PROGETTO FINALE 1°MODULO

EPICODE

Studente: Fabio Belforti

Docenti corso: Giuseppe Placanica | Edoardo Castelli

22 March 2025

PRESENTAZIONE

Nell’esercitazione di oggi, metteremo insieme le competenze apprese fino ad ora. Ricapitolando i passaggi principali per le più basiche tipologie di protocolli di comunicazione tra client e approfondendo la parte tecnica con i concetti acquisiti.

OBIETTIVI

1)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.

2) 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.

IP & 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

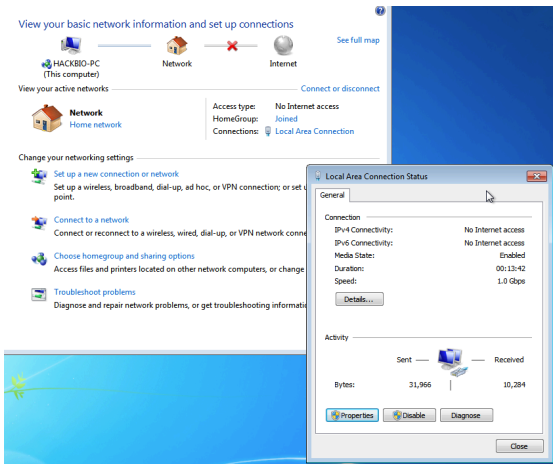
PRESENTAZIONE SVOLGIMENTO
PRIMA RICHIESTA

La svolgimento del progetto richiede la configurazione, all'interno di una stessa rete locale, di una comunicazione tra due client. In particolare la macchina Kali, dovrà fungere da HOST per l'utilizzo dei servizi HTTPS , per la cifratura della comunicazione e DNS per la risoluzione del dominio. Saranno attivati tramite relativi tools, scaricati da internet o già preinstallati. Questo consentirà alla macchina windows di raggiungere il server web tramite il nome di dominio anziché tramite indirizzo IP.

SVOLGIMENTO

Configurazione Windows 7

Una volta impostate correttamente le schede di rete, in condivisione solo sulla rete interna, lanceremo l'avvio delle 2 macchine virtuali



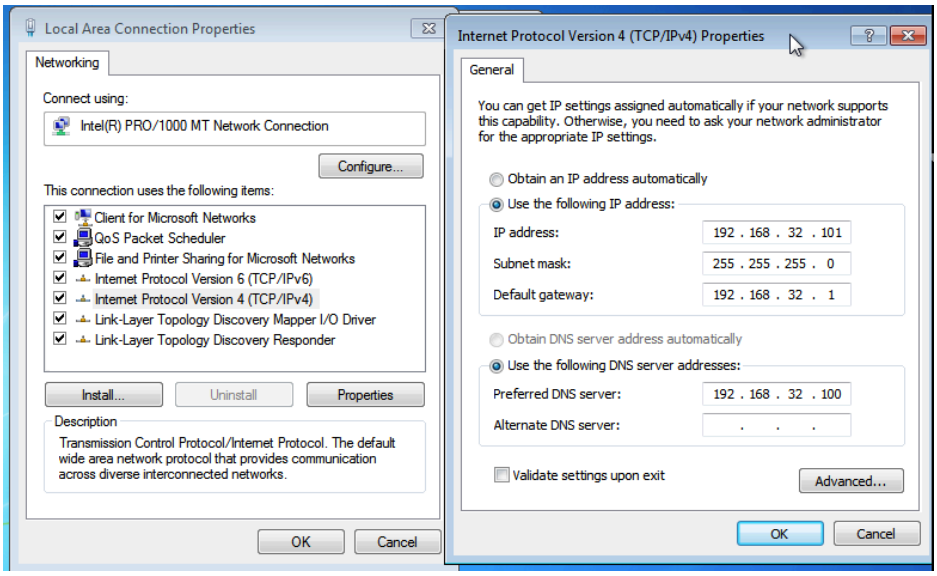
Per iniziare, occorre configurare la scheda di rete Windows pet prepararla alla comunicazione sulla rete con Kali Linux.

Aprendo impostazioni scheda, dovremo andare a configurare l'indirizzo IP statico, aggiungendo già anche l'indirizzo ip a cui assegneremo il server DNS che vedremo successivamente.

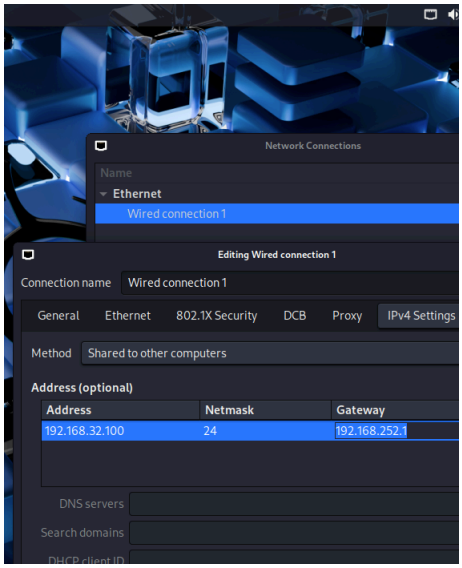
L'IP che andremo ad impostare è: 192.168.32.101 con Subnet Mask 255.255.255.0 con un indirizzo di default del Gateway di 192.168.32.1

L'IP invece dell'indirizzo del DNS è quello che sarà appartenente anche alla macchina di Kali che corrisponde a:192.168.32.100

Questo consentirà al server DNS impostato su Kali di risolvere le richieste di dominio generate dalla macchina virtuale Windows.



Configurazione Kali

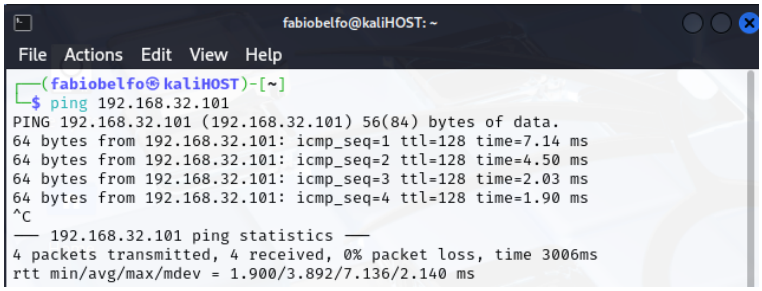


Come fatto per Windows, occorre impostare un indirizzo IP statico alla macchina

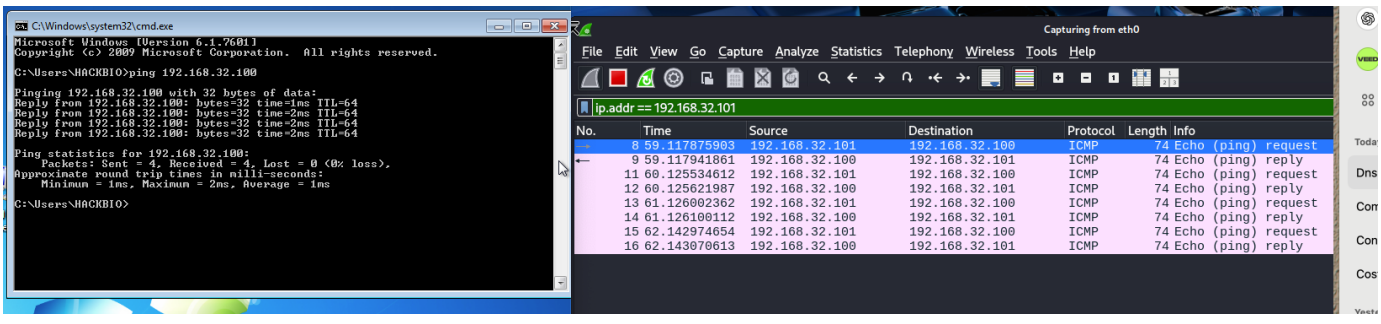
Importante la scheda EHT0 in modalità “Shared other computer” avremo la possibilità, manualmente, di aggiungere l’indirizzo ip 192.168.32.100 con relativa subnet, in questo caso scritta con solo 24, perché in notazione CIDR (Classless Inter-Domain Routing) per abbreviare.

Fatto ciò, verifichiamo le configurazioni appena effettuate tramite un comando di “PING” che sfrutta il protocollo ICMP (Internet Control Message Protocollo) al livello 3 del modello standard ISO/OSI.

KALI—->WINDOWS ✔

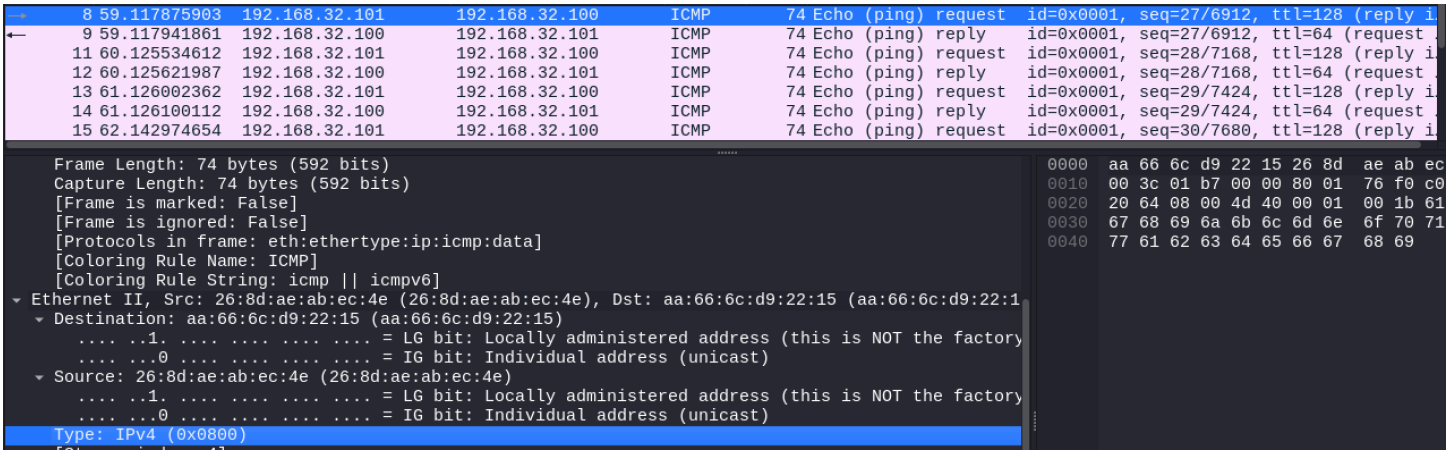


WINDOWS —->KALI ✔

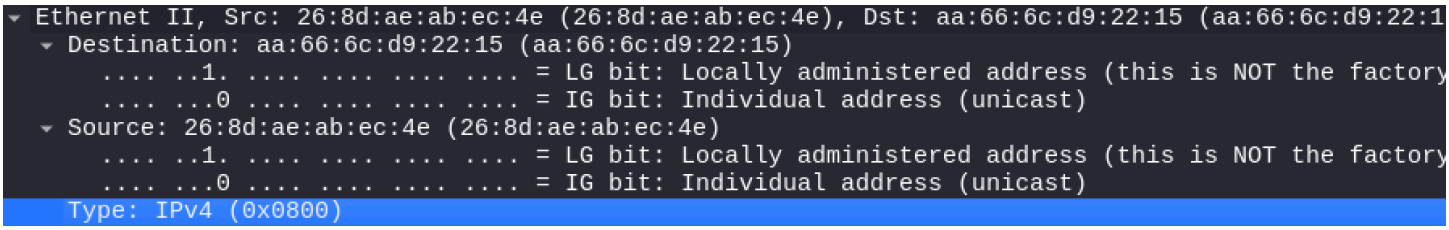


OSSERVAZIONE COMUNICAZIONE

Operazione di “SNIFFING” sulla porta ethernet di KALI “ETH0”, andiamo ad osservare la trasmissione del comando di PING tramite utilizzo di software specifico WIRESHARK, già pre-installato sulla macchina di Kali Linux, la comunicazione che avviene tra le due macchine tramite protocollo ICMP.

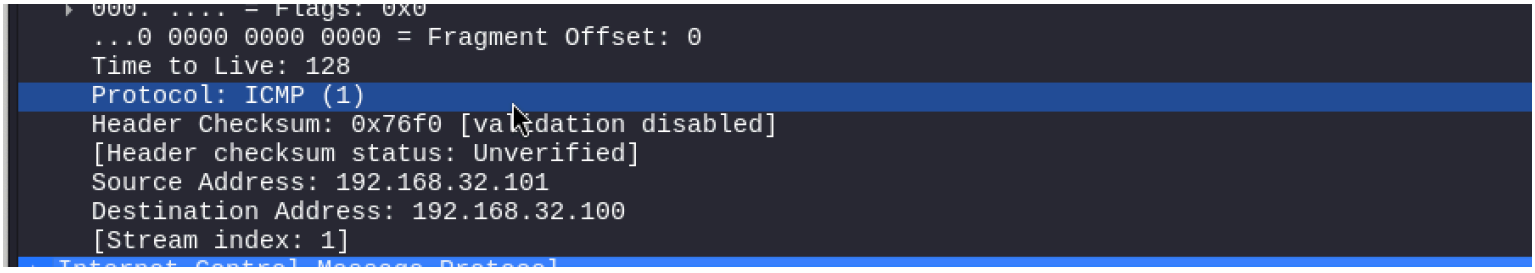


Come evidenziato dalla striscia di colore blu nell’immagine, è rappresentato il primo messaggio dall’IP sorgente all’IP destinatario. Chi invia, inoltrerà una ICMP Echo Request.

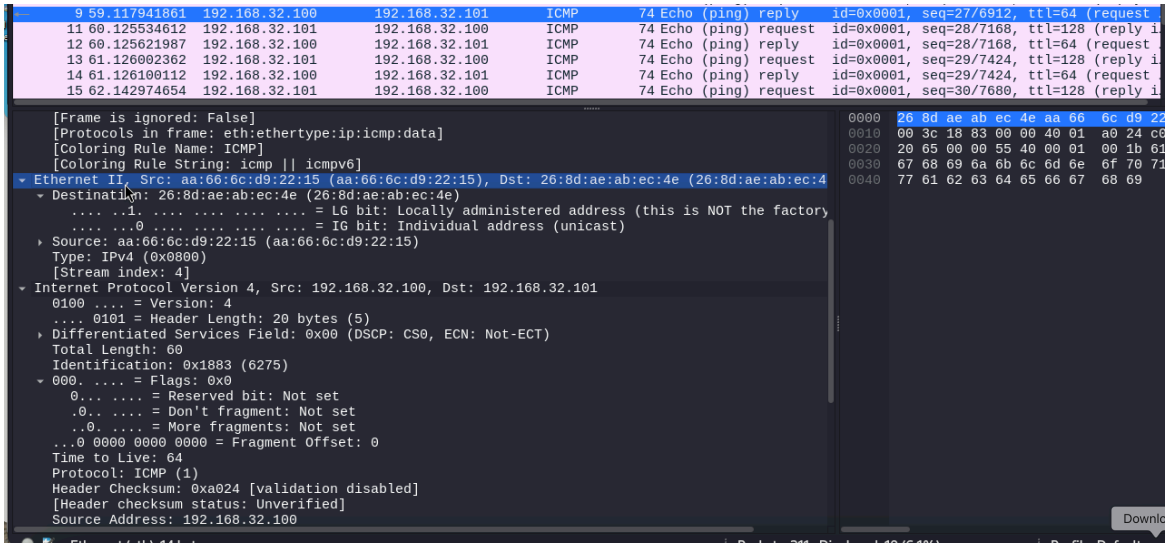


Inparticolare, cliccandoci sopra, potremo verificare la commissione del pacchetto in trasmissione. Questi indirizzi MAC sono fondamentali per la trasmissione del pacchetto nella rete locale (livello 2 del modello OSI). Se il dispositivo di origine non ha già associato l’indirizzo MAC del destinatario, il protocollo ARP entra in gioco per risolvere l’indirizzo IP del destinatario in un indirizzo MAC corrispondente. Questa operazione avviene “dietro le quinte” e non è esplicitamente visibile nel pacchetto ICMP, ma è implicita nel fatto che ARP ha precedentemente eseguito l’associazione tra gli indirizzi IP e MAC.

Mentre qui avviene lo scambio d indirizzi IP delle due macchine.



Stessa cosa avverrà in senso contrario, quando la macchina Kali, risponderà alla richiesta di connessione, questa volta con ICMP echo reply.



IMPOSTAZIONE SERVIZI

Inetsim (HTTPS)

```
# 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
#start_service smtps
#start_service pop3
#start_service pop3s
#start_service ftp
#start_service ftps
#start_service tftp
#start_service irc
#start_service ntp
#start_service finger
#start_service ident
#start_service syslog
```

Con un semplice comando installiamo il tool di Kali

//sudo apt install inetsim

Si procede con apertura file configurazione

//sudo nano /etc/inetsim/inetsim.conf (che non è altro che un file di testo presente nella directory ETC del nostro sistema)

Ci verrà mostrata una lista di servizi disponibili su INETSIM, a noi in questo caso interessa usare solo HTTPS da qui.

Per procedere con l'attivazione è necessario decommentare la riga interessata per renderla leggibile al compilatore.

Più avanti nel file, troveremo la sezione “Service_bind_address” ,andando ad inserire l’indirizzo IP di Kali, comanderemo l’ascolto dello specifico indirizzo IP, per offrire i servizi di cifratura che sono presenti nel protocollo HTTPS. Il protocollo, “HperText Transfer Protocol”, tramite altro protocollo “SSL/TLS”, garantisce la cifratura dei dati trasmessi e ricevuti sulla porta designata.

```
#############################################
# service_bind_address
#
# IP address to bind services to
#
 service_bind_address 192.168.32.100
#
```

```
############################################################
# Service HTTPS
############################################################

#####################################
# https_bind_port
#
# Port number to bind HTTPS service to
#
# Syntax: https_bind_port <port number>
#
# Default: 443
#
https_bind_port 443
```

Infine nella pagina dedicata al servizio HTTPS, potremo attivare la porta per lavorare e estensioni che posso utilizzare per simulare traffico.

Una volta scritti i parametri corretti occorre salvare il file, riavviare inetsim e successivamente attivare definitivamente il servizio inviando a terminale:
//Sudo inetsim

Se la procedura è stata svolta correttamente il terminale risponderà al comando lanciando l’esecuzione del tool INETSIM dichiarandoci lo stato di eseguita operazione e corretta attivazione come mostrato in figura.

```
┌──(fabiobelfo㉿kaliHOST)-[~]
└─$ sudo inetsim
[sudo] password for fabiobelfo:
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 209190) ===
Session ID:     209190
Listening on:   192.168.32.100
Real Date/Time: 2025-03-22 13:39:11
Fake Date/Time: 2025-03-22 13:39:11 (Delta: 0 seconds)
 Forking services...
  * https_443_tcp - started (PID 209192)
 done.
Simulation running.
```

IMPOSTAZIONE SERVIZI

Dnsmasq (DNS)

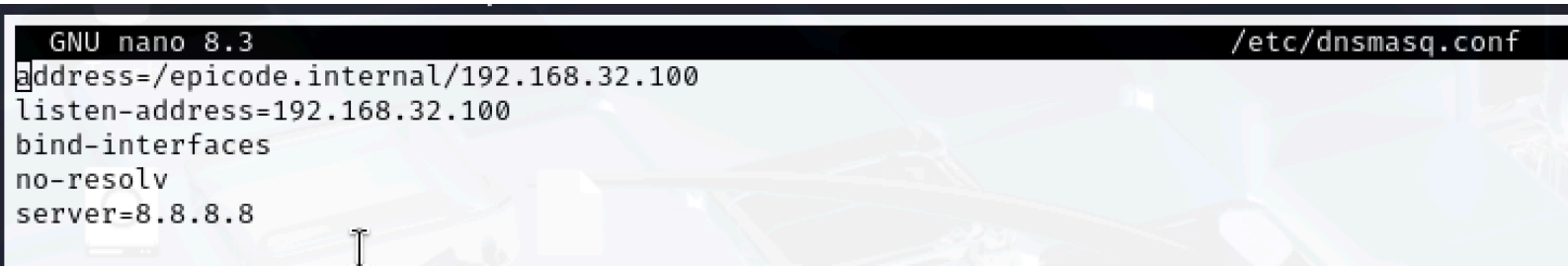
Dnsmasq è un tool installabile in maniera simile a quanto fatto prima con inetsim.
Occorrerà condividere internet dalla macchina HOST, per poter procedere al download del file direttamente da terminale di Kali.

// sudo apt install dnsmasq

Come prima andremo a modificare il file di configurazione, eseguendo sempre comando a terminale, scrivendo sempre “nano” per utilizzare editor di testo file.

//sudo nano /etc/dnsmaq.conf

Questo campo andrà compilato con ciascuna informazione essenziale per DNSMASQ:



- Indirizzo ip a cui associare il nome del dominio. (address=/..)
-L’indirizzo da cui ascolterà le richieste DNS. (Listen-address)
-Informazioni per fargli leggere specifico indirizzo (bind-interfaces)
-Server nel caso non riesca a risolvere una richiesta DNS(8.8.8.8)
-“No resolv” per non usare cartella di sistema per risoluzione DNS, ma quella indirizzata direttamente nella cartella DNSMASQ con l’informazione server.

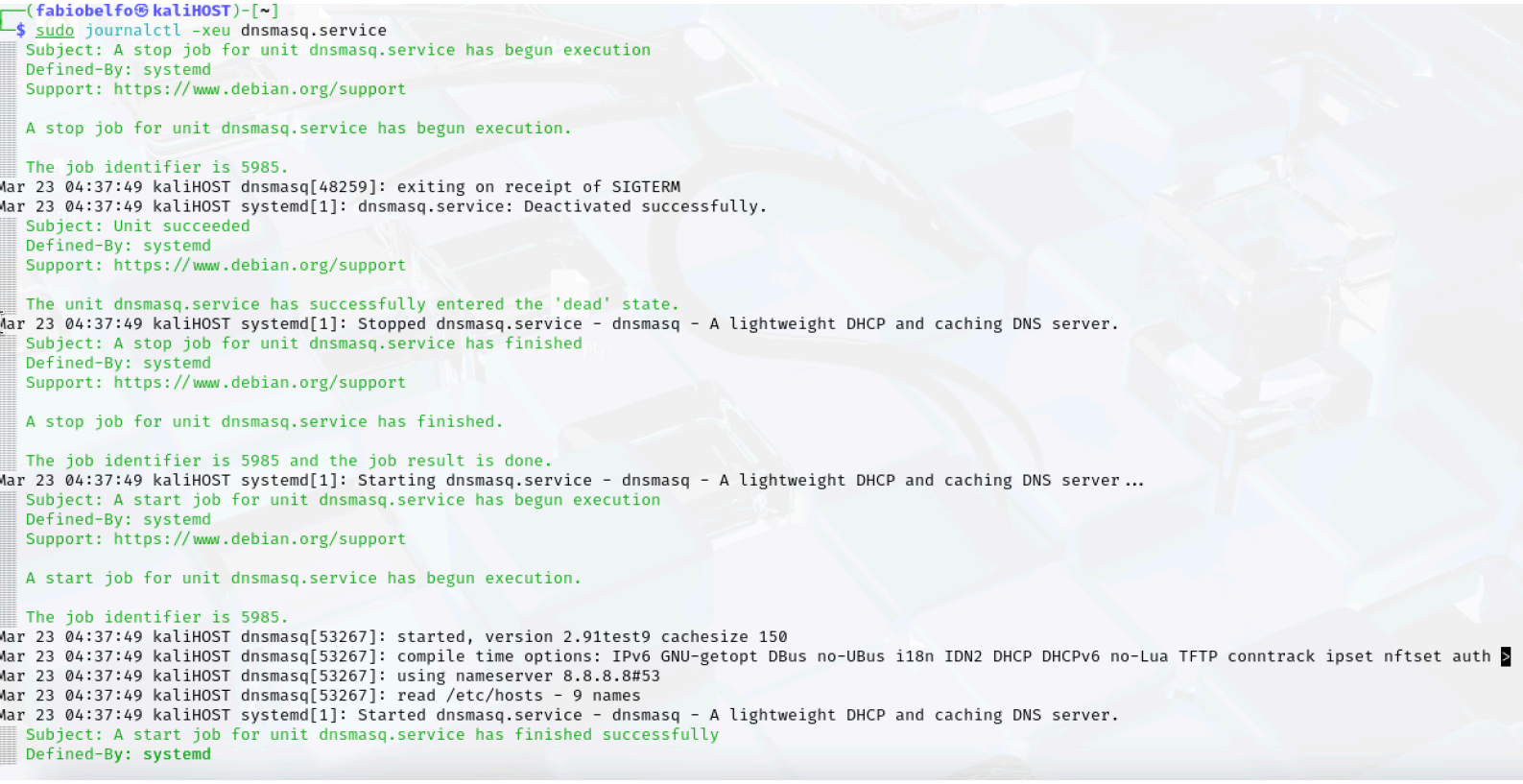
Una volta settato necessario riavvio del tool.

Ora controlliamo l’effettiva esecuzione corretta del tool con
Comando —-> //sudo systemctl status dnsmasq.service

Come possiamo osservare il servizio è stato correttamente attivato.

In caso di errori abbiamo a disposizione da tool dnsmasq due righe di comando per verificare i file log che si generano all'avvio dell'applicativo, utile in caso di errori di scrittura o configurazione.

//sudo journalctl -xeu dnsmasq.service

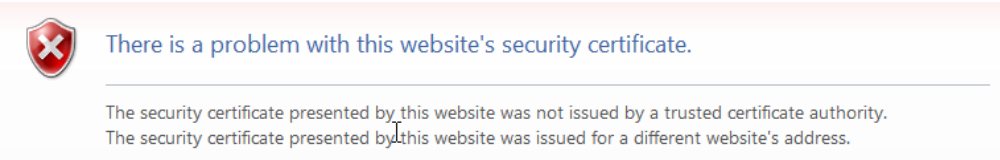


CHECK RISULTATO

- Da Windows, la prova per verificare la corretta inizializzazione del servizio DNS di “dnsmasq”, occorre:
  -Aprire Terminale;
  -Digitare comando // nslookup epicode.internal;

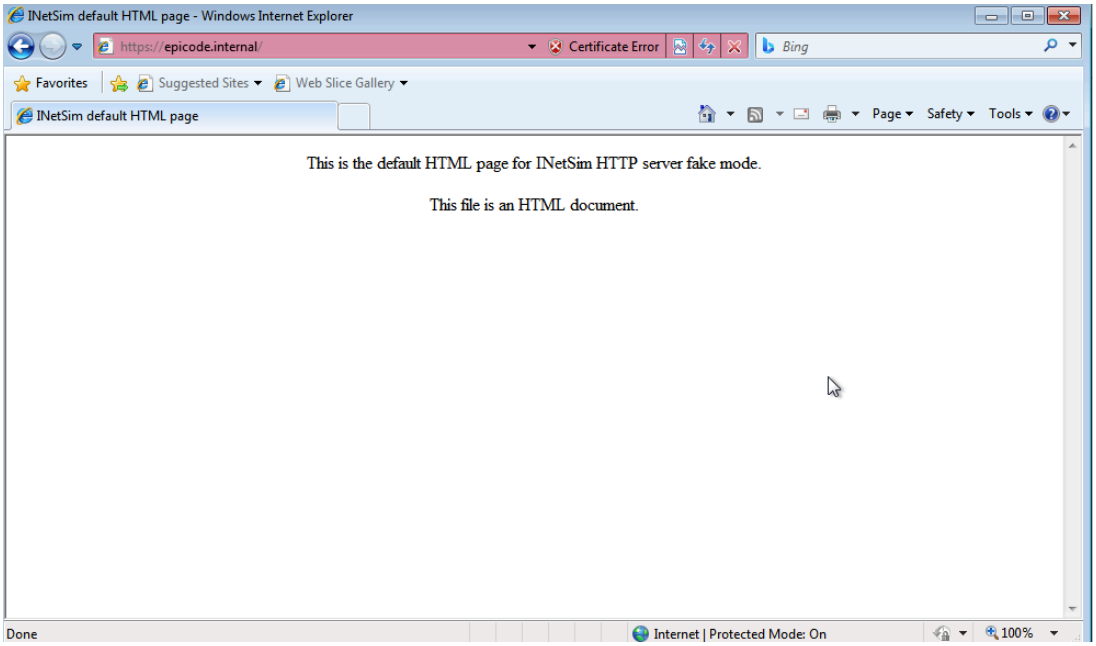
```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\HACKBIO>nslookup epicode.internal
Server:  epicode.internal
Address:  192.168.32.100

Name:    epicode.internal
Address:  192.168.32.100
```

Il servizio “dnsmasq” risponderà con il nome di dominio associato all’indirizzo ip configurato, possiamo quindi affermare che la risoluzione è avvenuta con successo. Prova ulteriore ci verrà fornita controllando anche tramite Web browser, digitando “https://epicode.internal" grazie al tool “inetsim” che genererà un servizio HTTPS.



-Il problema del certificato è dovuto poiché “inetsim” genera un certificato che non è ufficiale.

Proseguendo nella ricerca web otterremo la nostra pagina desiderata corrispondente al dominio Epiode.internal



SNIFFING CON WIRESHARK DELLA COMUNICAZIONE

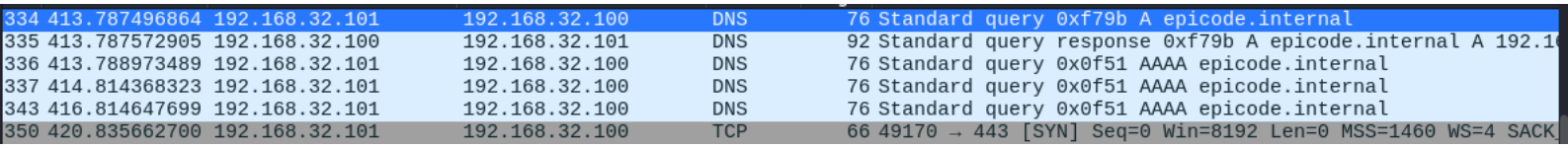
Grazie all'applicazione wireshark abbiamo la possibilità di analizzare i pacchetti in entrata ed uscita dalla nostra macchina Kali Linux. Occorre mettersi in ascolto sulla porta ETH0 e lanciare la richiesta DNS da windows richiedendo la risoluzione del nome epicode.internal.

SERVICE DNS

DNS Query,"RICHIESTE"(pacchetti 1, 7, 10)

Il protocollo DNS viene utilizzato per risolvere i nomi di dominio in indirizzi IP. In questi pacchetti, possiamo vedere delle richieste DNS per risolvere il nome di dominio www.download.windowsupdate.com:

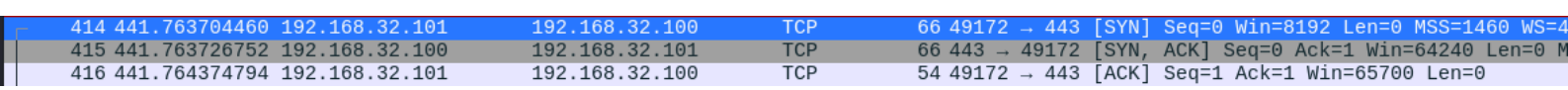
·Fase 1 (Richiesta DNS): Il client (192.168.32.101) invia una richiesta DNS al server DNS (192.168.32.100) per risolvere il nome di dominio epicode.internal. La richiesta è per un record A (IPv4).
·Fase 2 (Risposta DNS per IPv4): Il server DNS risponde con l'indirizzo IPv4 associato al dominio, ovvero 192.168.32.100.
·Fase 3 (Tentativo di risoluzione IPv6): Successivamente, il client invia una nuova richiesta DNS, questa volta per un record AAAA (IPv6), cercando di risolvere un possibile indirizzo IPv6 per epicode.internal. Tuttavia, non riceve una risposta.



THREE-WAY HANDSHAKE

Qui avviene lo scambio tra i pacchetti TCP al livello 4 standard ISO/OSI, per stabilire la connessione tra client(192.168.32.101) e server (192.168.32.100) in modo sicuro la sequenza avviene cosi:

·SYN (pacchetto 414): Il client invia un pacchetto SYN per avviare la connessione.
·SYN-ACK (pacchetto 415): Il server risponde con un pacchetto SYN-ACK per confermare la richiesta di connessione.
·ACK (pacchetto 416): Il client conferma la connessione con un pacchetto ACK.

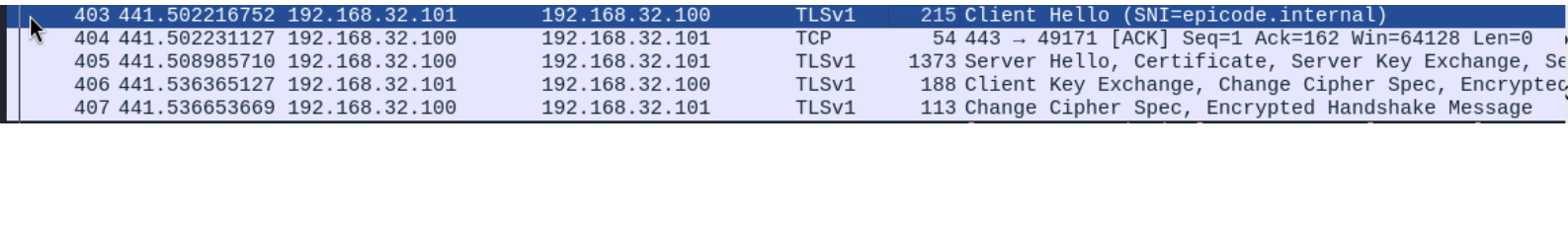


TLS HANDSHAKE (CRITTOGRAFIA)

Una volta che la connessione TCP è stabilita, inizia il processo di negoziazione TLS (Transport Layer Security), che è il protocollo utilizzato per proteggere i dati durante la trasmissione tra client e server.

Fasi del TLS Handshake:

- **Client Hello** (pacchetto 403): Il client invia un pacchetto "Client Hello" per avviare il handshake TLS. Questo pacchetto contiene informazioni importanti, come il nome del dominio (SNI) e le suite di cifratura supportate.
- **Server Hello e Certificato** (pacchetto 405): Il server risponde con un "Server Hello", il proprio certificato SSL, e altre informazioni necessarie per stabilire una connessione sicura. Il certificato SSL contiene la chiave pubblica del server, che verrà utilizzata per la crittografia dei dati.
- **Key Exchange** (pacchetto 406): Dopo che il server ha inviato il suo certificato, il client esegue un "Client Key Exchange" per inviare la chiave segreta necessaria per stabilire una connessione sicura. A questo punto, entrambe le parti concordano su una chiave segreta condivisa che verrà utilizzata per cifrare e decifrare i dati durante la sessione.
- **Change Cipher Spec** (pacchetto 407 e 421): Entrambe le parti inviano pacchetti per informare l'altra che la crittografia è ora attiva e che il resto della comunicazione avverrà in modo sicuro.
- **Encrypted Handshake** (pacchetto 406, 407): A questo punto, la negoziazione TLS è completata e i dati sono cifrati, il che significa che non è possibile leggere il contenuto dei pacchetti successivi senza la chiave segreta condivisa.



Operando in osservazione sulla porta eth0 della macchina virtuale di Kali possiamo osservare un normale processo di stabilimento di una connessione HTTPS, che include il Three-Way Handshake per stabilire la connessione TCP, seguito dal TLS Handshake per stabilire una connessione sicura e crittografata. I pacchetti che contengono i dati applicativi sono crittografati, quindi non sono visibili senza la chiave privata del server.

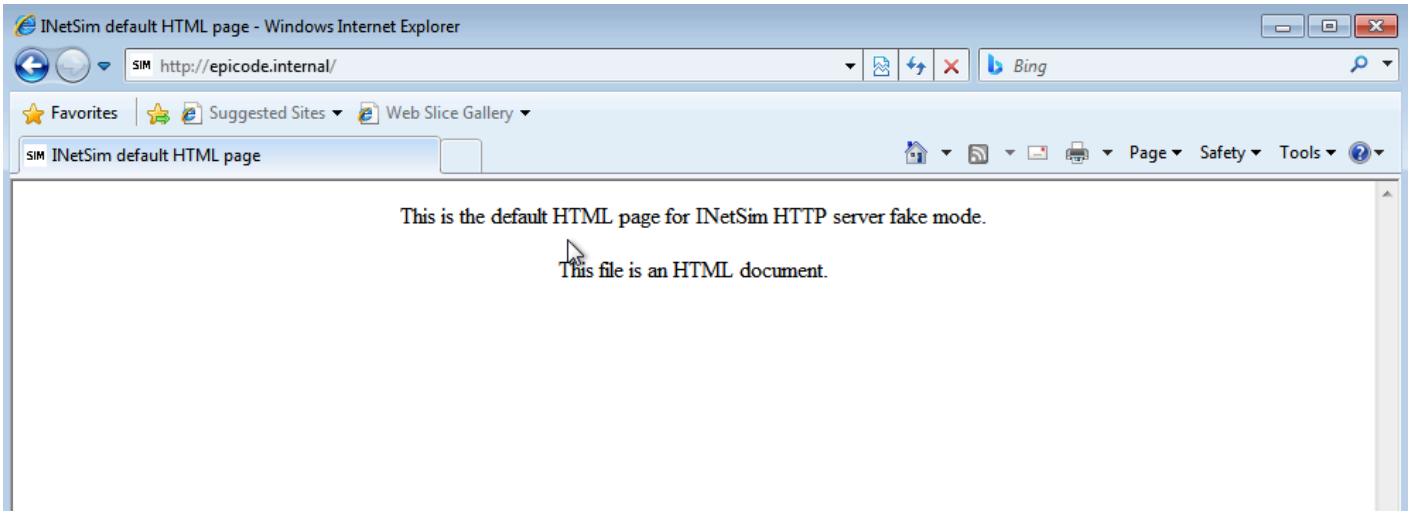
Fine prima parte

PRESENTAZIONE SVOLGIMENTO SECONDA RICHIESTA

In questa fase, ripeterò l'esercizio sostituendo il server HTTPS con un server HTTP. Questo mi permetterà di intercettare il traffico HTTP anziché quello HTTPS e di osservare le differenze tra i due.

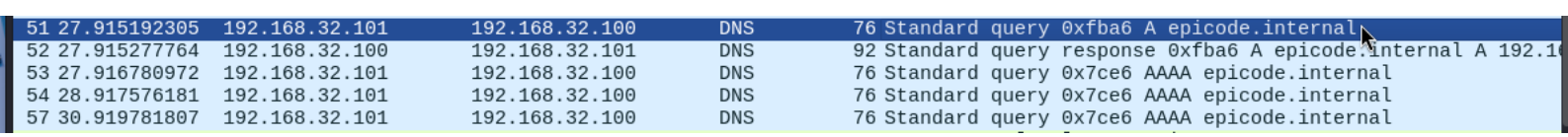
SVOLGIMENTO

Sul servizio INETSIM, come mostrato precedentemente sarà sufficiente commentare la riga utilizzata prima per il servizio HTTPS e decommentare quella con scritto HTTP, proprio per utilizzare un altro tipo di protocollo di cui osserveremo le differenze più avanti.

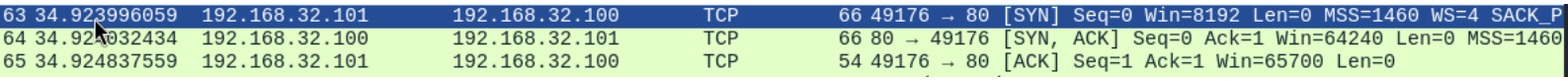


Una volta ottenuta la pagina andiamo ad analizzare le principali differenze nella fase di sniffino tra "HTTPS" e "HTTP"

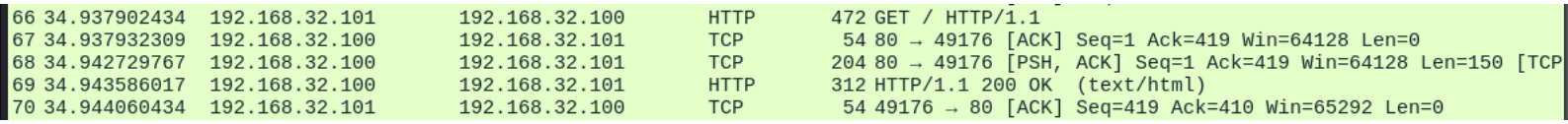
1. Richiesta DNS per "epicode.internal" (pacchetti 51-54):
   - Pacchetti 51-54: Il client (192.168.32.101) invia una richiesta DNS per risolvere il nome di dominio "epicode.internal" (tipo A e AAAA).
   - Pacchetto 52: Il server DNS (192.168.32.100) risponde con l'indirizzo IPv4 di "epicode.internal" 192.168.32.100).
   - Pacchetti 53-54: Il client invia ulteriori richieste per l'indirizzo IPv6 (AAAA).



2. Connessione TCP (Three-Way Handshake):
   - Pacchetto 63: Il client (192.168.32.101) invia un pacchetto SYN per stabilire la connessione TCP sulla porta 80.
   - Pacchetto 64: Il server (192.168.32.100) risponde con un pacchetto SYN-ACK.
   - Pacchetto 65: Il client invia un pacchetto ACK per completare il Three-Way Handshake.



3. Richiesta HTTP:
   - Pacchetto 66: Dopo aver stabilito la connessione TCP, il client invia una richiesta HTTP GET per la risorsa principale ("/") sul server (HTTP/1.1).
   - Pacchetto 67: Il server risponde con un pacchetto TCP di ACK.
   - Pacchetto 68: Il server invia un pacchetto PSH-ACK contenente la risposta HTTP (un documento HTML).
   - Pacchetto 69: Il server invia la risposta HTTP con codice di stato "200 OK" e il contenuto della risorsa richiesta (HTML).
   - Pacchetto 70: Il client invia un pacchetto TCP ACK per confermare la ricezione della risposta.



4. Chiusura della connessione TCP:
   - Pacchetto 71: Il client invia un pacchetto FIN-ACK per chiudere la connessione.
   - Pacchetto 72: Il server risponde con un pacchetto ACK per confermare la chiusura della connessione.

DIFFERENZE PRINCIPALI HTTPS & HTTP

1. **Connessione Sicura (HTTPS)**:

- HTTP: La comunicazione avviene in chiaro, senza crittografia, il che significa che tutti i dati (come username, password, ecc.) sono trasmessi senza alcuna protezione. La connessione avviene direttamente tra client e server senza alcun processo di handshake per la cifratura.
- HTTPS: La connessione è cifrata tramite SSL/TLS. La comunicazione avviene su una connessione sicura tra client e server, che viene stabilita con un processo di handshake SSL/TLS. Questo garantisce la riservatezza dei dati trasmessi.

2. **Processo di Handshake**:

- HTTP: Non c'è alcun handshake di crittografia, i dati vengono inviati "in chiaro".
- HTTPS: C'è un handshake SSL/TLS prima che la connessione HTTP possa essere utilizzata per trasferire i dati. Durante l'handshake, il server invia il suo certificato SSL/TLS al client, che verifica la validità del certificato e stabilisce una chiave di sessione segreta per cifrare la comunicazione.

3. **Porta di Comunicazione**:

- HTTP: La connessione avviene sulla porta 80.
- HTTPS: La connessione avviene sulla porta 443.

4. **Visibilità dei Dati**:

- HTTP: I dati sono visibili a chiunque intercetti il traffico. Ad esempio, se un attaccante esegue un attacco man-in-the-middle (MITM), può leggere o alterare i dati.
- HTTPS: I dati sono cifrati, quindi anche se qualcuno intercetta il traffico, non può leggere i dati (ad esempio, le credenziali o altre informazioni sensibili).

5. **Certificati SSL/TLS**:

- HTTP: Non sono utilizzati certificati SSL/TLS.
- HTTPS: Il server deve avere un certificato SSL/TLS valido che autentica il server e cifrano i dati.

GRAZIE PER LA VOSTRA ATTENZIONE

FABIO BELFORTI

EPICODE "CYBESECURITY ANALYST"