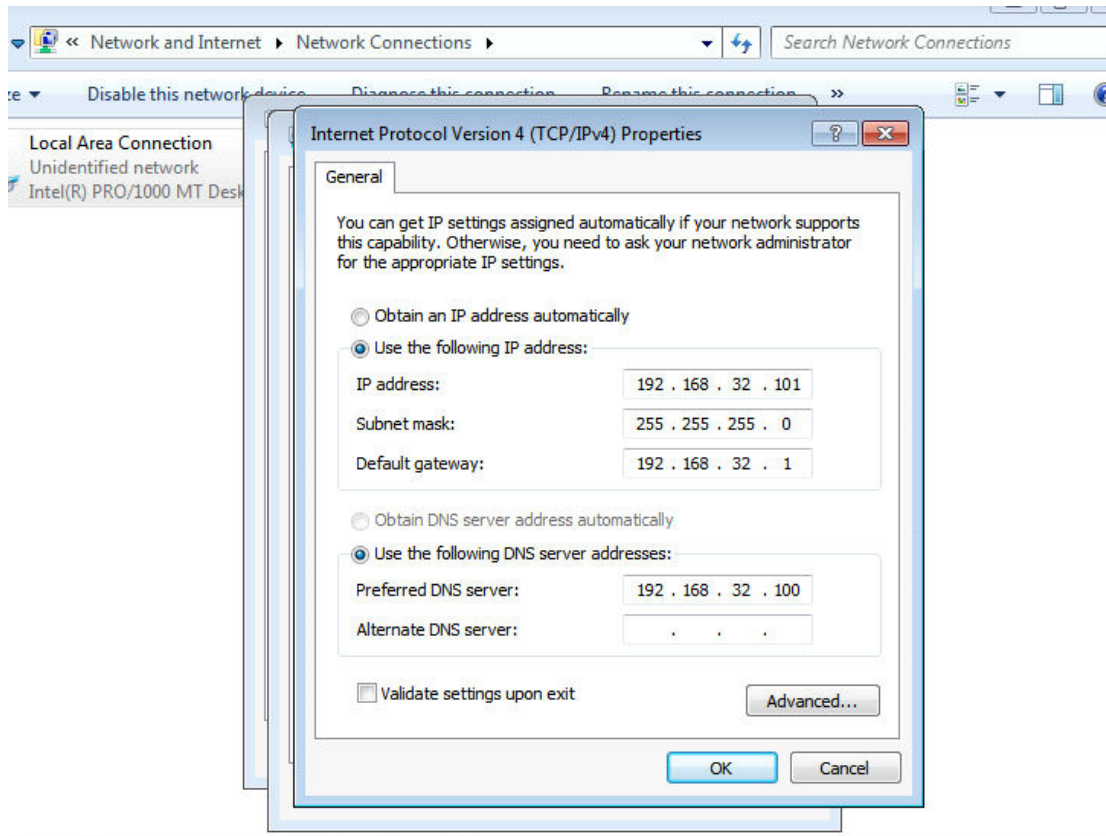


# W4D4 Esercitazione

Di Florin Eugen Peticaru

L'esercitazione di oggi richiedeva di simulare nell'ambiente virtuale una comunicazione tra un client con indirizzo IP 192.168.32.101, nel nostro caso Windows 7, che richiede tramite web browser una risorsa all'hostname *Epicode.internal* che risponde all'indirizzo 192.168.32.100, nel nostro caso Kali Linux che fungerà da server HTTP e HTTPS.

Iniziamo configurando indirizzo ip e indirizzo DNS sulla macchina di Windows 7 come segue.



in seguito andiamo a configurare l'indirizzo IP anche su Kali immettendo come indirizzo IP 192.168.32.100

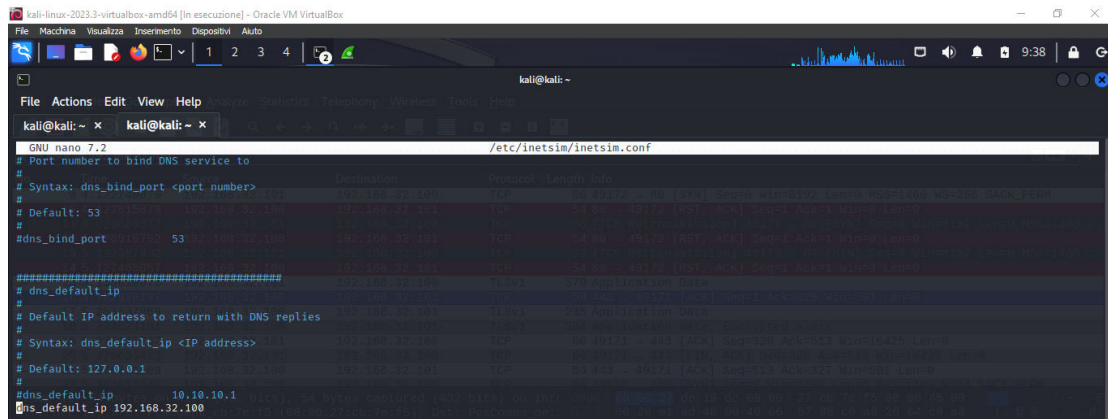
```
kali@kali: ~  
File Actions Edit View Help  
$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.32.100 netmask 255.255.255.0 broadcast 192.168.32.255  
    inet6 fe80::4f30:7b6f:3247:652e prefixlen 64 scopeid 0<link>  
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)  
    RX packets 172 bytes 15601 (15.2 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 115 bytes 19984 (19.5 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 24 bytes 1240 (1.2 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 24 bytes 1240 (1.2 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
"the quieter you become, the more you are able to hear"  
(kali@kali)~  
$
```

Quindi procediamo configurando il tool di InetSim che ci permetterà di emulare la nostra macchina come server (inizialmente come

```
kali@kali: ~  
File Actions Edit View Help  
$ nano /etc/inetSim/inetSim.conf  
GNU nano 7.2 /etc/inetSim/inetSim.conf  
start_service time_udp  
start_service daytime_tcp  
start_service daytime_udp  
start_service echo_tcp  
start_service echo_udp  
start_service discard_tcp  
start_service discard_udp  
start_service quotd_tcp  
start_service quotd_udp  
start_service chargen_tcp  
start_service chargen_udp  
start_service dummy_tcp  
start_service dummy_udp  
#####  
# service_bind_address  
#  
# IP address to bind services to  
# Syntax: service_bind_address <IP address>  
#  
# Default: 127.0.0.1  
service_bind_address 192.168.32.100
```

HTTPS)

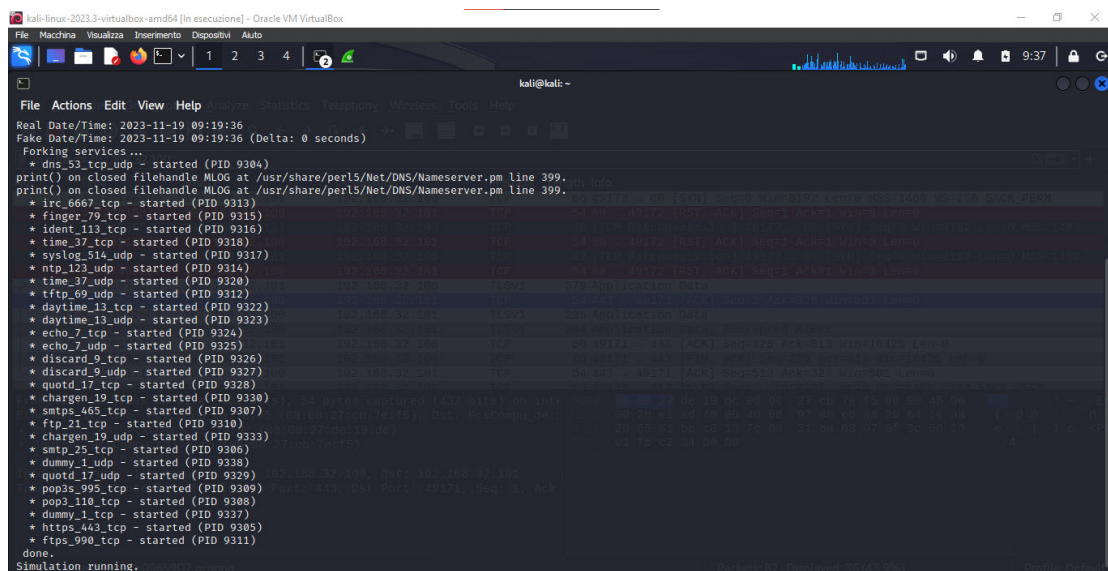
Attiviamo quindi il servizio di bind\_address, che serve a dire alle macchine che tentano di accedere ai servizi a quale indirizzo ip rivolgersi, dando l'indirizzo ip della nostra macchina.



```
kali@kali: ~  
# GNU nano 7.2 /etc/inetsim/inetsim.conf  
#  
# Port number to bind DNS service to  
#  
# Syntax: dns_bind_port <port number>  
#  
# Default: 53  
#  
# dns_bind_port 53  
#####  
# dns_default_ip  
#  
# Default IP address to return with DNS replies  
#  
# Syntax: dns_default_ip <IP address>  
#  
# Default: 127.0.0.1  
#  
# dns_default_ip 10.10.10.1  
# dns_default_ip 192.168.32.100
```

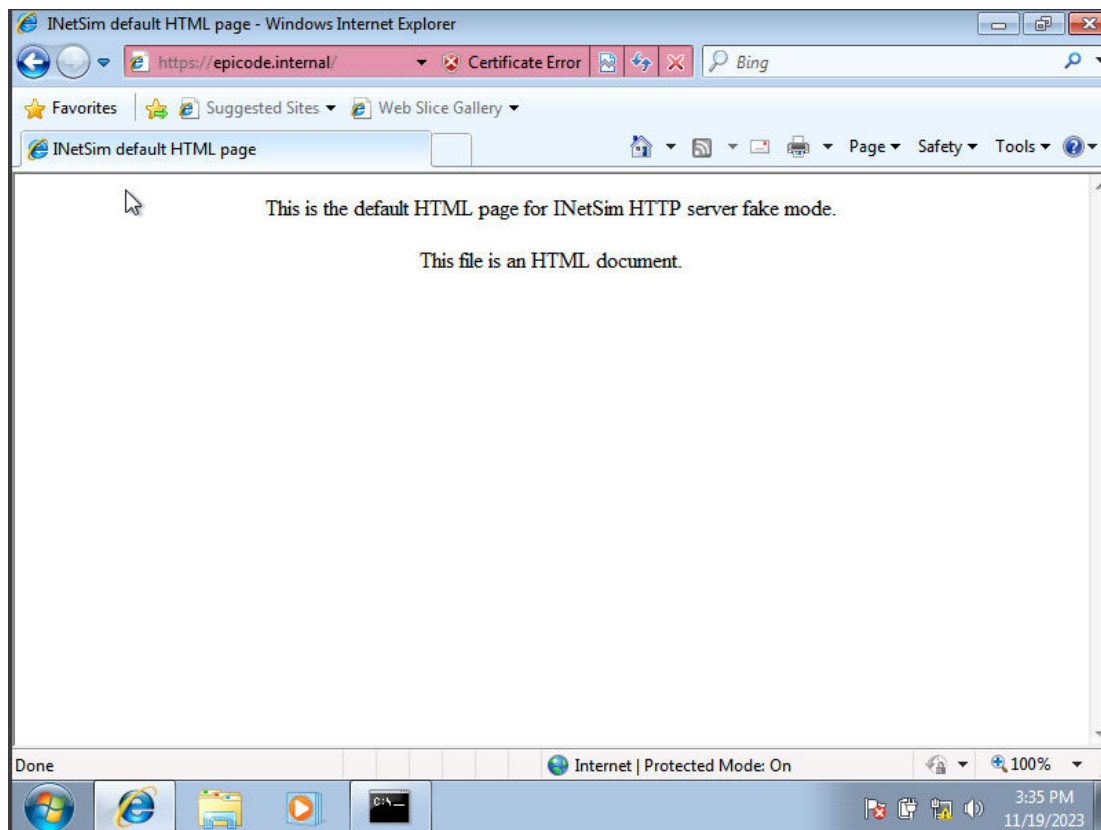
In seguito abilitiamo il servizio di DNS con la stringa `dns_default_ip` dandogli sempre l'indirizzo della nostra macchina.

Facciamo quindi partire la simulazione di HTTPS attivando InetSim con il comando **sudo inetsim**

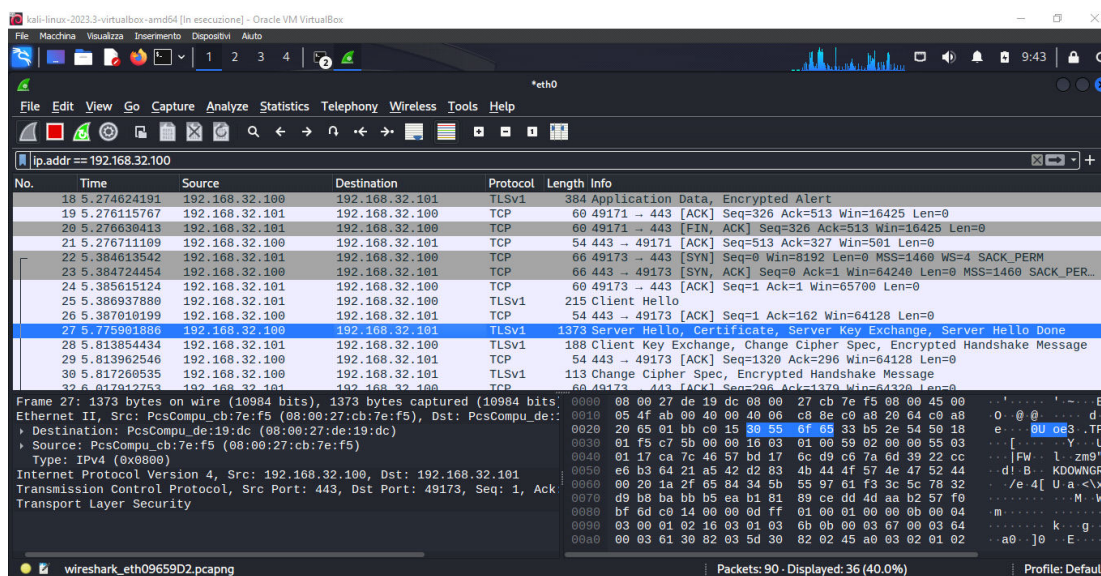


```
kali@kali: ~  
File Actions Edit View Help  
Real Date/Time: 2023-11-19 09:19:36  
Fake Date/Time: 2023-11-19 09:19:36 (delta: 0 seconds)  
Forking services...  
* dns_53_tcp_udp - started (PID 9304)  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm line 399.  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm line 399.  
* irc_6667_tcp - started (PID 9313)  
* finger_79_tcp - started (PID 9315)  
* ident_113_tcp - started (PID 9316)  
* time_37_tcp - started (PID 9318)  
* syslog_514_udp - started (PID 9317)  
* ntp_123_udp - started (PID 9314)  
* time_37_udp - started (PID 9320)  
* tftp_69_udp - started (PID 9322)  
* daytime_13_tcp - started (PID 9322)  
* daytime_13_udp - started (PID 9323)  
* echo_7_tcp - started (PID 9324)  
* echo_7_udp - started (PID 9325)  
* discard_9_tcp - started (PID 9326)  
* discard_9_udp - started (PID 9327)  
* quotd_17_tcp - started (PID 9328)  
* chargen_19_tcp - started (PID 9330)  
* smtps_465_tcp - started (PID 9307)  
* ftp_21_tcp - started (PID 9310)  
* chargen_19_udp - started (PID 9333)  
* smtp_25_tcp - started (PID 9306)  
* dummy_1_udp - started (PID 9338)  
* quotd_17_udp - started (PID 9329)  
* pop3s_995_tcp - started (PID 9309)  
* pop3_110_tcp - started (PID 9308)  
* dummy_1_tcp - started (PID 9337)  
* https_443_tcp - started (PID 9305)  
* ftps_990_tcp - started (PID 9311)  
done.  
Simulation running.
```

Come possiamo notare il servizio di HTTPS è attivo e la simulazione è in esecuzione, quindi possiamo tornare alla macchina di Windows 7 per fare la prova di accesso alla pagina *epicode.internal*



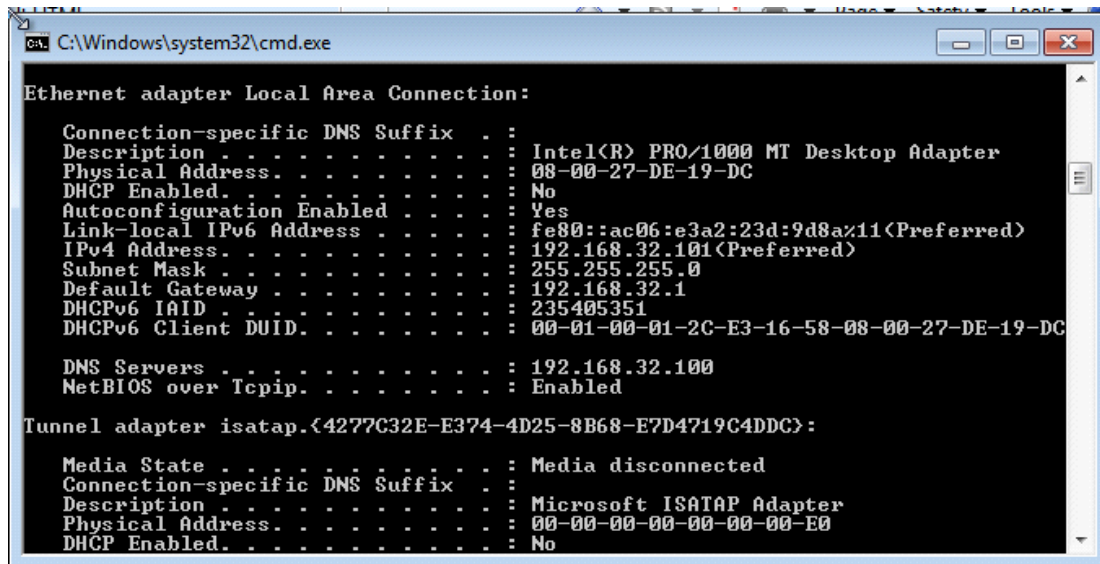
Vediamo quindi che il nostro client riesce a raggiungere la pagina correttamente, ora come richiesto dalla consegna eseguiamo una cattura dei pacchetti tramite **Wireshark** da Kali



Notiamo che i pacchetti viaggiano in maniera regolare e anche cifrata tramite il protocollo TLSv1 e possiamo anche notare i due indirizzi MAC delle nostre macchine (*visionabili nel riquadro in basso a sinistra, source è la macchina di Kali e*



*destination* è la macchina di Windows) che ci fa capire che fisicamente sono le macchine corrette ad essere connesse.



```
C:\Windows\system32\cmd.exe

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    Description . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
    Physical Address. . . . . : 08-00-27-DE-19-DC
    DHCP Enabled. . . . . : No
    Autoconfiguration Enabled . . . . : Yes
    Link-local IPv6 Address . . . . . : fe80::ac06:e3a2:23d:9d8a%11(Preferred)
    IPv4 Address. . . . . : 192.168.32.101(Preferred)
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.32.1
    DHCPv6 IAID . . . . . : 235405351
    DHCPv6 Client DUID. . . . . : 00-01-00-01-2C-E3-16-58-08-00-27-DE-19-DC

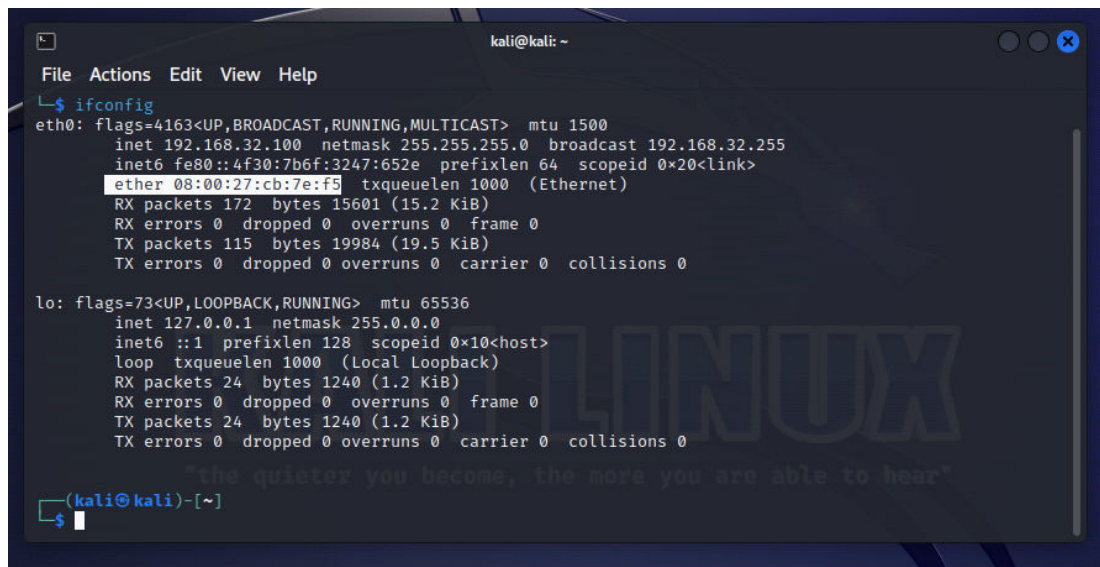
    DNS Servers . . . . . : 192.168.32.100
    NetBIOS over Tcpip. . . . . : Enabled

Tunnel adapter isatap.{4277C32E-E374-4D25-8B68-E7D4719C4DDC}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 
    Description . . . . . : Microsoft ISATAP Adapter
    Physical Address. . . . . : 00-00-00-00-00-00-E0
    DHCP Enabled. . . . . : No
```

Qui possiamo vedere l'indirizzo MAC della macchina alla riga 5

E di seguito vediamo l'indirizzo MAC di Kali alla voce *ether* (evidenziato nell'immagine)



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

└─$ 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::4f30:7b6f:3247:652e prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)
    RX packets 172 bytes 15601 (15.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 115 bytes 19984 (19.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

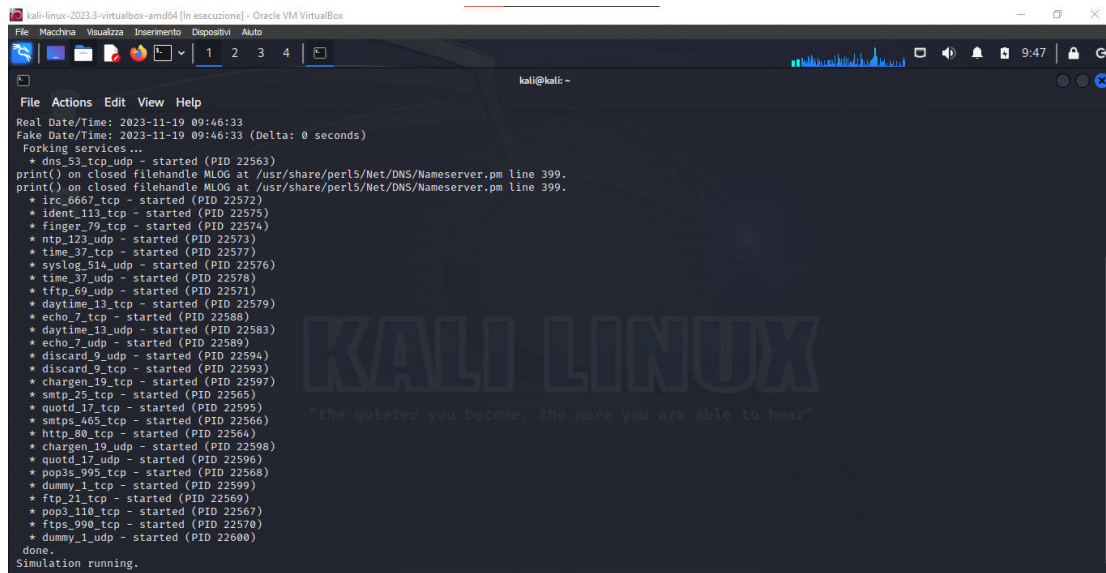
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 24 bytes 1240 (1.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 24 bytes 1240 (1.2 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

"the quieter you become, the more you are able to hear"
(kali@kali)-[~]
└─$
```

Possiamo quindi essere sicuri che le due macchine che stanno interagendo sono proprio quelle che abbiamo configurato.

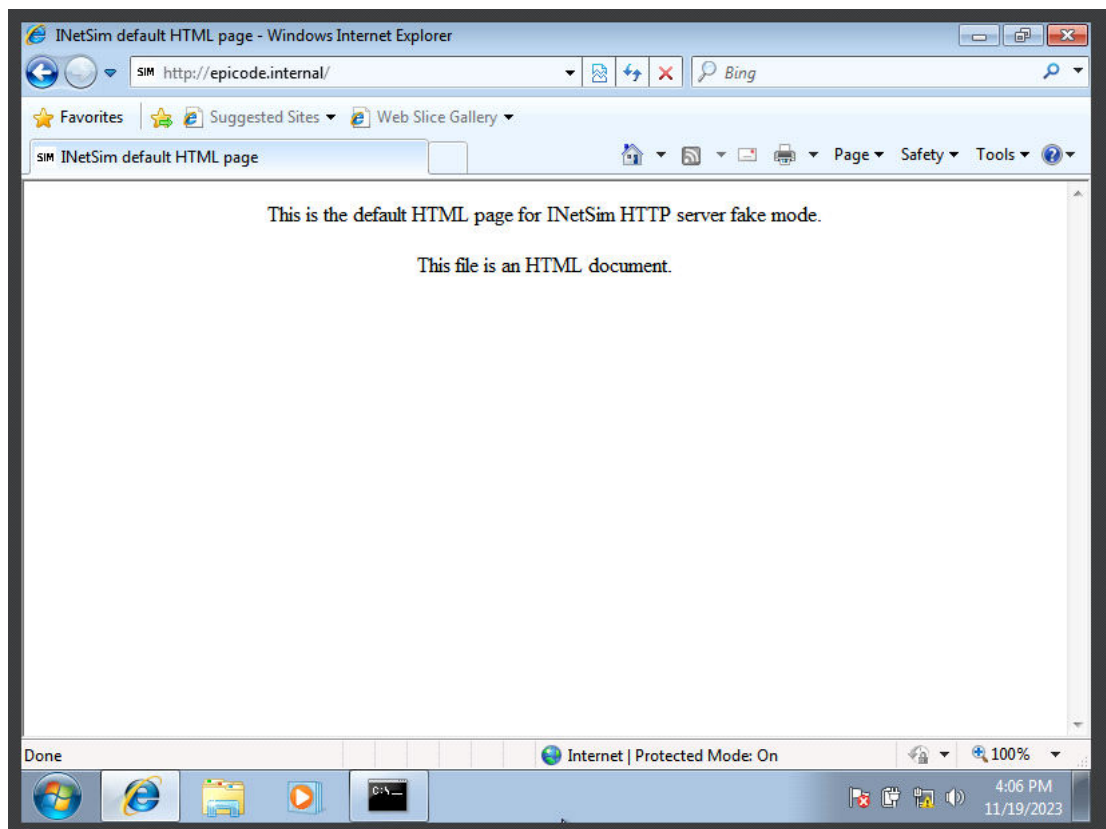
Passiamo ora all'ultimo punto della consegna che ci richiedeva di eseguire di nuovo i servizi di InetSim ma stavolta con la differenza che il servizio attivo sarà HTTP e non HTTPS, riconfiguriamo quindi di nuovo il file di InetSim, modifichiamo

questo aspetto e riavviamo la simulazione.



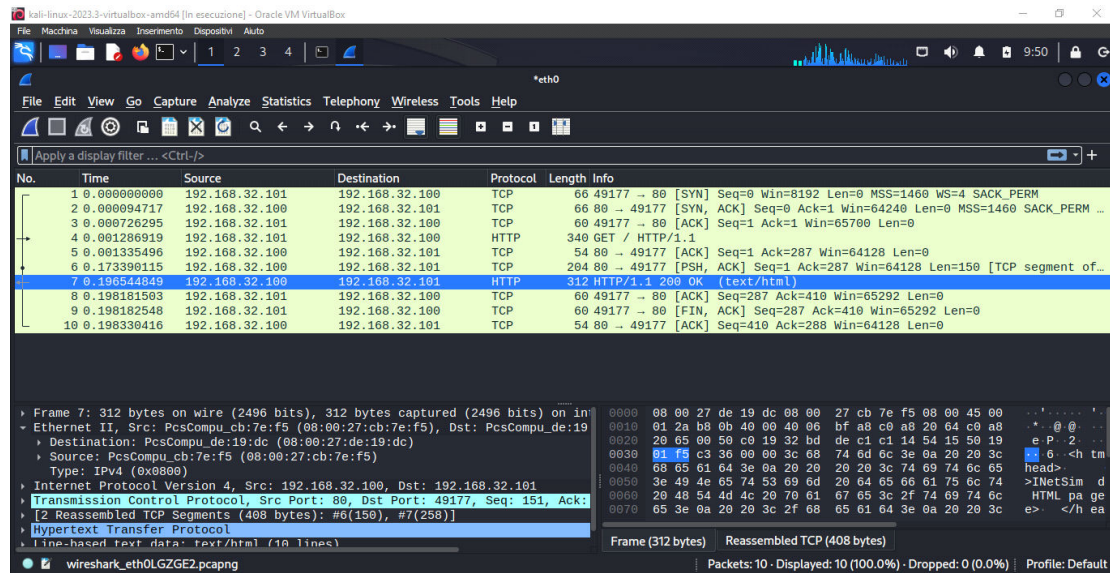
```
kali@kali:~$ nmap -sS -p 80 192.168.1.100
Nmap scan report for 192.168.1.100
Host is up (0.0000s latency).
Not shown: 65534 closed ports
PORT      STATE SERVICE
80/tcp    open  HTTP/1.1
```

Vediamo che il servizio di HTTPS non è attivo ma al suo posto si è attivato quello di HTTP, ora tentiamo di accedere di nuovo all'indirizzo `epicode.internal` dal client



Il nostro client riesce ad accedere alla pagina e vediamo anche che nella barra di ricerca l'indirizzo è cambiato da **HTTPS://** a **HTTP://** per notare le differenze a

livello di pacchetti riapriamo Wireshark ed effettuiamo nuovamente una cattura dei pacchetti



Possiamo notare di nuovo che gli indirizzi MAC sono quelli delle nostre macchine ma come differenze notiamo che ci sono meno pacchetti ma soprattutto che non passano in maniera sicura e cifrata come precedentemente a causa dell'assenza del protocollo TLSv1 che si occupava proprio di questo.