W4 - D5 - PRATICA - 18 07 205 - Giuseppe Gigliotti- PROGETTO M1 - INTRODUZIONE ALL'HACKING

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.

Per semplicità usiamo direttamente gli indirizzi già utilizzati durante le configurazioni iniziali. Su Windows l'indirizzo sarà 192.168.50.102, mentre per kali l'indirizzo sarà 192.168.50.100

- Verifichiamo che la nostra macchina kali sia connessa all'indirizzo IP richiesto

```
┌──(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d1:f8:5d brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.100/24 brd 192.168.50.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::416:9a41:8519:128/64 scope link noprefixroute
       valid_lft forever preferred_lft forever

┌──(kali㉿kali)-[~]
└─$ sudo nano /etc/inetsim/inetsim.conf
```

Fatto ciò utilizzando in comando apposito ed utilizzando un editor di testo , passiamo sul file di configurazione di inetsim

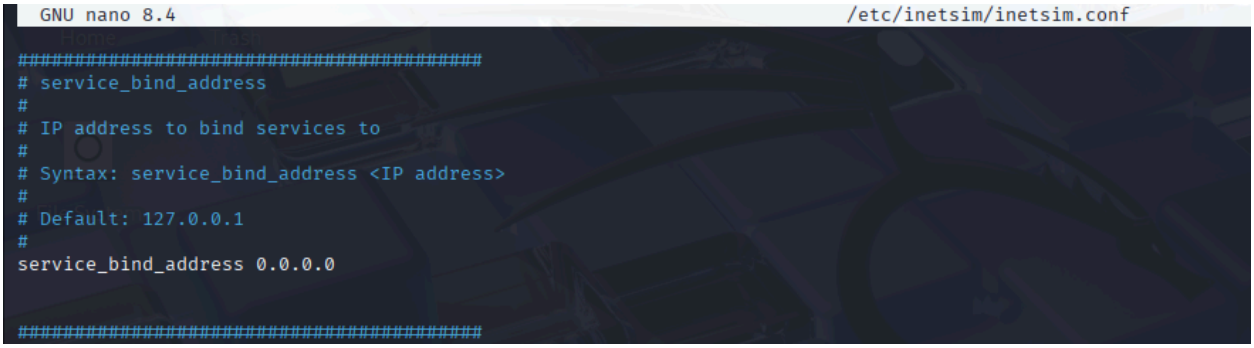
```
  GNU nano 8.4                                    /etc/inetsim/inetsim.conf *
# INetSim configuration file
#
##########################################################


##########################################################
# Main configuration
##########################################################

########################################
# start_service
#
# The services to start
#
# Syntax: start_service <service name>
#
# Default: none
#
# Available service names are:
# dns, http, smtp, pop3, tftp, ftp, ntp, time_tcp,
# time_udp, daytime_tcp, daytime_udp, echo_tcp,
# echo_udp, discard_tcp, discard_udp, quotd_tcp,
# quotd_udp, chargen_tcp, chargen_udp, finger,
# ident, syslog, dummy_tcp, dummy_udp, smtps, pop3s,
# ftps, irc, https
#
start_service dns
#start_service http
start_service https
#start_service smtp
```

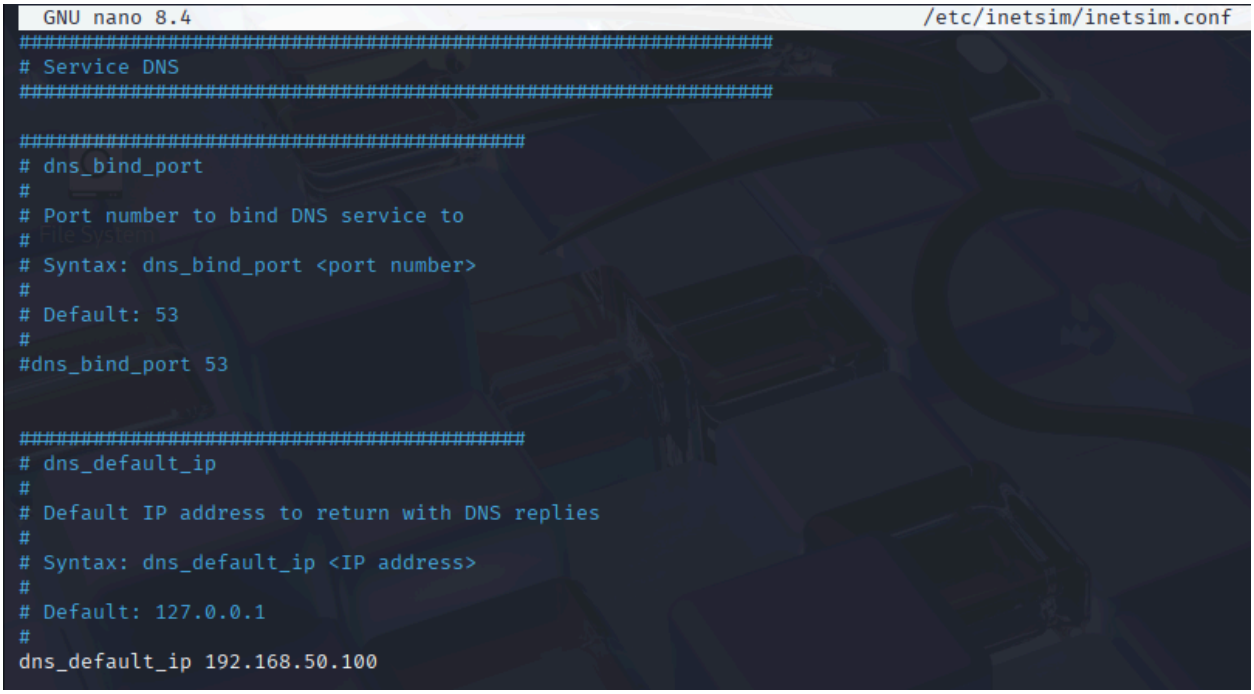
Nel nostro primo caso avremo bisogno di abilitare i servizi https e del dns.

Questa configurazione ci dice che, inetsim esporrà i servizi sull'IP che diremo noi (o precisamente su tutto) , in questo caso l'indirizzo visualizzato tramite il comando ip a 192.168.50.100.

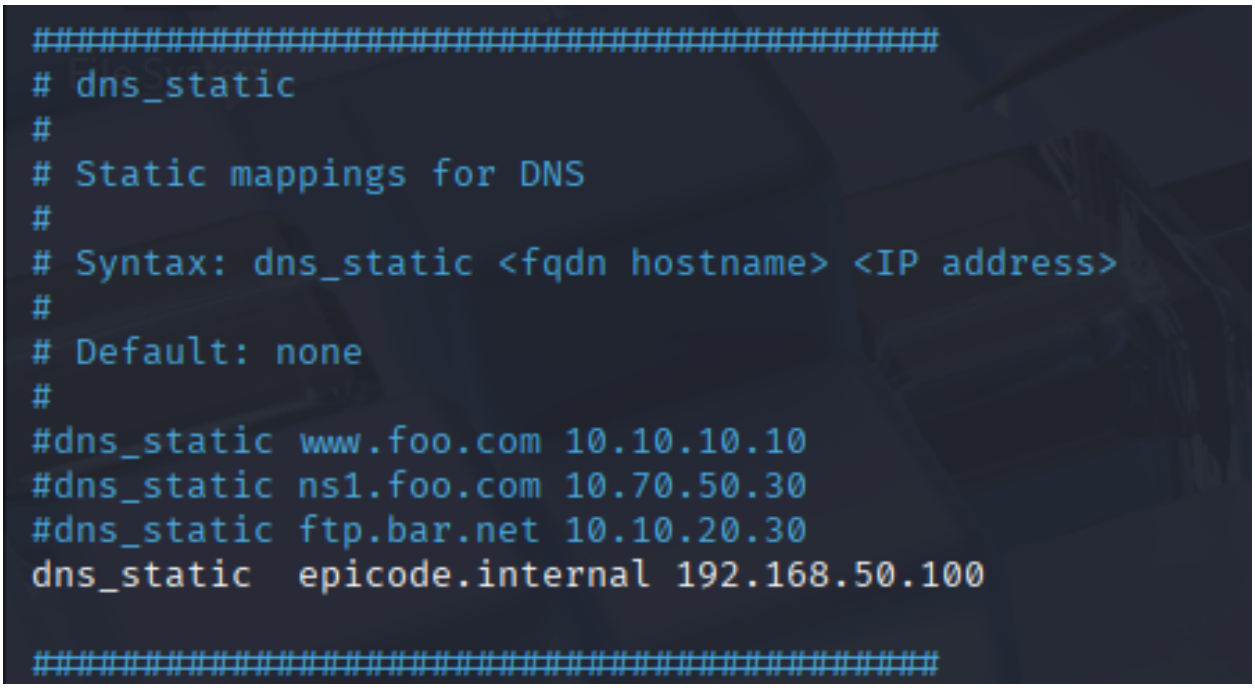


Configuriamo il dns

- Modifichiamo l'IP 10.10.10.1 con il nostro IP da utilizzare



- Scendiamo in basso verso i record dns statici e aggiungiamo il nostro come richiesto nella traccia (epicode.internal)



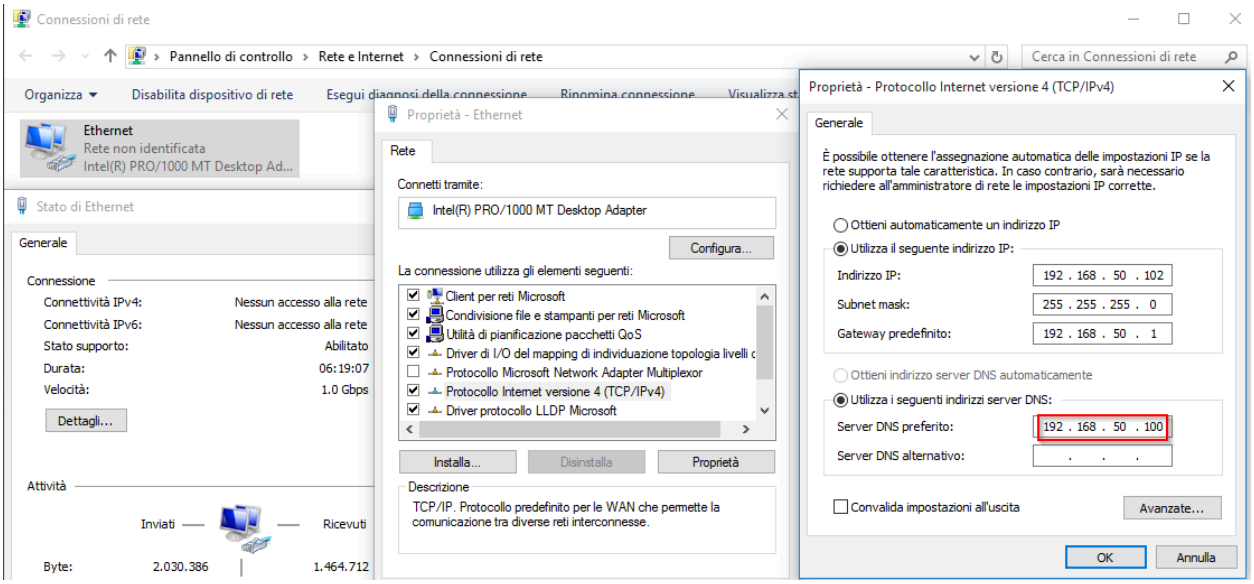
Per quanto riguarda sia il servizio https che http non facciamo modifiche.

Salvando e chiudendo il file configurazione, avviamo inetsim tramite l'apposito comando

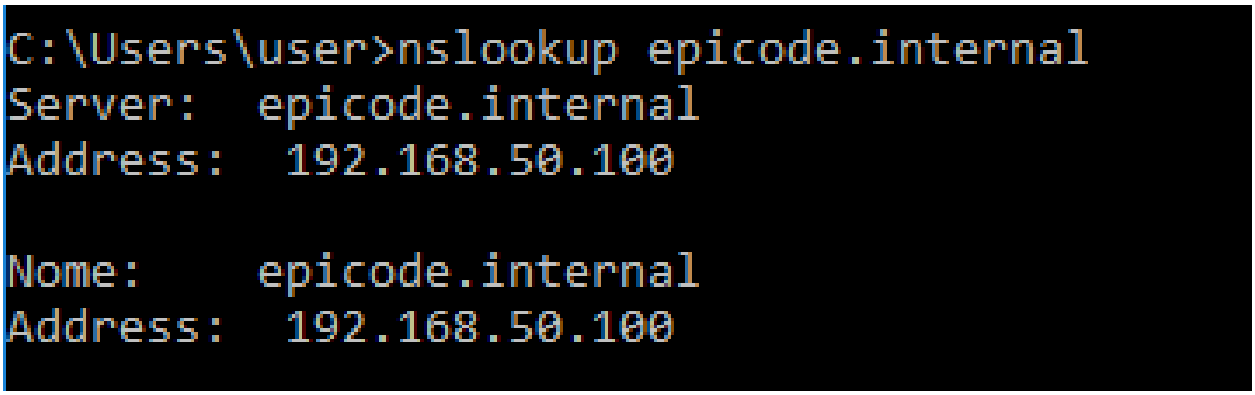
sudo inetsim

```
┌──(kali㉿kali)-[~]
└─$ sudo inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory:      /var/log/inetsim/
Using data directory:     /var/lib/inetsim/
Using report directory:   /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 196980) ===
Session ID:     196980
Listening on:   0.0.0.0
Real Date/Time: 2025-07-24 15:20:23
Fake Date/Time: 2025-07-24 15:20:23 (Delta: 0 seconds)
 Forking services ...
  * dns_53_tcp_udp - started (PID 196984)
  * https_443_tcp - started (PID 196985)
 done.
Simulation running.
```

Lasciamo la simulazione attiva sul terminale di kali e passiamo alla macchina windows 10 dove impostiamo l'IP del nostro record dns statico aggiunto dalle opzioni della scheda di rete



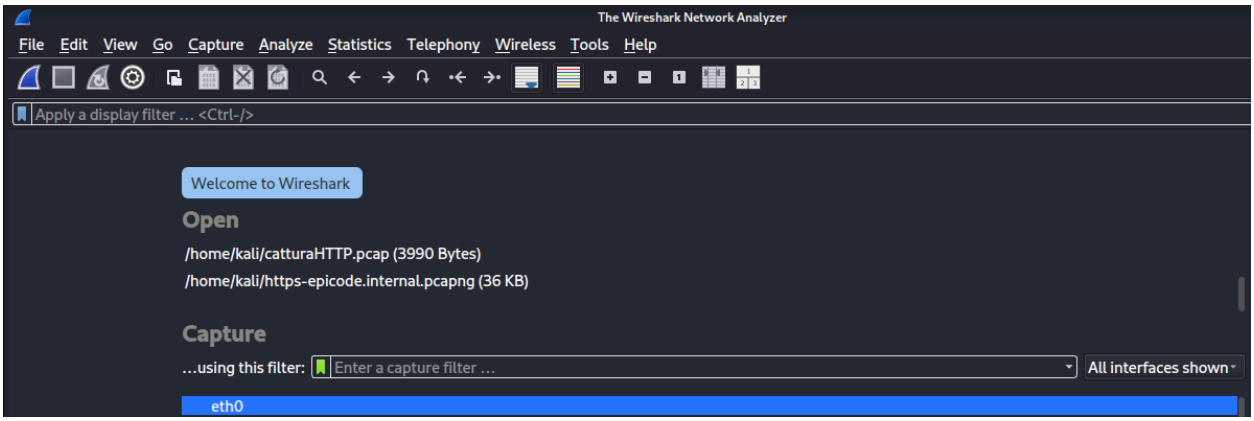
- Convalidiamo all’uscita e un ulteriore verifica se tutto procede bene, apriamo un prompt dei comandi e utlizziamo il comando nslookup con il nome del record dns aggiunto epicode.internal.



Possiamo procedere agli altri punti

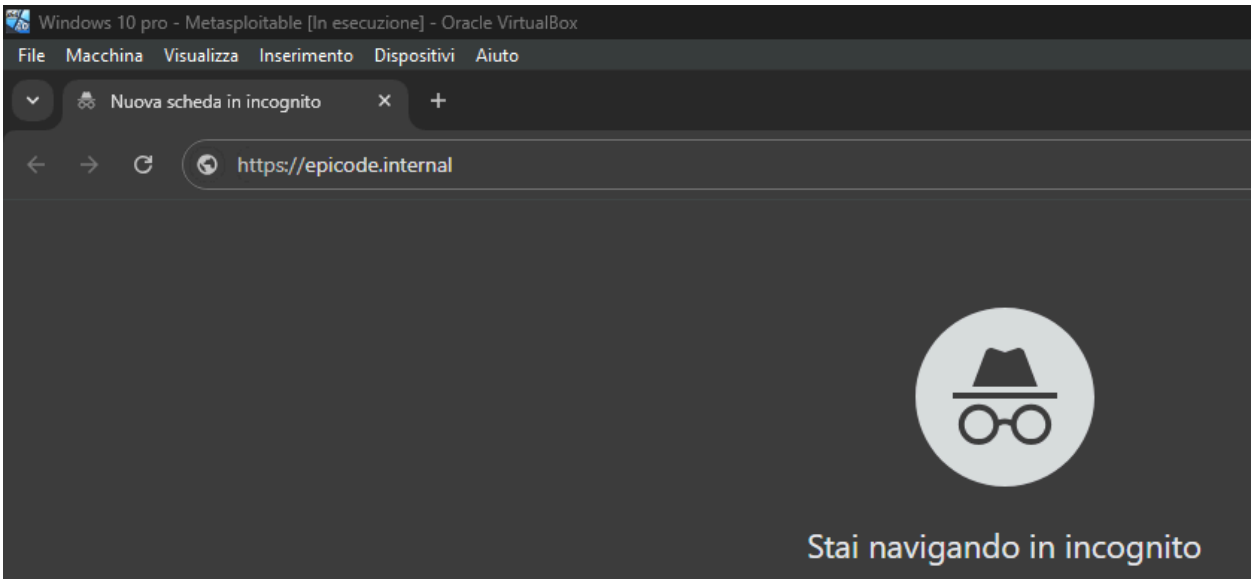
1. Si intercetti poi la comunicazione con Wireshark, evidenziando i MAC address di sorgente e destinazione ed il contenuto della richiesta HTTPS.

- Avviamo wireshark sulla macchina kali sull’interfaccia eth0 e torniamo sulla macchina windows.

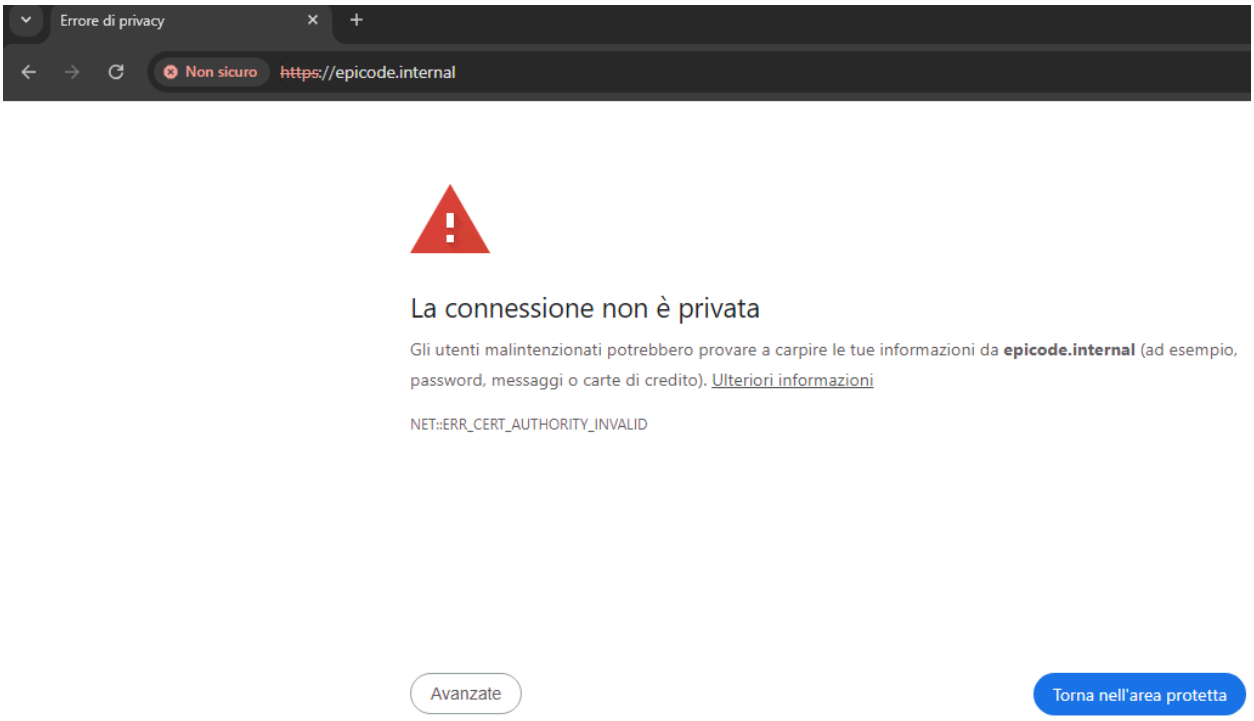


- Apriamo un web browser e sulla barra di ricerca scriviamo il seguente url:

https://epicode.internal

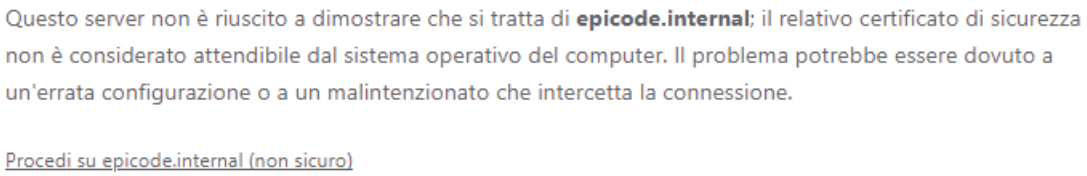


- Cliccando invio la nostra richiesta dovrebbe farci visualizzare la pagina di default di inetsim, ma ricordando che siamo su https e il record dns aggiunto non ha un certificato di sicurezza valido la pagina visualizzata sarà:

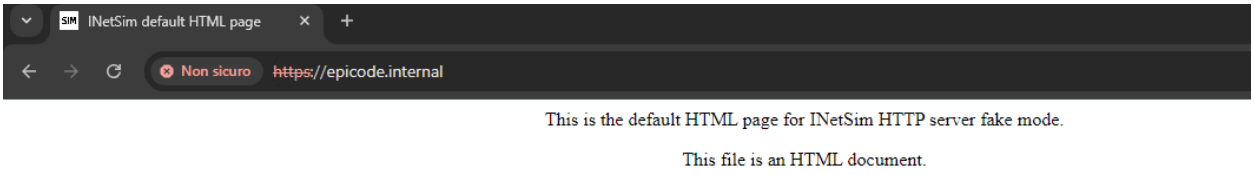


- Clicchiamo sul pulsante in basso avanzate ed mandiamo di nuovo la richesta cliccando

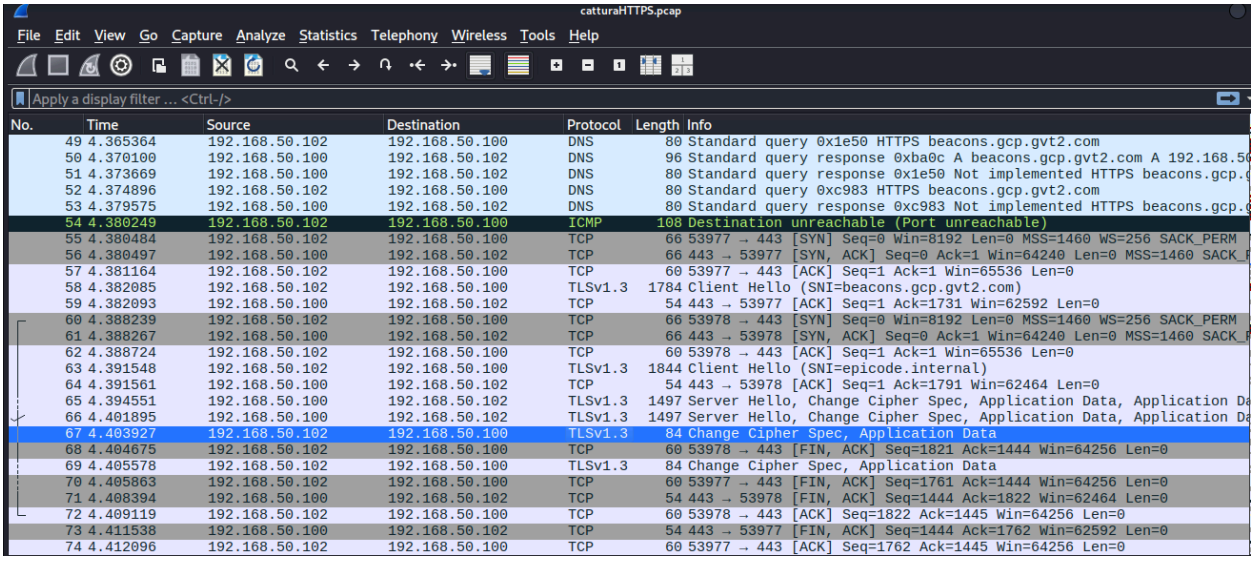
Procedi su epicode.internal (non sicuro)



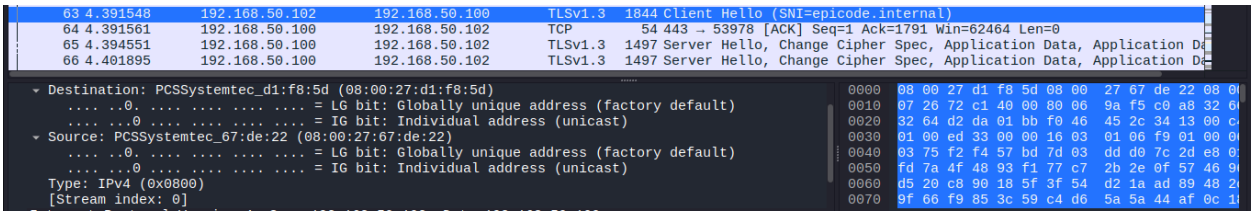
- Non appena la nostra richiesta viene inviata visualizzeremo la pagina sottostante



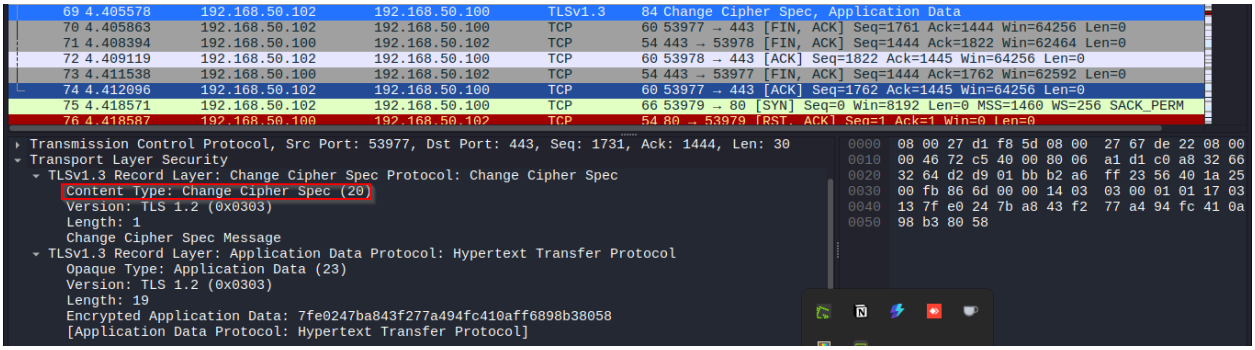
- Torniamo su wireshark e vediamo cosa abbiamo intercettato



- Mettiamo in evidenza i MAC address di sorgente e destinazione

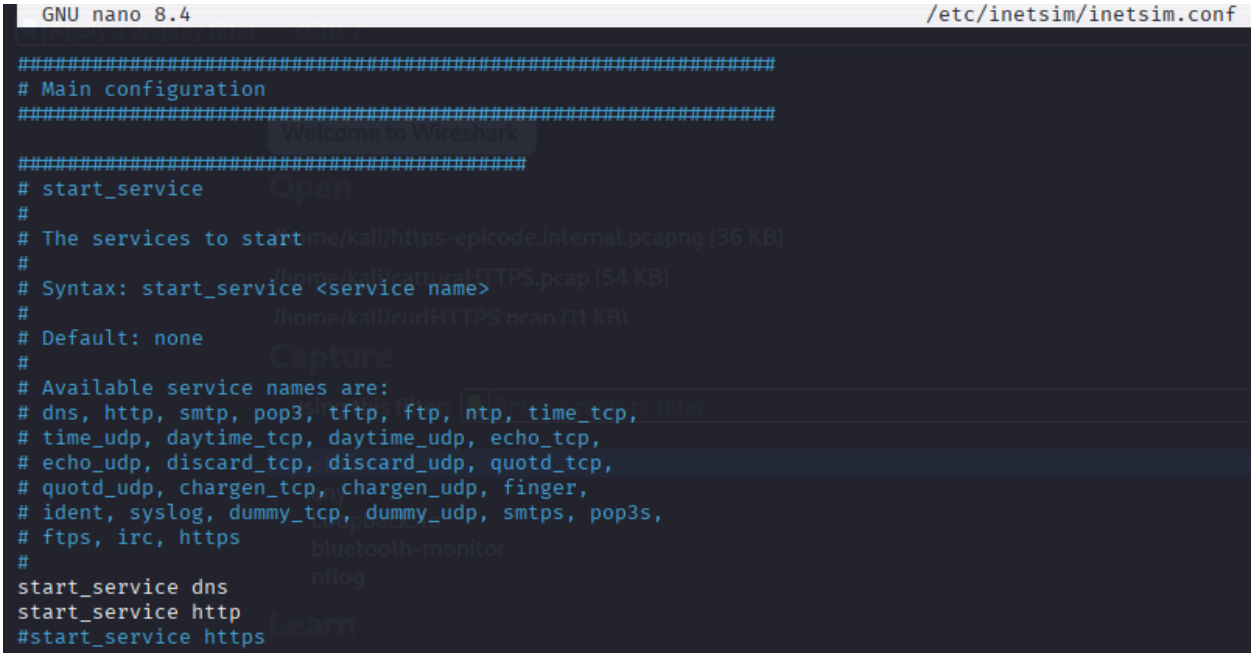


- Cerchiamo di controllare il contenuto, ma come possiamo vedere è crittografato

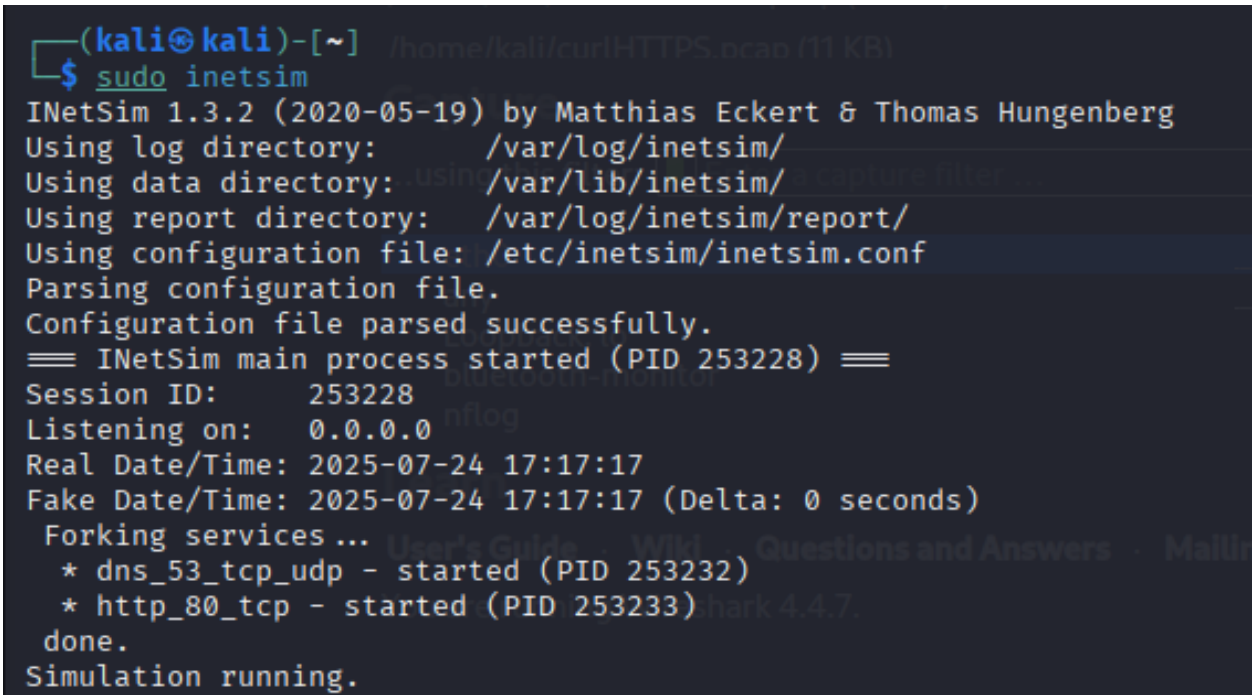


2. Ripetere l’esercizio, sostituendo il server HTTPS, con un server HTTP. Si intercetti nuovamente il traffico.

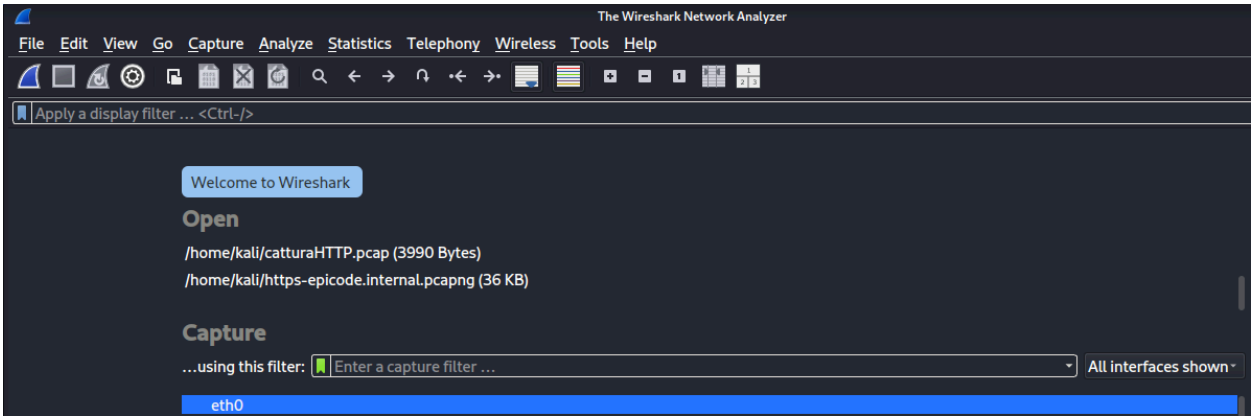
- Torniamo sul file di configurazione di inetsim e commentiamo il servizio https e mettiamo in esecuzione il servizio http.



- Dopodiché mettiamo in esecuzione inetsim con l'apposito comando usato precedentemente

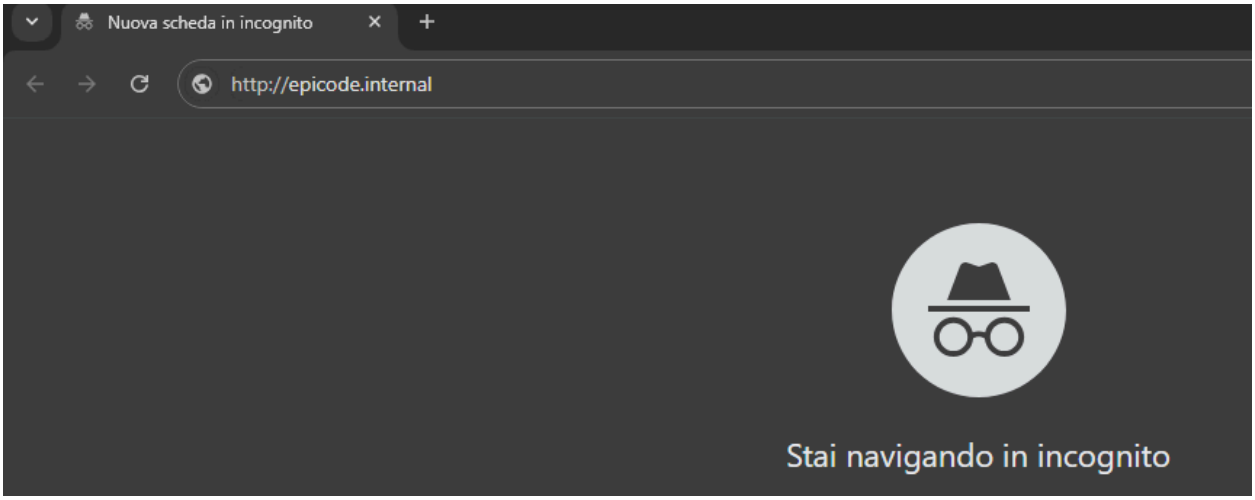


- Avviamo wireshark sulla macchina kali sull'interfaccia eth0 e torniamo sulla macchina windows.

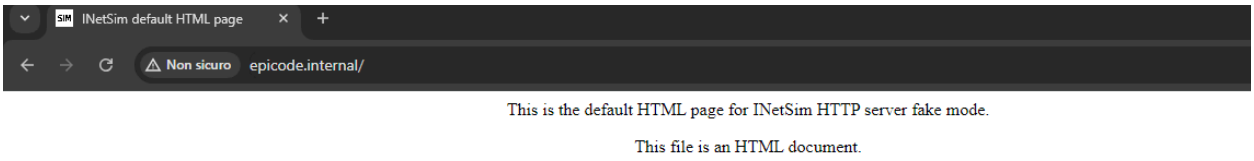


- Apriamo un web browser e sulla barra di ricerca scriviamo il seguente url:

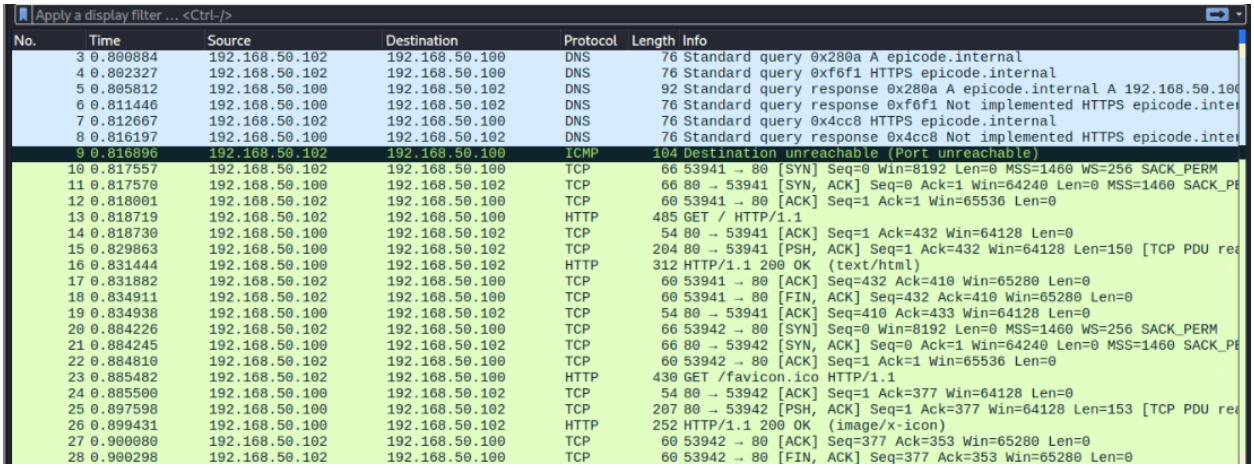
http://epicode.internal



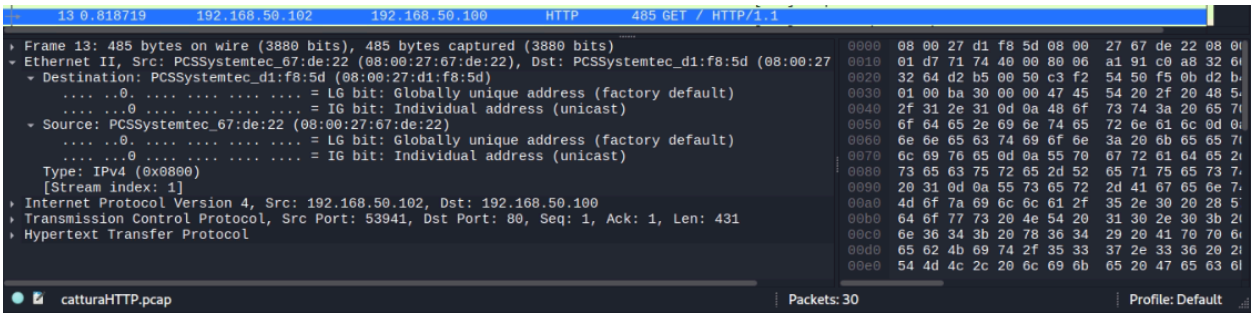
- Cliccando invio la nostra richiesta dovrebbe farci visualizzare la pagina di default di inetsim



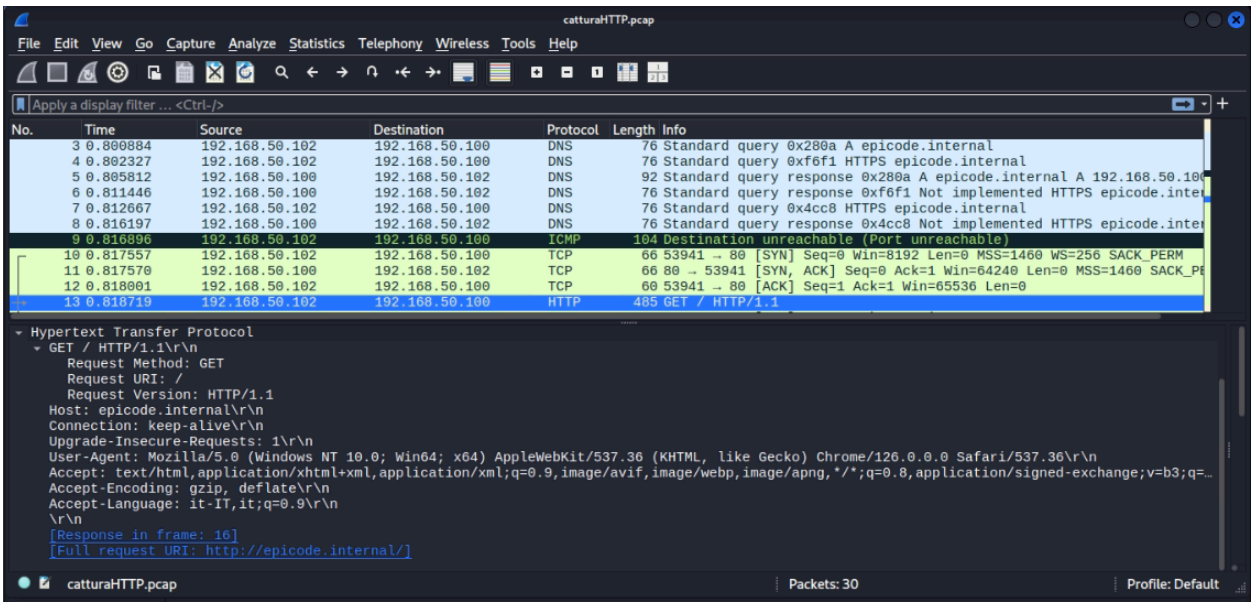
- Torniamo su wireshark e vediamo cosa abbiamo intercettato



- Mettiamo in evidenza i MAC address



- Controlliamo il contenuto della richiesta



3. Evidenzia le eventuali differenze tra il traffico appena catturato in HTTP ed il traffico precedente in HTTPS.

Come possiamo ben vedere rispetto a http, https prevede una comunicazione tramite TLS, quindi cifrando il contenuto, rispetto che lasciarlo in chiaro.

4. Spiegare, motivandole, le principali differenze se presenti

La differenza maggiore è sullo scambio tramite i 3-way hanshake, mentre per l'http vengono svolti solo 3 scambi (SYN, SYN ACK e ACK), inviando subito dopo il contenuto, per l'https utilizzando TLS (è una negoziazione di sicurezza che avviene prima di trasmettere qualsiasi dato. Il client e il server si accordano su quale algoritmo di crittografia usare, come scambiare le chiavi segrete, come verificare l'identità del server) dopo lo scambio di chiavi il contenuto della richiesta sarà cifrato

N.B. Durante lo svolgimento dell'esercizio abbiamo riscontrato dei problemi con inetsim, risolti tramite la sequenza di questi quattro comandi

```
┌──(kali㉿kali)-[~]
└─$ sudo cpan -f -u Net::DNS
Loading internal logger. Log::Log4perl recommended for better logging
Reading '/root/.cpan/Metadata'
Warning: Found only 0 objects in /root/.cpan/Metadata
Fetching with HTTP::Tiny:
https://cpan.org/authors/01mailrc.txt.gz
Reading '/root/.cpan/sources/authors/01mailrc.txt.gz'
.....................................................................DONE
Fetching with HTTP::Tiny:
https://cpan.org/modules/02packages.details.txt.gz
Reading '/root/.cpan/sources/modules/02packages.details.txt.gz'
  Database was generated on Thu, 24 Jul 2025 06:52:43 GMT
.............
  New CPAN.pm version (v2.38) available.
  [Currently running version is v2.36]
  You might want to try
    install CPAN
    reload cpan
  to both upgrade CPAN.pm and run the new version without leaving
  the current session.


.....................................................................DONE
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install cpanminus
The following packages were automatically installed and are no longer required:
  python3-packaging-whl  python3-pyinstaller-hooks-contrib  python3-wheel-whl
Use 'sudo apt autoremove' to remove them.

Installing:
  cpanminus
```

```
┌──(kali㉿kali)-[~]
└─$ sudo cpanm --notest Net::DNS@1.22
--> Working on Net::DNS
Fetching http://backpan.perl.org/authors/id/N/NL/NLNETLABS/Net-DNS-1.22.tar.gz ... OK
Configuring Net-DNS-1.22 ... OK
Building Net-DNS-1.22 ... OK
Successfully installed Net-DNS-1.22 (downgraded from 1.50)
1 distribution installed
```