VirtualBox e configuro le schede di rete virtuali su rete interna.
- Ora andremo ad assegnare gli indirizzi IP sia a Windows (192.168.32.101) che a Kali Linux (192.168.32.100).

### WINDOWS:



```
Windows10 (config ip windows) [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

Amministratore: Prompt dei comandi
Microsoft Windows [Versione 10.0.19045.3803]
(c) Microsoft Corporation. Tutti i diritti sono riservati.

C:\Windows\system32>ipconfig /all

Configurazione IP di Windows

Nome host . . . . . : DESKTOP-5T0VKJT
Suffisso DNS primario . . . . . :
Tipo nodo . . . . . : Ibrido
Routing IP abilitato. . . . . : No
Proxy WINS abilitato . . . . . : No

Scheda Ethernet Ethernet:

Suffisso DNS specifico per connessione:
Descrizione . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Indirizzo fisico. . . . . : 08-00-27-4B-7A-9E
DHCP abilitato. . . . . : No
Configurazione automatica abilitata . . . . . : Si
Indirizzo IPv6 locale rispetto al collegamento . . : fe80::3076:adb8:5d4:4b7c%5(Preferenziale)
Indirizzo IPv4. . . . . : 192.168.32.101(Preferenziale)
Subnet mask . . . . . : 255.255.255.0
Gateway predefinito . . . . . : 192.168.32.1
IAID DHCPv6 . . . . . : 101187623
GUID Client DHCPv6. . . . . : 00-01-00-01-2F-0E-3E-0D-08-00-27-4B-7A-9E
Server DNS . . . . . : 192.168.32.100
NetBIOS su TCP/IP . . . . . : Attivato

C:\Windows\system32>
```

## KALI:

```
kali-linux-2024.4-virtualbox-amd64 [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

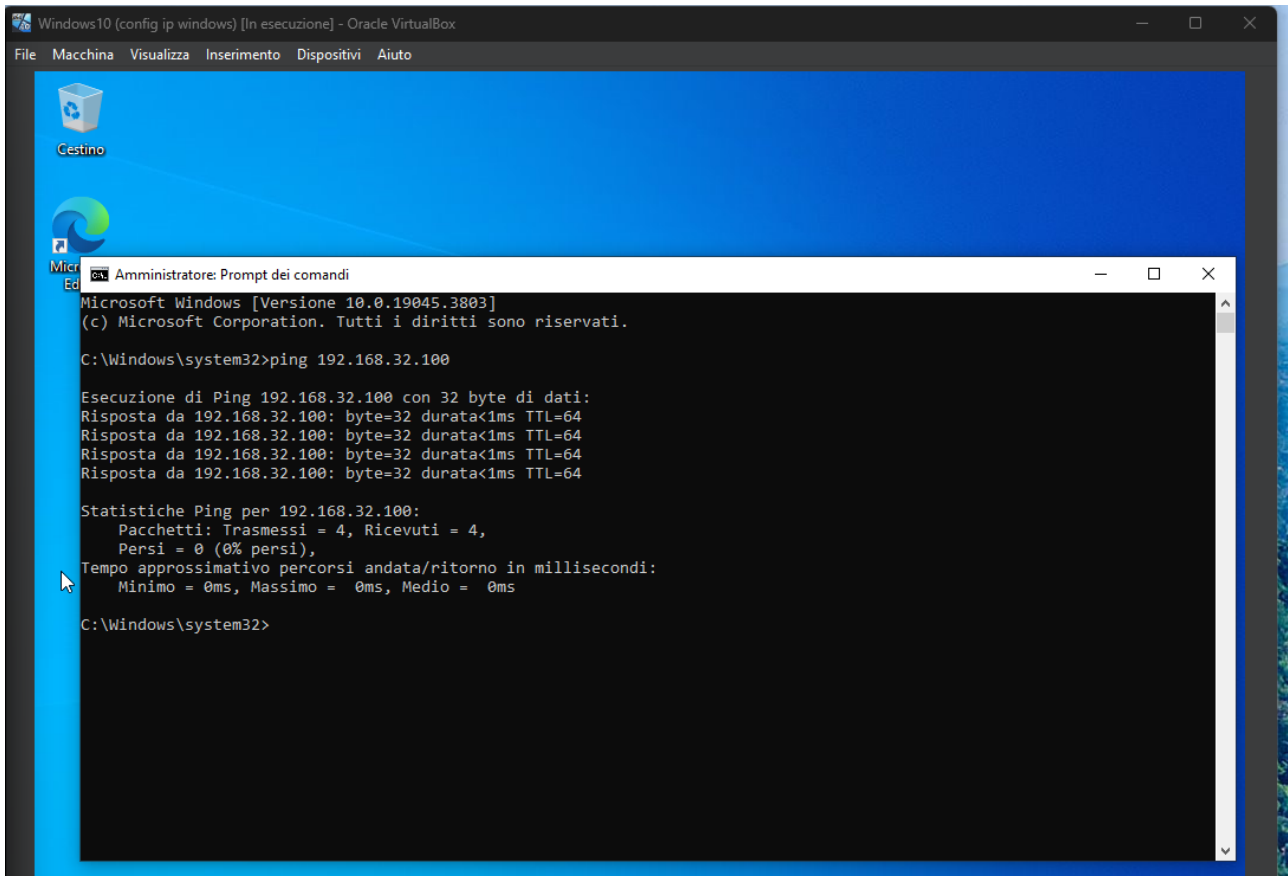
kali@epicode: ~
File Actions Edit View Help
* http_80_tcp - stopped (PID 25651)
Simulation stopped.
Report written to '/var/log/inetSim/report/report.25644.txt' (13 lines)
== INetSim main process stopped (PID 25644) ==

(kali@epicode)-[~]
$ 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::ea40:c073:c2e7:7e11 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6e:13:6e txqueuelen 1000 (Ethernet)
    RX packets 4997 bytes 430877 (420.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2308 bytes 266103 (259.8 KiB)
    TX errors 0 dropped 6 overruns 0 carrier 0 collisions 0

    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 242 bytes 70092 (68.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 242 bytes 70092 (68.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@epicode)-[~]
$
```

- Impostati gli indirizzi IP alle macchine faccio un test di comunicazione tra le due macchine tramite un PING.



```
Windows10 (config ip windows) [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

Cestino

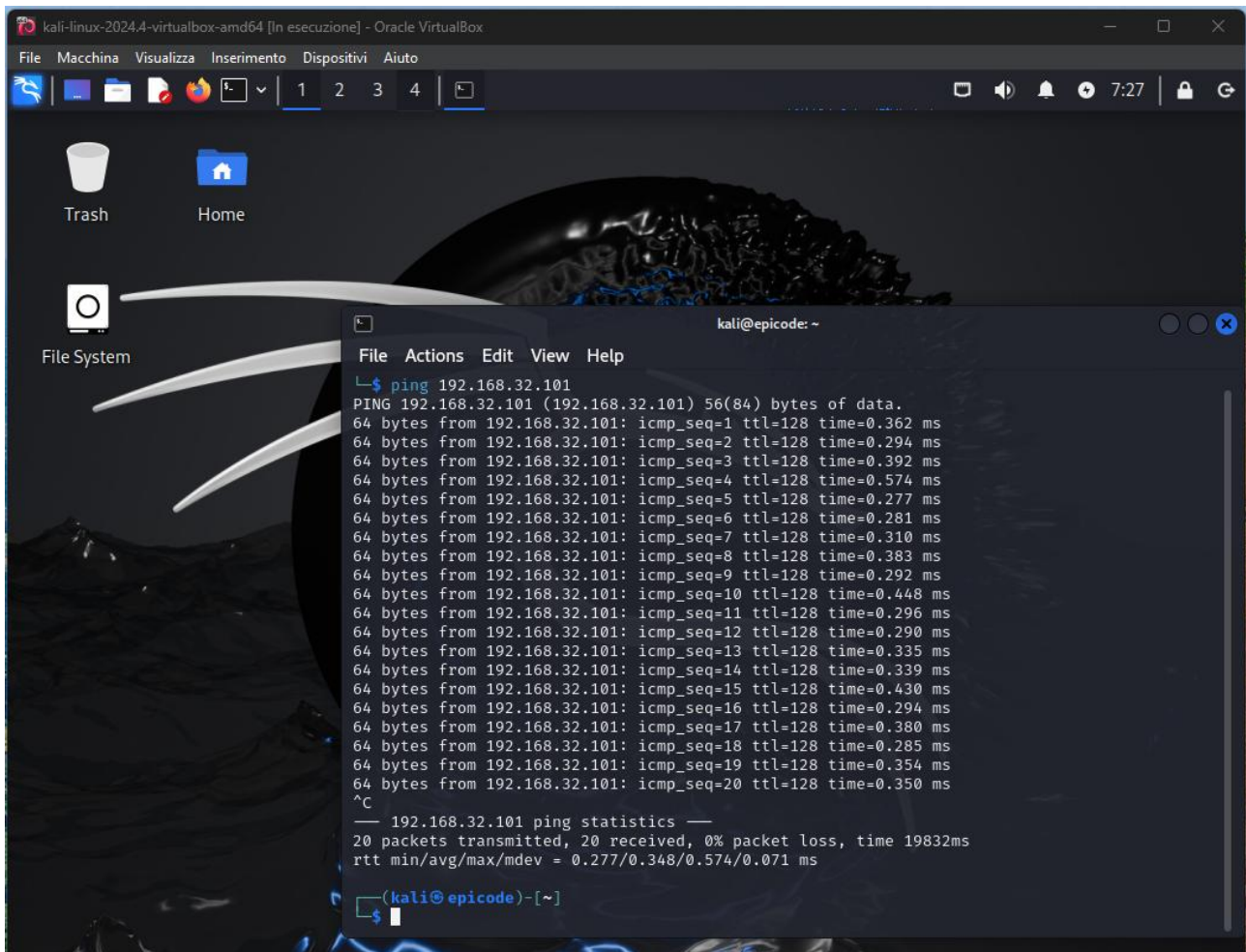
Micro Ed
Amministratore: Prompt dei comandi
Microsoft Windows [Versione 10.0.19045.3803]
(c) Microsoft Corporation. Tutti i diritti sono riservati.

C:\Windows\system32>ping 192.168.32.100

Esecuzione di Ping 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 = 0ms, Medio = 0ms

C:\Windows\system32>
```



```
kali-linux-2024.4-virtualbox-amd64 [In esecuzione] - Oracle VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

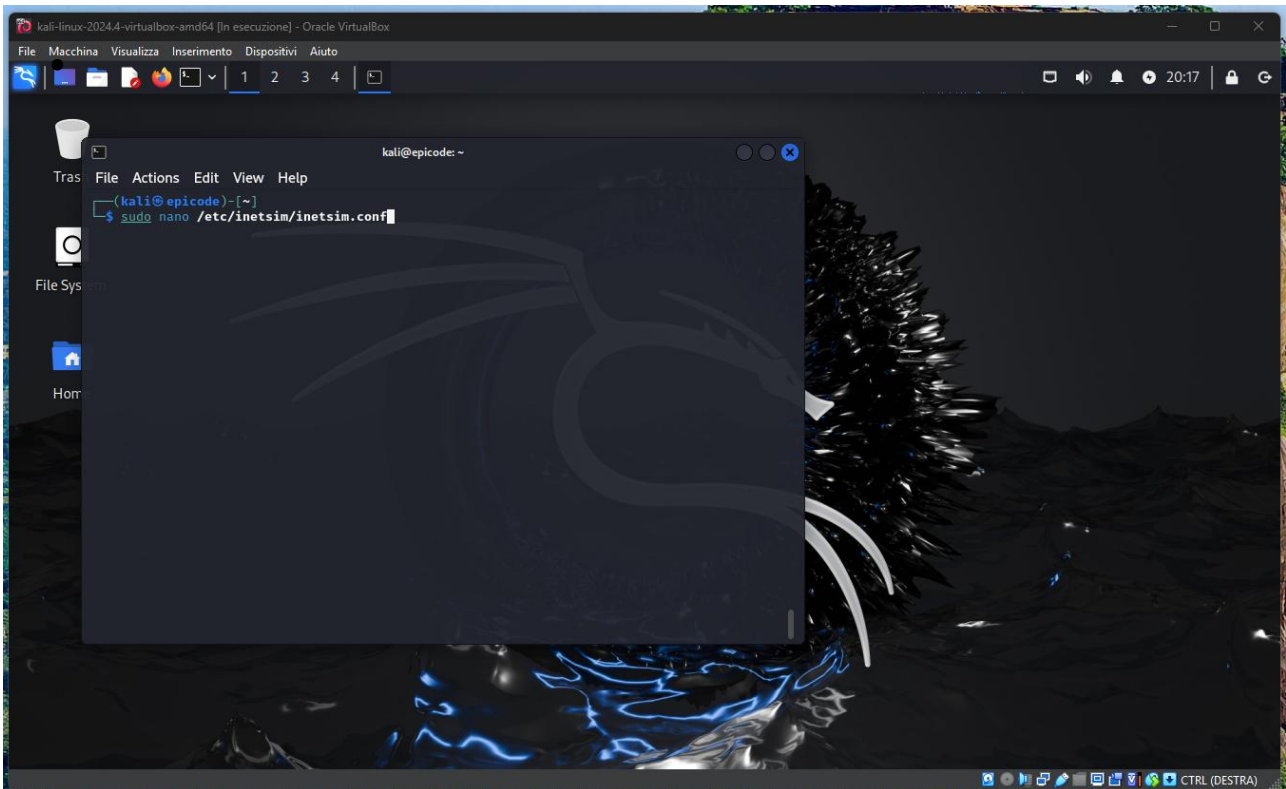
Trash Home
File System

kali@epicode: ~
File Actions Edit View Help

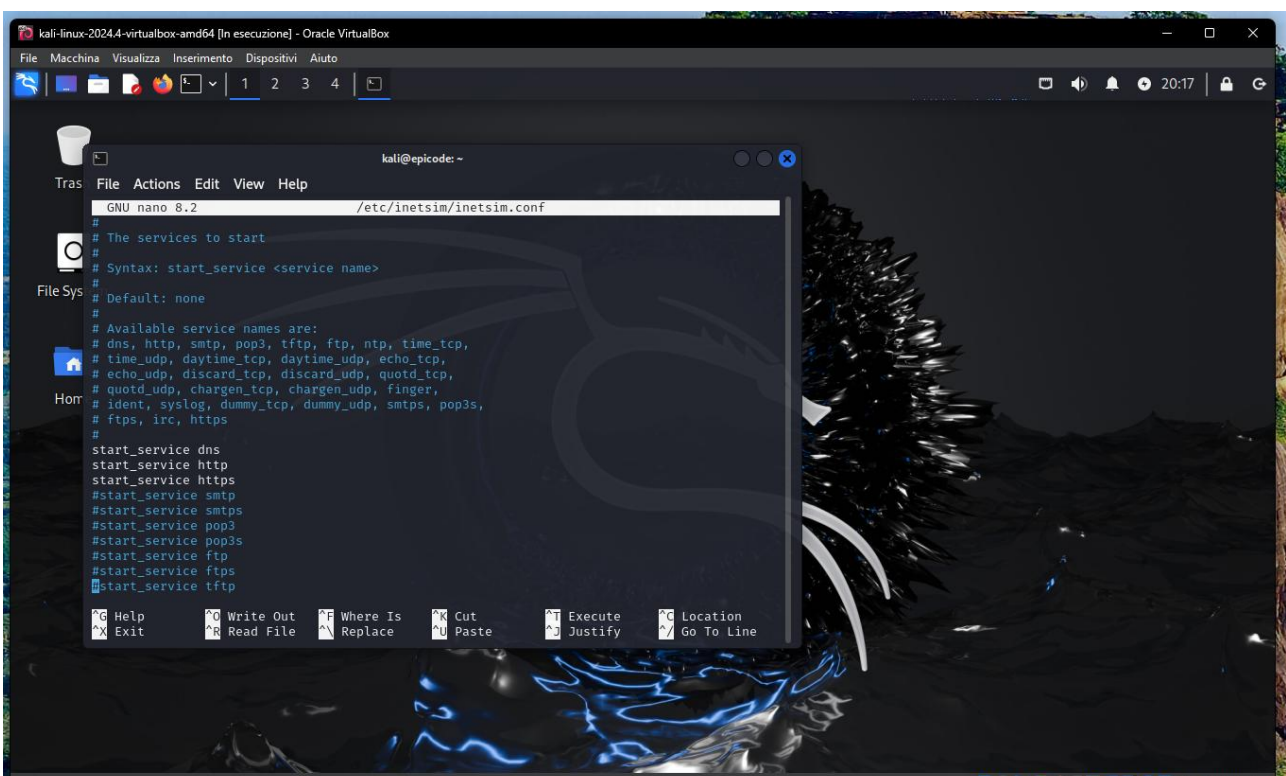
~$ ping 192.168.32.101
PING 192.168.32.101 (192.168.32.101) 56(84) bytes of data:
64 bytes from 192.168.32.101: icmp_seq=1 ttl=128 time=0.362 ms
64 bytes from 192.168.32.101: icmp_seq=2 ttl=128 time=0.294 ms
64 bytes from 192.168.32.101: icmp_seq=3 ttl=128 time=0.392 ms
64 bytes from 192.168.32.101: icmp_seq=4 ttl=128 time=0.574 ms
64 bytes from 192.168.32.101: icmp_seq=5 ttl=128 time=0.277 ms
64 bytes from 192.168.32.101: icmp_seq=6 ttl=128 time=0.281 ms
64 bytes from 192.168.32.101: icmp_seq=7 ttl=128 time=0.310 ms
64 bytes from 192.168.32.101: icmp_seq=8 ttl=128 time=0.383 ms
64 bytes from 192.168.32.101: icmp_seq=9 ttl=128 time=0.292 ms
64 bytes from 192.168.32.101: icmp_seq=10 ttl=128 time=0.448 ms
64 bytes from 192.168.32.101: icmp_seq=11 ttl=128 time=0.296 ms
64 bytes from 192.168.32.101: icmp_seq=12 ttl=128 time=0.290 ms
64 bytes from 192.168.32.101: icmp_seq=13 ttl=128 time=0.335 ms
64 bytes from 192.168.32.101: icmp_seq=14 ttl=128 time=0.339 ms
64 bytes from 192.168.32.101: icmp_seq=15 ttl=128 time=0.430 ms
64 bytes from 192.168.32.101: icmp_seq=16 ttl=128 time=0.294 ms
64 bytes from 192.168.32.101: icmp_seq=17 ttl=128 time=0.380 ms
64 bytes from 192.168.32.101: icmp_seq=18 ttl=128 time=0.285 ms
64 bytes from 192.168.32.101: icmp_seq=19 ttl=128 time=0.354 ms
64 bytes from 192.168.32.101: icmp_seq=20 ttl=128 time=0.350 ms
^C
— 192.168.32.101 ping statistics —
20 packets transmitted, 20 received, 0% packet loss, time 19832ms
rtt min/avg/max/mdev = 0.277/0.348/0.574/0.071 ms

kali@epicode: ~$
```

- Ora che abbiamo appurato che le due macchine comunicano tra di loro, andiamo a configurare e ad attivare il server HTTP e HTTPS ed il servizio DNS tramite il tool InetSim (tool per la simulazione di servizi Internet standard) che abbiamo usato nell'esercizio precedente su Kali.
- Tramite il comando: **`sudo nano /etc/inetsim/inetsim.conf`**, vado a modificare il file .conf del tool a seconda dei servizi che voglio attivare.

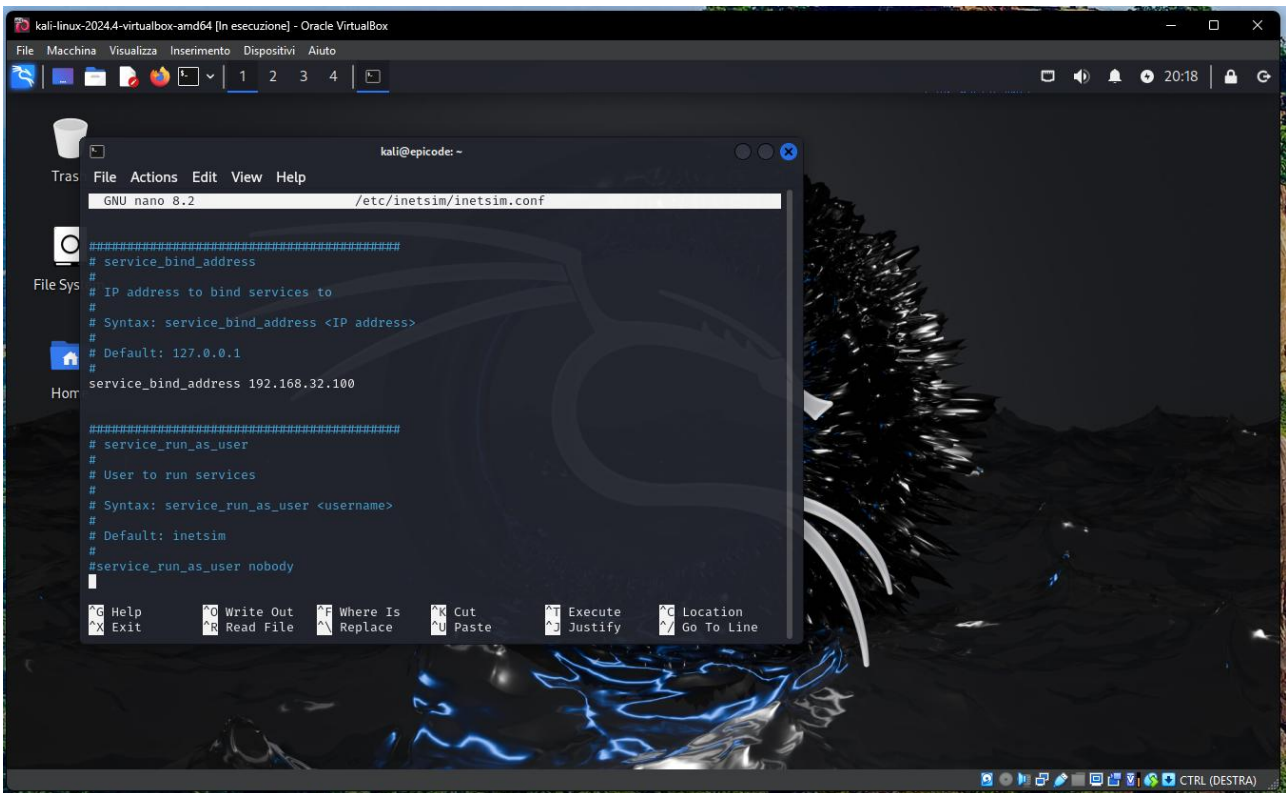


- Attivo i servizi DNS, HTTP e HTTPS tramite la rimozione dell'hashtag.

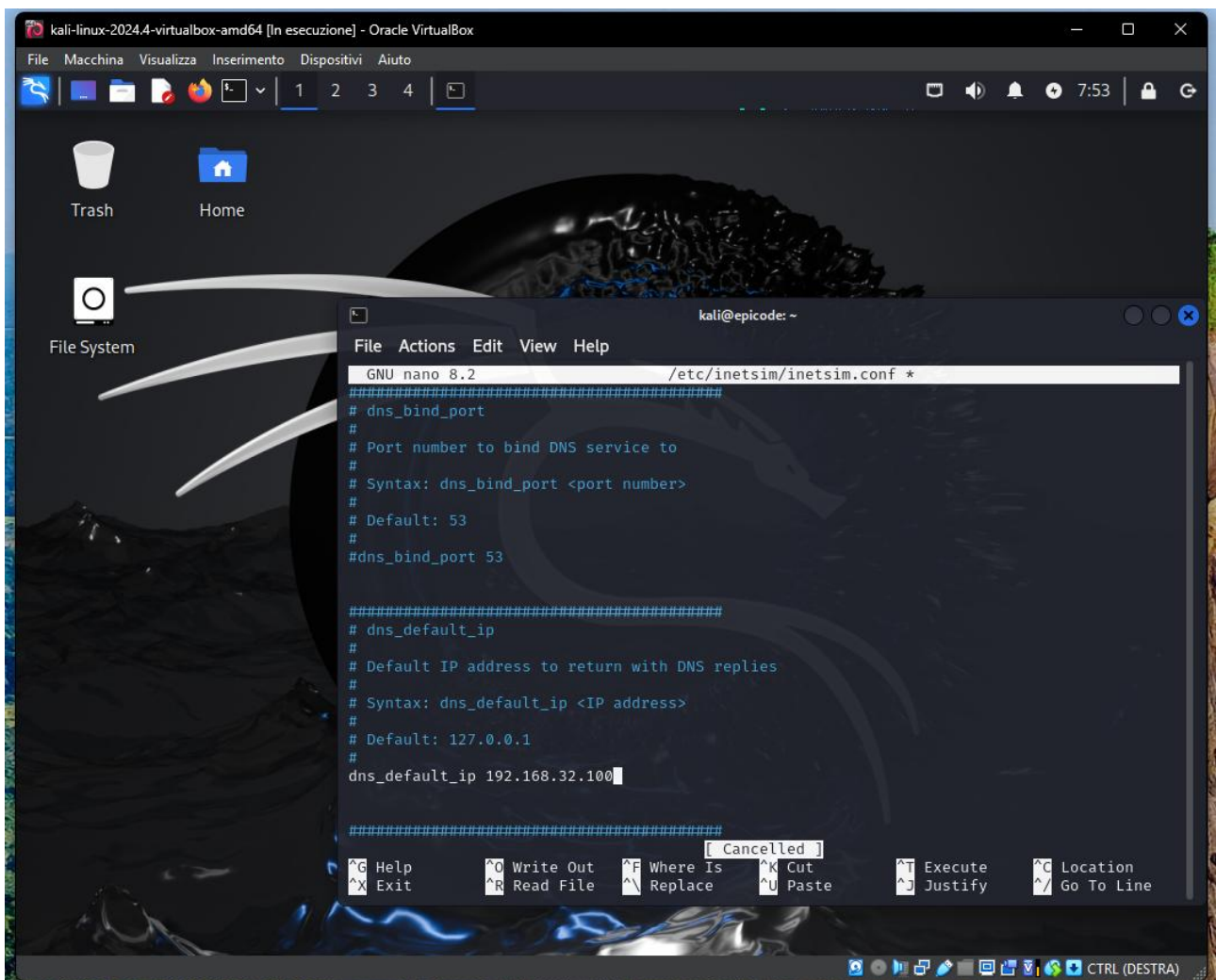




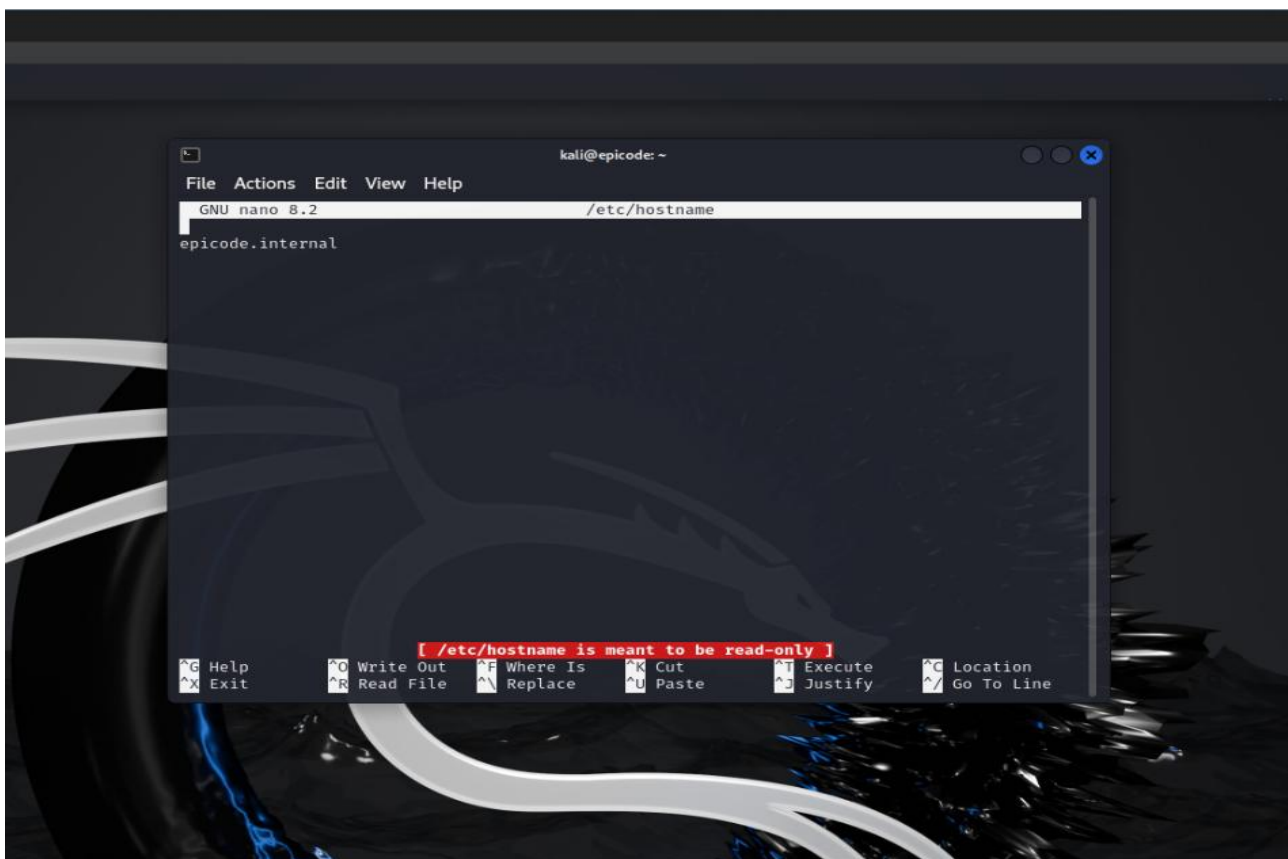
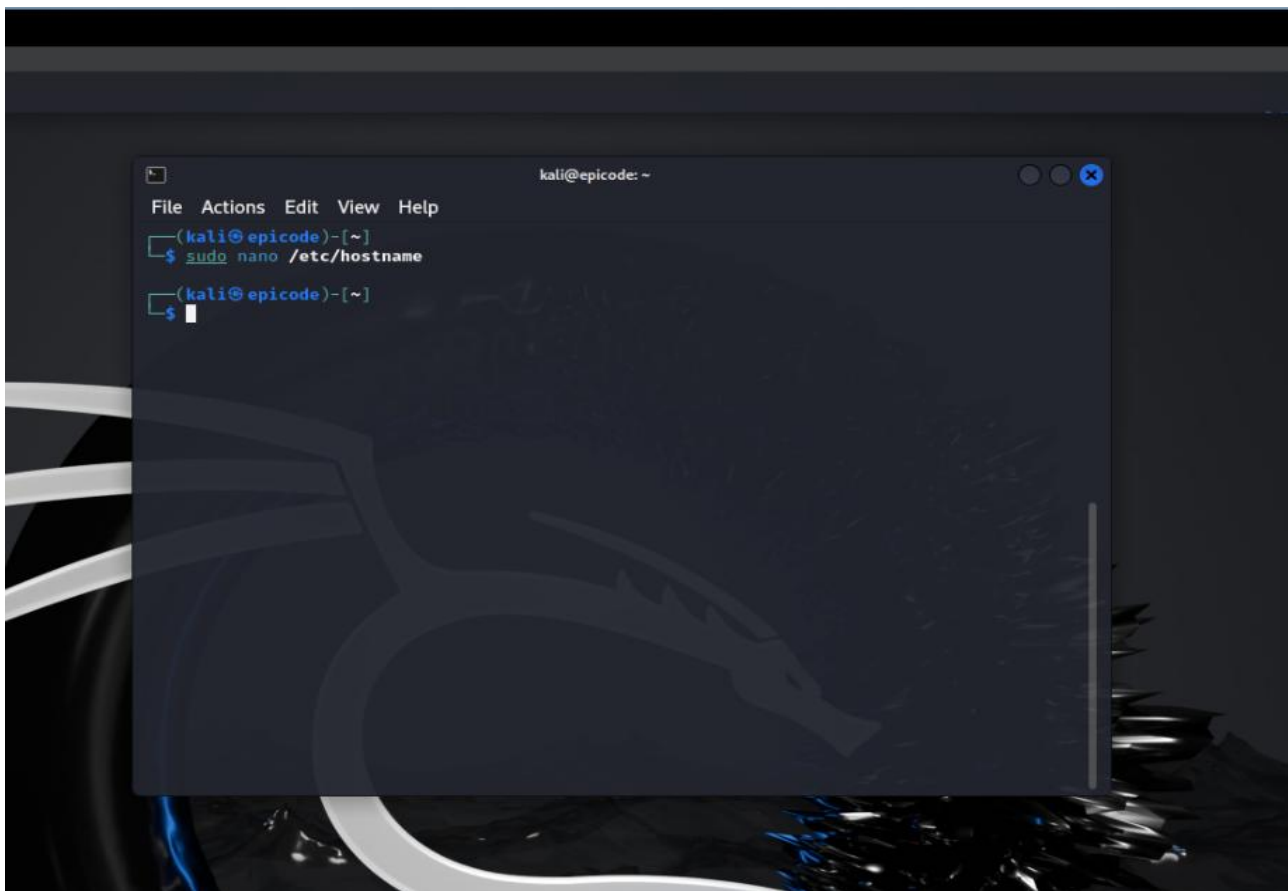
- Attivo la voce ***service\_bind\_address*** e modifico l'IP di default con l'IP della macchina.



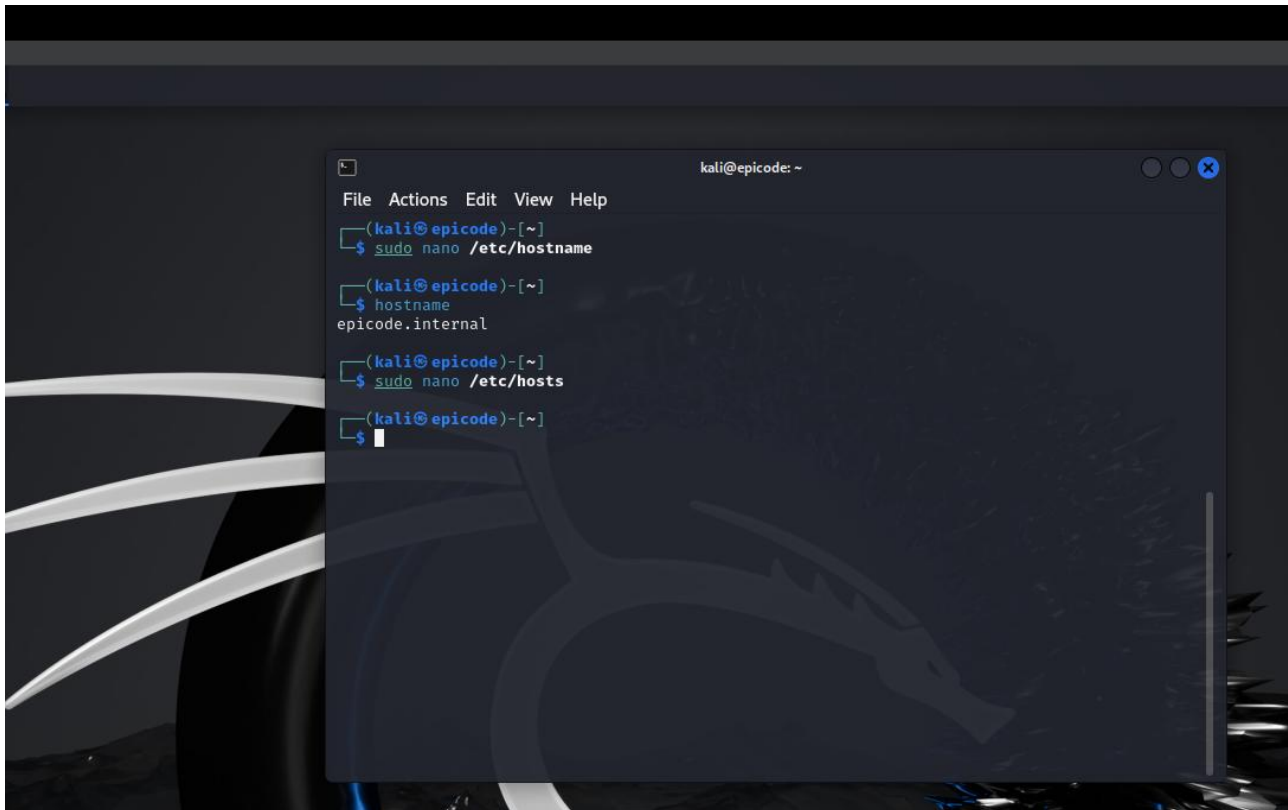
- Imposto il ***dns\_default\_ip*** con il mio IP.



- Tramite il comando: ***sudo nano /etc/hostname***, vado a cambiare l'hostname con ***epicode.internal***.



- Vado ad associare l'hostname all'indirizzo IP della macchina (192.168.32.100) tramite il comando: **sudo nano /etc/hosts**.



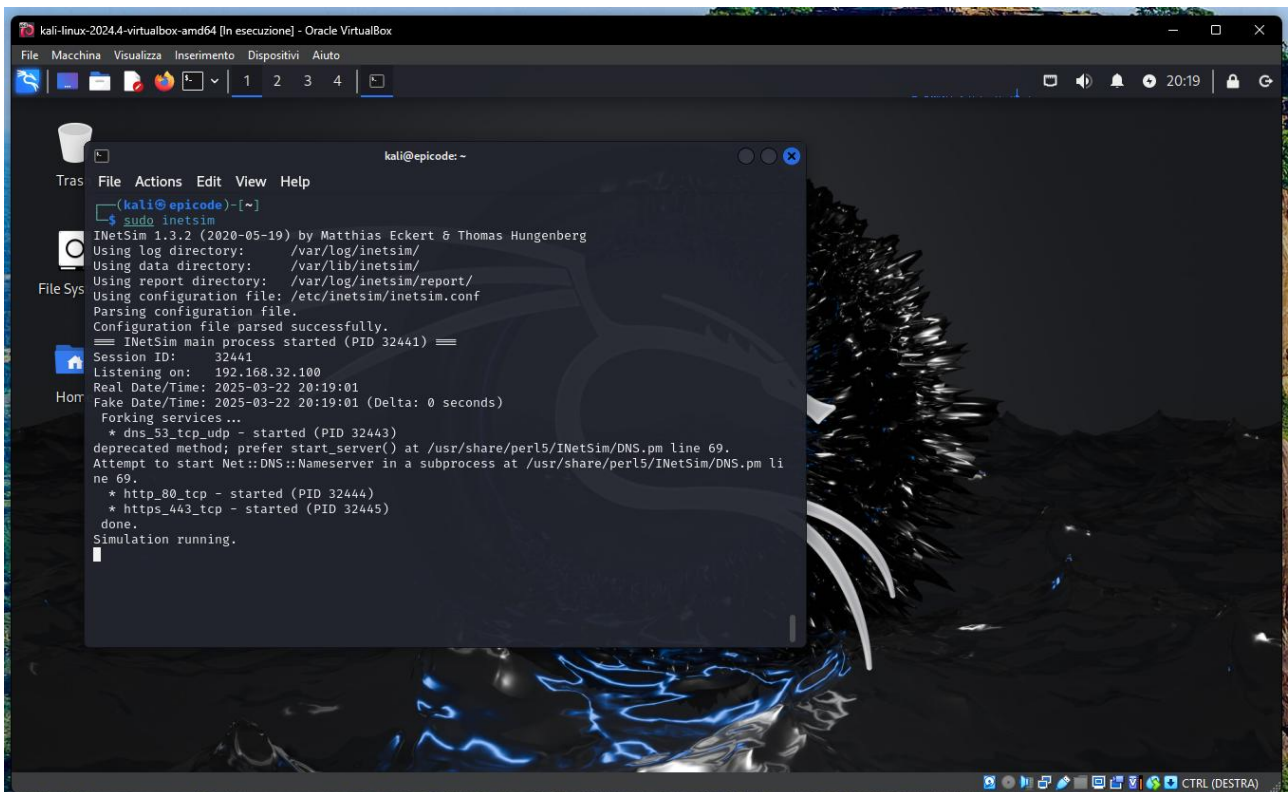
```
kali@epicode: ~  
File Actions Edit View Help  
(kali@epicode)-[~]  
$ sudo nano /etc/hostname  
(kali@epicode)-[~]  
$ hostname  
epicode.internal  
(kali@epicode)-[~]  
$ sudo nano /etc/hosts  
(kali@epicode)-[~]  
$
```



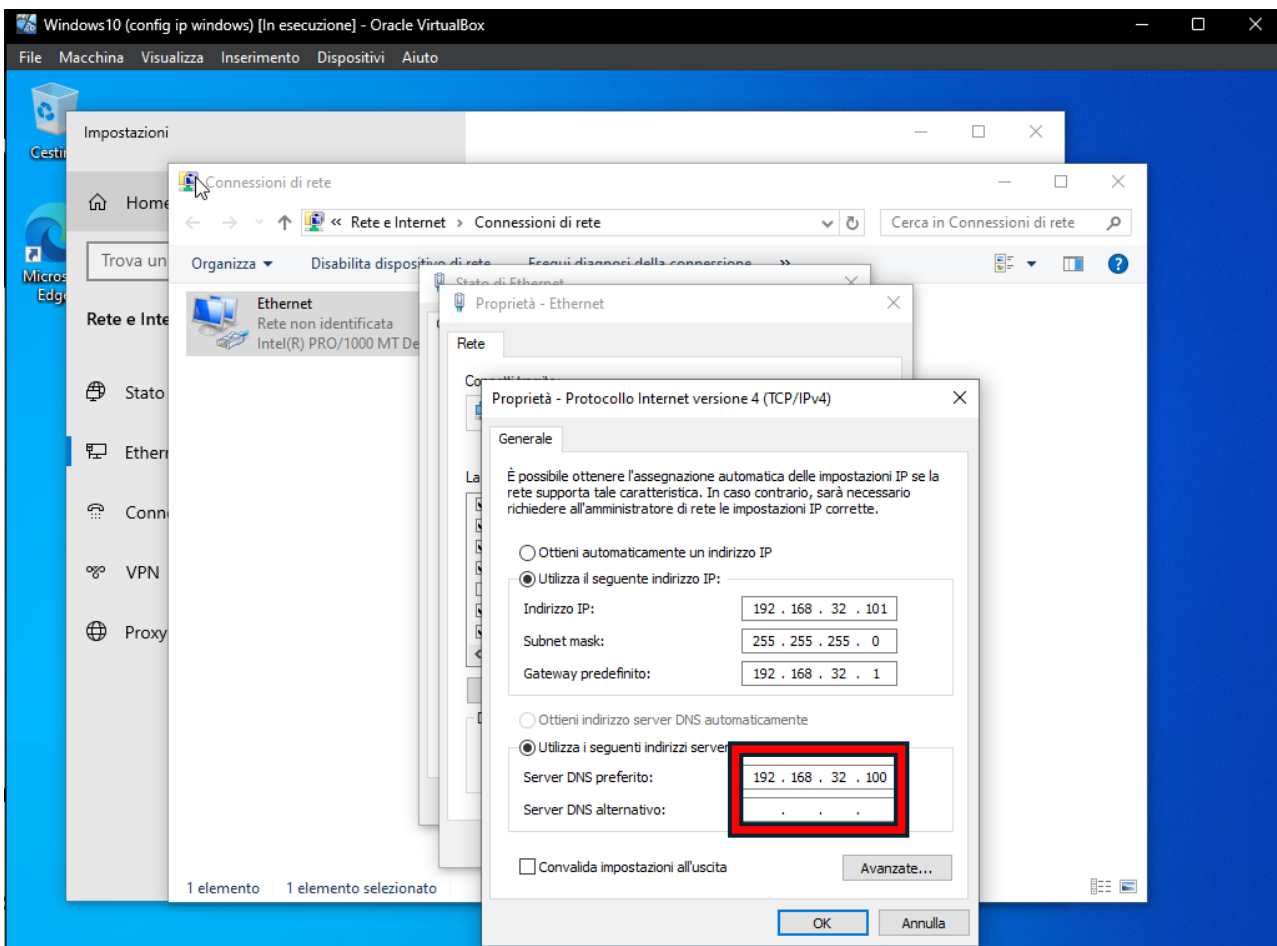
```
kali@epicode: ~  
File Actions Edit View Help  
GNU nano 8.2 /etc/hosts  
127.0.0.1 localhost  
127.0.1.1 kali  
192.168.32.100 epicode.internal  
::1 localhost ip6-localhost ip6-loopback  
ff02::1 ip6-allnodes  
ff02::2 ip6-allrouters  
[ Read 7 lines ]  
^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute ^C Location  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line
```



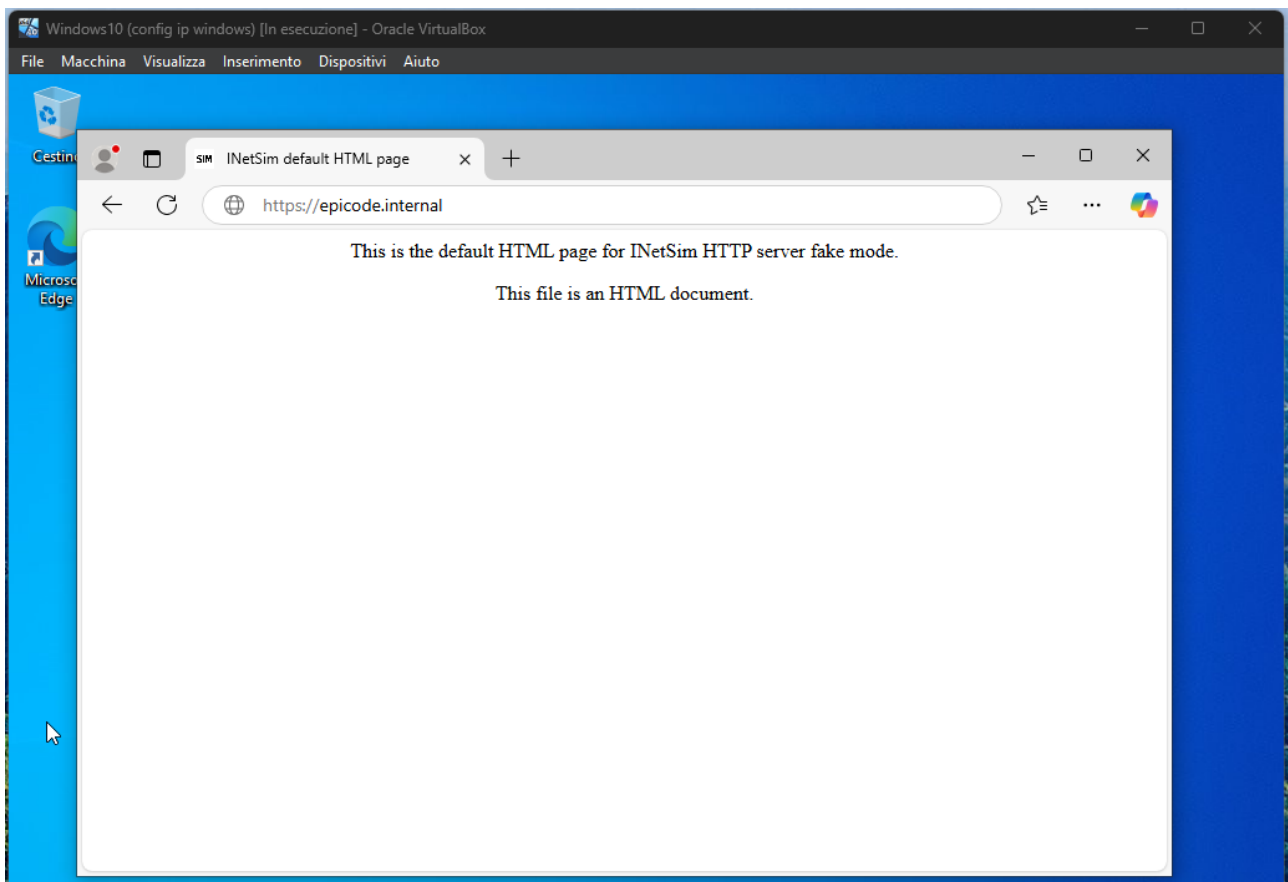
- Ora riavvio la macchina (KALI) e faccio partire il tool InetSim che mi permetterà di simulare i servizi richiesti per l'esercizio che ho configurato in precedenza.



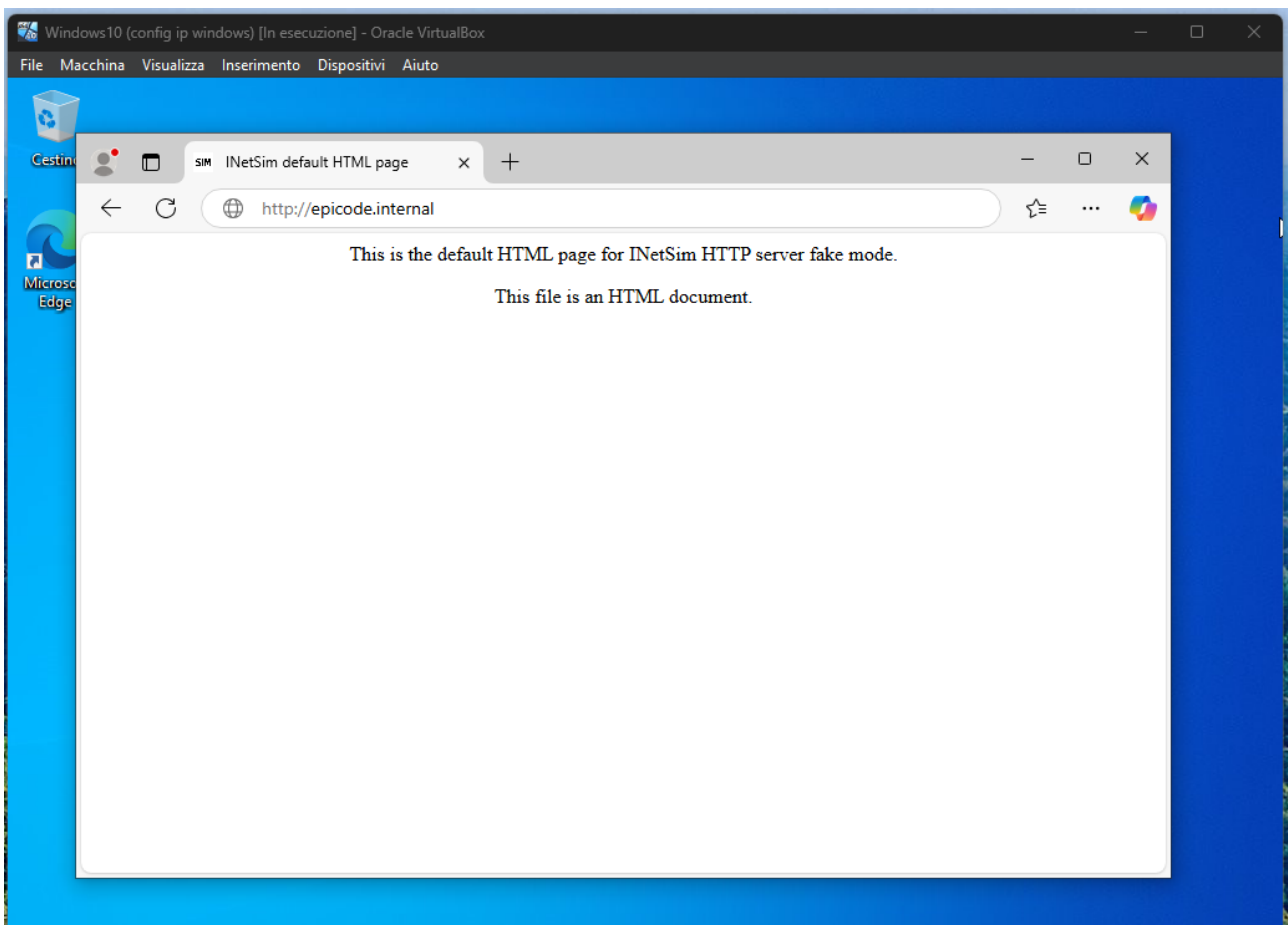
- Mi sposto sulla macchina con Windows dove imposto l'IP del server DNS e procedo con una richiesta HTTP e HTTPS tramite web browser all'hostname: epicode.internal.



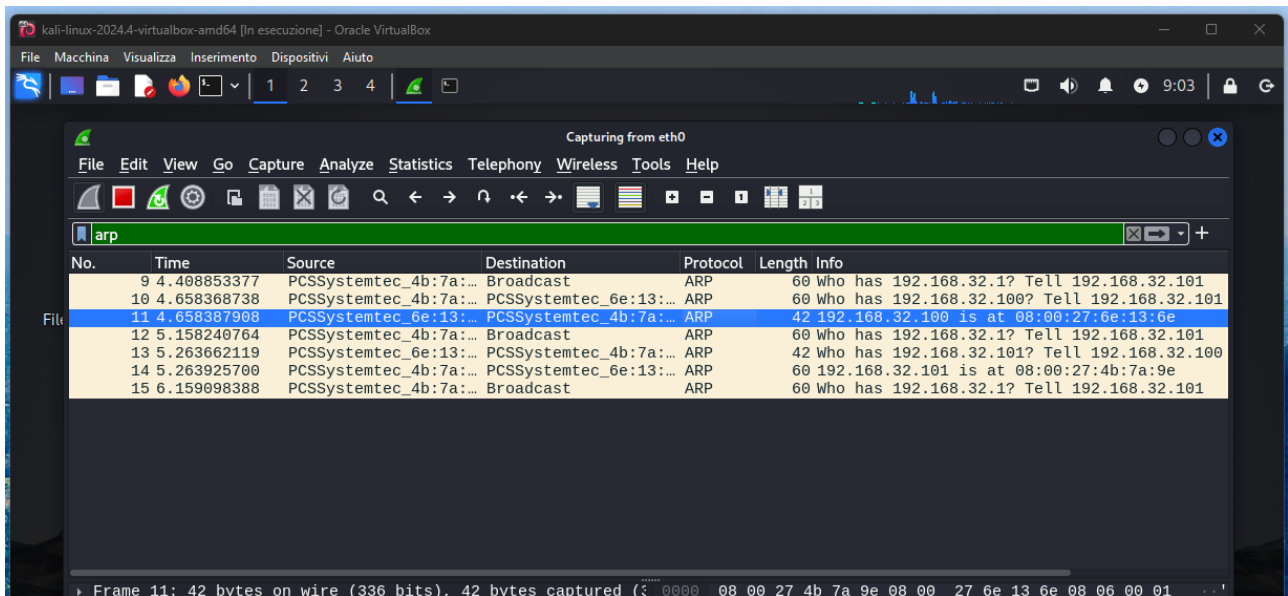
- Richiesta HTTPS.



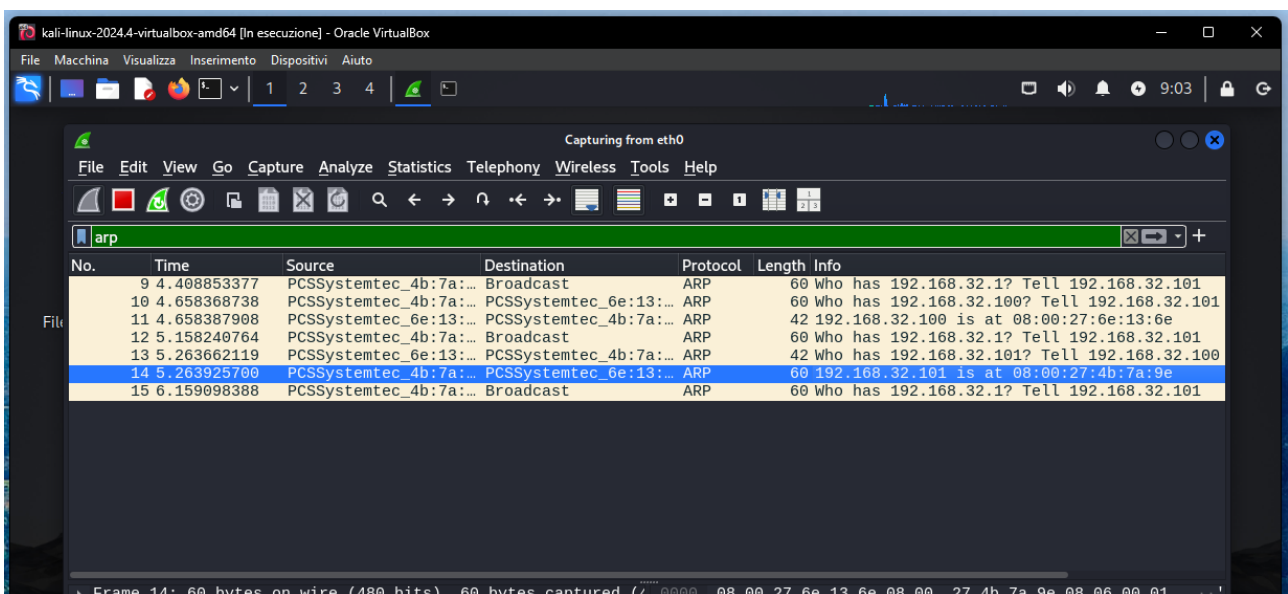
- Richiesta HTTP.



- Intercetto quindi i MAC address delle due macchine con Wireshark.



Come si può vedere dalla prima immagine di Wireshark con il filtro ARP applicato, abbiamo individuato il MAC address a cui è associato l'IP 192.168.32.100 quindi la macchina con KALI LINUX.



Mentre nella seconda immagine abbiamo individuato il MAC address a cui è associato l'IP 192.168.32.101 cioè quello della macchina con WINDOWS.



- Individuati i MAC address delle due macchine, imposto ora il filtro su TCP ed andiamo ad analizzare le differenze tra una richiesta HTTP e HTTPS.

## Intercettazione HTTP

The screenshot shows a Wireshark capture of network traffic on the 'eth0' interface. The filter is set to 'tcp'. The selected packet is a GET request from 192.168.32.101 to 192.168.32.100. The packet details pane shows the following structure:

- Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
- Transmission Control Protocol, Src Port: 80, Dst Port: 58585, Seq: 151, Ack: 477, Window: 65535, Len: 408, Ecn: 0, Flags: [RST, ACK, FIN]
- Hypertext Transfer Protocol
- HTTP/1.1 200 OK
- Server: InetSim HTTP Server
- Date: Sun, 23 Mar 2025 00:24:57 GMT
- Connection: close
- Content-Type: text/html
- Content-Length: 258
- Cache-Control: no-cache
- ETag: "192.168.32.101"
- File Data: 258 bytes
- Line-based text data: text/html (10 lines)
- <html>
- <head>
- <title>InetSim default HTML page</title>
- <body>
- <p>
- <p align="center">This is the default HTML page for InetSim HTTP server f...
- <p align="center">This file is an HTML document.</p>
- </body>
- </html>

The packet bytes pane shows the raw data of the packet, and the packet list pane shows the packet details.

## Intercettazione HTTPS

The screenshot shows a Wireshark capture of network traffic on the 'eth0' interface. The filter is set to 'tcp'. The selected packet is a GET request from 192.168.32.101 to 192.168.32.100. The packet details pane shows the following structure:

- Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
- Transmission Control Protocol, Src Port: 80, Dst Port: 58585, Seq: 151, Ack: 477, Window: 65535, Len: 408, Ecn: 0, Flags: [RST, ACK, FIN]
- Hypertext Transfer Protocol
- HTTP/1.1 200 OK
- Server: InetSim HTTP Server
- Date: Sun, 23 Mar 2025 00:24:57 GMT
- Connection: close
- Content-Type: text/html
- Content-Length: 258
- Cache-Control: no-cache
- ETag: "192.168.32.101"
- File Data: 258 bytes
- Line-based text data: text/html (10 lines)
- <html>
- <head>
- <title>InetSim default HTML page</title>
- <body>
- <p>
- <p align="center">This is the default HTML page for InetSim HTTP server f...
- <p align="center">This file is an HTML document.</p>
- </body>
- </html>

The packet bytes pane shows the raw data of the packet, and the packet list pane shows the packet details.



- Come possiamo vedere in entrambe le intercettazioni viene effettuata una tripla stretta di mano come si evince dalle sigle SYN, SYN-ACK, ACK poiché entrambi i protocolli HTTP e HTTPS (livello applicazione della pila ISO/OSI) hanno bisogno del protocollo TCP (livello trasporto della pila ISO/OSI) per il loro funzionamento.
- Vediamo che una volta effettuata la connessione, nell'immagine dell'intercettazione HTTP il client effettua una richiesta GET, a cui il server risponde con il codice 200, ciò significa che la richiesta è andata a buon fine, inoltre possiamo vedere chiaramente quello che viene trasmesso al cliente poiché essendo una richiesta HTTP le informazioni non sono cifrate.
- Mentre nella intercettazione HTTPS vediamo che viene usato il protocollo TLS, che è uno dei protocolli che viene usato per le richieste HTTPS e che effettua una cifratura delle informazioni, perciò come vediamo dall'immagine non riusciamo a vedere le informazioni in chiaro che vengono restituite al client come nell'intercettazione HTTP.





