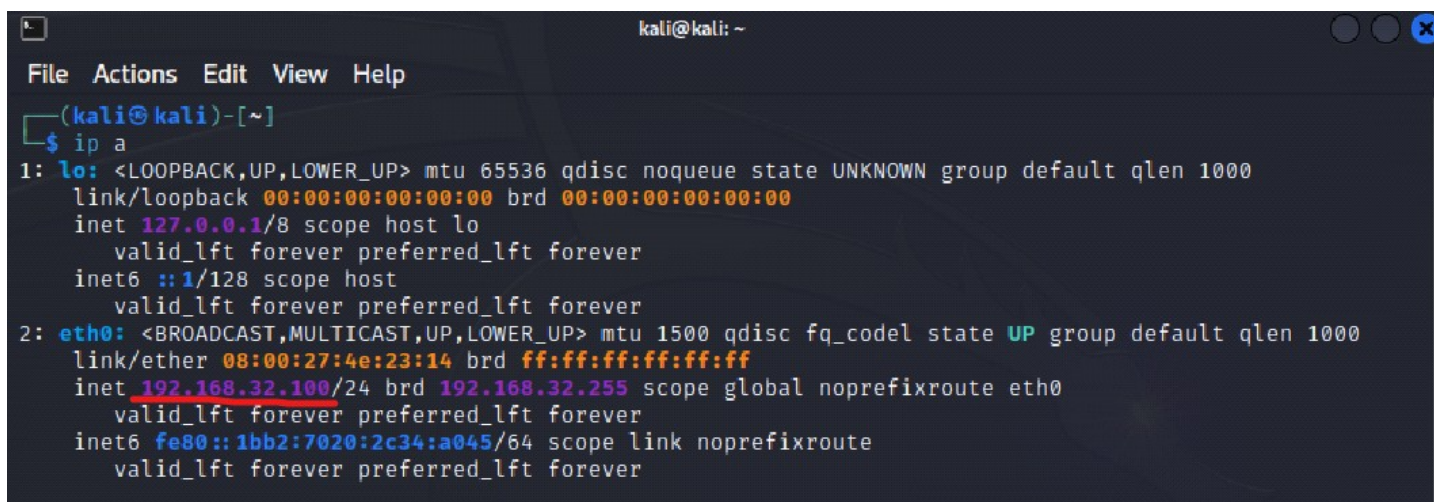


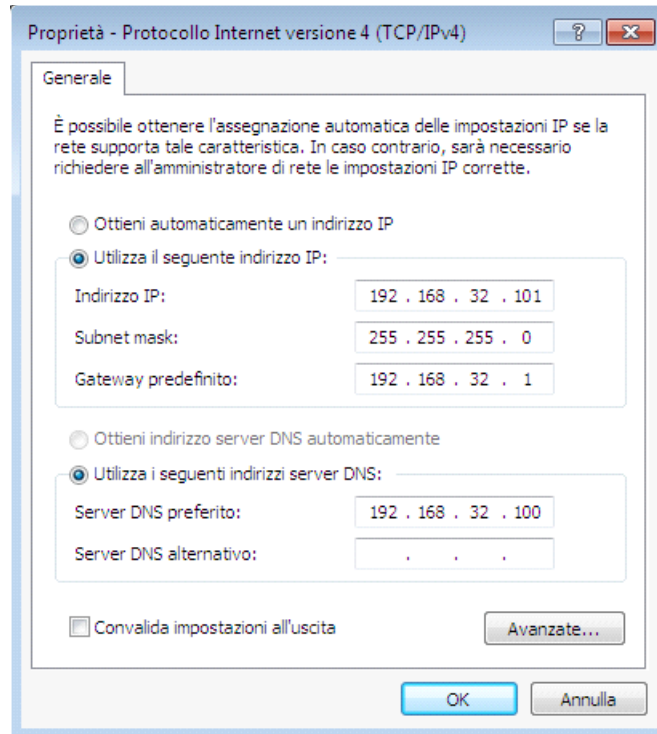
Report

Comincio l'esercizio avviando la VM (Virtual Machine) Kali Linux su rete interna, dopodichè configuro la rete con l'indirizzo IP statico richiesto dall'esercizio, andando anche a configurare lo spazio "DNS server" con lo stesso IP. Questo IP servirà a configurare il server virtuale DNS su InetSim. Inserisco 192.168.32.1 come gateway e 24 come netmask. (Vedi sotto)



```
kali@kali: ~
File Actions Edit View Help
(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
        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 08:00:27:4e:23:14 brd ff:ff:ff:ff:ff:ff
    inet 192.168.32.100/24 brd 192.168.32.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::1bb2:7020:2c34:a045/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Adesso avvio la VM Windows 7 su rete interna. Dal Pannello di Controllo vado a configurare la rete e ad assegnare l'indirizzo IP statico richiesto dall'esercizio, così come il gateway e il server DNS preferito. Il gateway rimane lo stesso di Kali Linux, essendo su rete interna, e il server DNS equivale all'IP di Kali Linux.



Torno su Kali Linux. Apro il Terminal e scrivo il comando "sudo mousepad /etc/inetsim/inetsim.conf". Modifico il service_bind_address con l'IP statico di Kali Linux, associando così il servizio alla VM. Inoltre, rimuovo i cancelletti per disattivare i commenti.

```
60 #####
61 # service_bind_address
62 #
63 # IP address to bind services to
64 #
65 # Syntax: service_bind_address <IP address>
66 #
67 # Default: 127.0.0.1
68 #
69 service_bind_address 192.168.32.100
```

Scorrendo in basso, modifico anche il dns_default_ip con lo stesso IP usato su Kali, cioè l'indirizzo IP che andrà a ricevere le risposte DNS.

```
199 # dns_default_ip
200 #
201 # Default IP address to return with DNS replies
202 #
203 # Syntax: dns_default_ip <IP address>
204 #
205 # Default: 127.0.0.1
206 #
207 dns_default_ip 192.168.32.100|
```

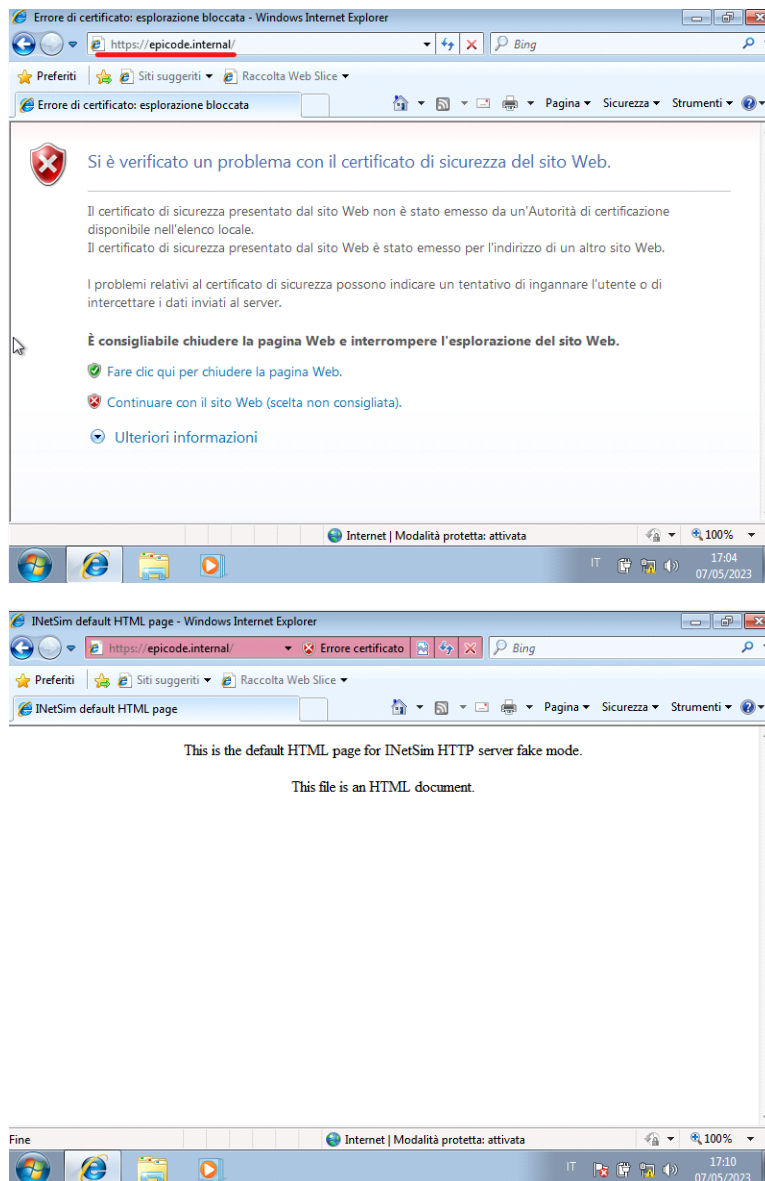
Modifico dns_static con l'IP usato in precedenza, aggiungendo l'hostname "epicode.internal". Ho così configurato un DNS server virtuale su Kali Linux, che andrà a interagire con il client Windows 7.

```
235 # dns_static
236 #
237 # Static mappings for DNS
238 #
239 # Syntax: dns_static <fqdn hostname> <IP address>
240 #
241 # Default: none
242 #
243 dns_static epicode.internal 192.168.32.100|
```

Salvo la configurazione, chiudo InetSim e apro il terminale. Inserisco il comando "sudo inetsim" per avviare il server virtuale con tutti i servizi attivi più il DNS appena configurato.

```
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 9887) ===
Session ID:      9887
Listening on:    192.168.32.100
Real Date/Time:  2023-05-07 10:55:39
Fake Date/Time: 2023-05-07 10:55:39 (Delta: 0 seconds)
Forking services ...
* dns_53_tcp_udp - started (PID 9897)
* ident_113_tcp - started (PID 9910)
* irc_6667_tcp - started (PID 9907)
* time_37_tcp - started (PID 9912)
* daytime_13_udp - started (PID 9915)
* discard_9_tcp - started (PID 9918)
* smtp_25_tcp - started (PID 9900)
* smtps_465_tcp - started (PID 9901)
* ntp_123_udp - started (PID 9908)
* ftps_990_tcp - started (PID 9905)
* pop3s_995_tcp - started (PID 9903)
* syslog_514_udp - started (PID 9911)
* ftp_21_tcp - started (PID 9904)
* https_443_tcp - started (PID 9899)
* http_80_tcp - started (PID 9898)
* tftp_69_udp - started (PID 9906)
* daytime_13_tcp - started (PID 9914)
* finger_79_tcp - started (PID 9909)
* echo_7_tcp - started (PID 9916)
* pop3_110_tcp - started (PID 9902)
* time_37_udp - started (PID 9913)
* discard_9_udp - started (PID 9919)
* echo_7_udp - started (PID 9917)
* quotd_17_udp - started (PID 9921)
* dummy_1_tcp - started (PID 9924)
* chargen_19_udp - started (PID 9923)
* dummy_1_udp - started (PID 9925)
* chargen_19_tcp - started (PID 9922)
* quotd_17_tcp - started (PID 9920)
done.
Simulation running.
```

Mantenendo il terminale aperto con la simulazione in corso, torno su Windows 7 per richiedere una risorsa sul web browser all'hostname configurato. Il browser non trova il certificato SSL e di conseguenza il sito appare come pericoloso. Tuttavia, se si ignora l'avviso e si entra comunque nel sito, esso funziona a dovere.

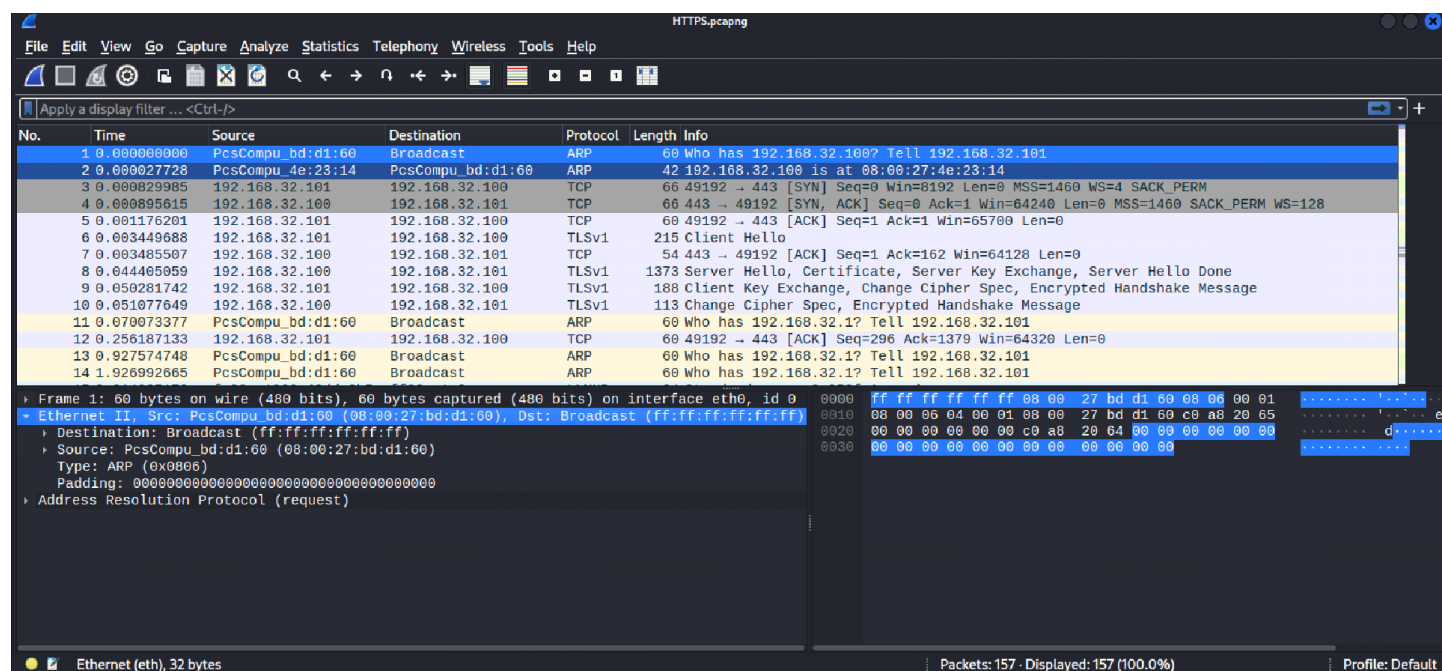


Tornando su Kali Linux, apro l'applicativo Wireshark per intercettare la comunicazione tra il client (Windows 7) e il server (Kali Linux). Per farlo avvio la cattura tramite il pulsante in alto a sinistra e vado su Windows ad aggiornare la pagina "epicode.internal". Quando torno su Wireshark fermo la cattura e ho così ottenuto i pacchetti derivati dalla richiesta al sito web, che risponde all'IP di Kali Linux.

Nel primo pacchetto, il protocollo ARP con porta 443 (HTTPS) viene mandato con destinazione broadcast (MAC corrispondente ff:ff:ff:ff:ff:ff) per la richiesta di associazione MAC dell'indirizzo IP 192.168.32.100

Nel secondo pacchetto, il protocollo ARP invia una risposta associando il MAC che sarebbe 08:00:27:4e:23:14 (Vedi sotto). I pacchetti 3, 4 e 5 appartengono al client e al server che aprono la comunicazione tramite protocollo TCP con i flag SYN, SYN ACK e ACK per il trasporto dei dati.

I pacchetti con protocollo TLSv1 sono coloro che si occupano di crittografare i dati e rendere la connessione sicura dalla sorgente al destinatario.



Adesso sostituisco il server HTTPS con uno HTTP e intercetto i pacchetti per individuare le differenze.

Nel protocollo HTTP si può notare che il contenuto del pacchetto è un testo leggibile, e perciò non crittografato. Mancano infatti i pacchetti con protocollo TLSv1.

Il protocollo TCP ha come porta 80 (HTTP).

