



W4D4 – PROGETTO FINALE

CORSO EPICODE

ESERCIZIO:

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).
2. Si intercetti poi la comunicazione con Wireshark, evidenziando i MAC address di sorgente e destinazione ed il contenuto della richiesta HTTPS.
3. 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.

CREAZIONE DEL SETUP:

Il primo step sarà la configurazione del client e del server con i seguenti indirizzi IP statici:

- Kali Linux (Server) – 192.168.32.100 255.255.255.0
- Windows 7 (Client) – 192.168.32.101 255.255.255.0
- HTTPS server: attivo
- Servizio DNS per risoluzione nomi di dominio: attivo

CONFIGURAZIONE DEGLI INDIRIZZI IP STATICI:

Dopo aver configurato le schede di rete delle due macchine virtuali come “rete interna“, procedere con la configurazione delle schede di rete dei due PC. Per verificare che la configurazione sia avvenuta correttamente, digitare nel terminale di Windows “ipconfig /all“ e nel terminale di Kali Linux “ifconfig“. Il risultato dovrebbe corrispondere a quello mostrato negli screenshot qui sotto:

```
Media Ethernet Connessione alla rete locale (LAN):
Suffisso DNS specifico per connessione:
Descrizione . . . . . : Scheda desktop Intel(R) PRO/1000 MT
Indirizzo fisico . . . . . : 08-00-27-32-12-5A
DHCP abilitato . . . . . : No
Configurazione automatica abilitata . . . . . : Sì
Indirizzo IPv6 locale rispetto al collegamento . : fe80::6cba:413c:1b6f:243a%11<Preferenziale>
Indirizzo IPv4 . . . . . : 192.168.32.101<Preferenziale>
Subnet mask . . . . . : 255.255.255.0
Gateway predefinito . . . . . : 192.168.32.1
IAID DHCPv6 . . . . . : 235405351
DUID Client DHCPv6 . . . . . : 00-01-00-01-2F-50-55-CF-08-00-27-32-12-5A

Server DNS . . . . . : fec0:0:0:ffff::1%1
                       fec0:0:0:ffff::2%1
                       fec0:0:0:ffff::3%1
NetBIOS su TCP/IP . . . . . : Attivato

Scheda Tunnel isatap.<DEB67373-6CF3-4A0D-90C2-BDAEF1761B8C>:
Stato supporto . . . . . : Supporto disconnesso
Suffisso DNS specifico per connessione:
Descrizione . . . . . : Microsoft ISATAP Adapter
Indirizzo fisico . . . . . : 00-00-00-00-00-00-E0
DHCP abilitato . . . . . : No
Configurazione automatica abilitata . . . . . : Sì
```

Windows 7

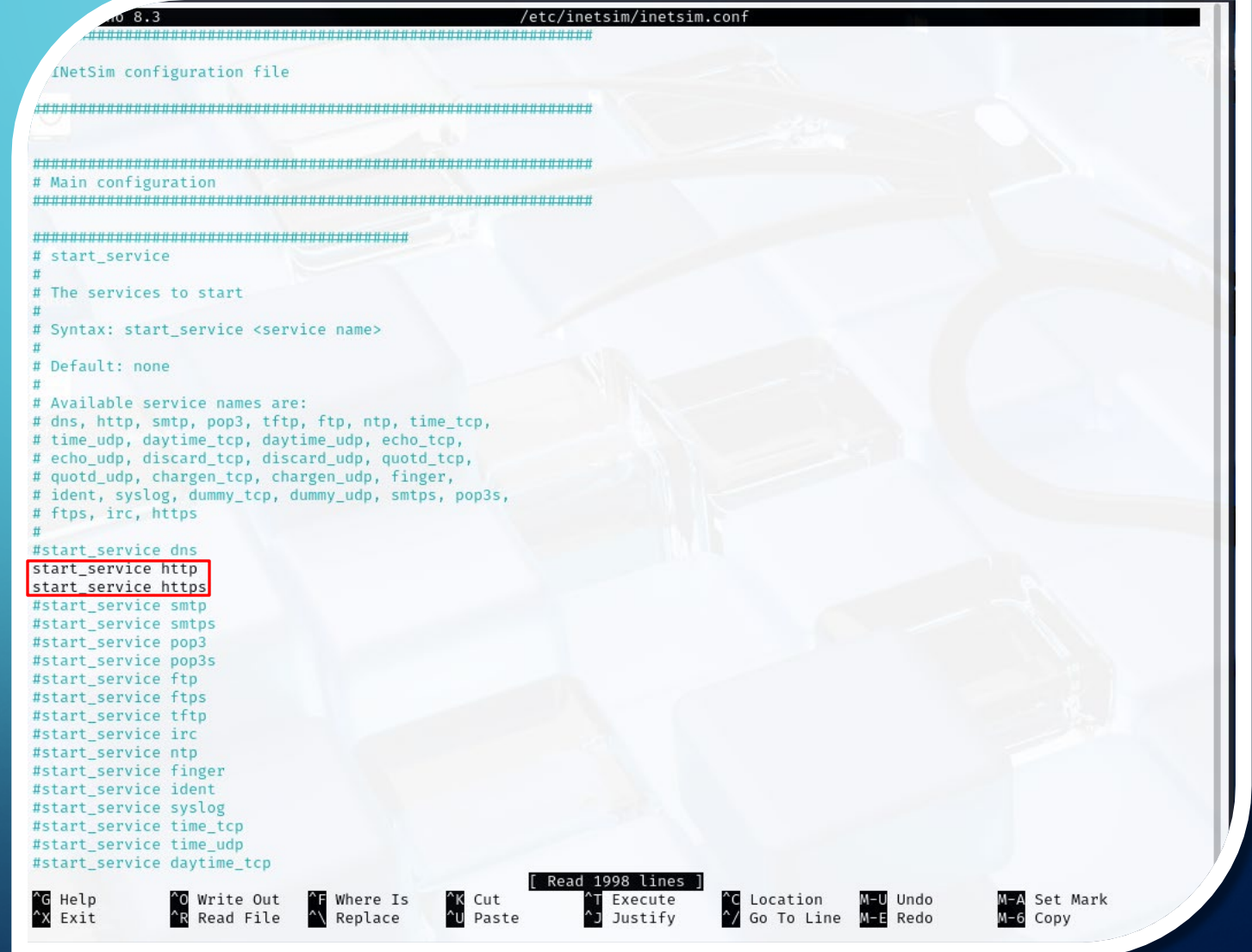
```
-(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.32.100 netmask 255.255.255.0 broadcast 192.168.32.255
    inet6 fe80::a00:27ff:fe6e:136e prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:6e:13:6e txqueuelen 1000 (Ethernet)
    RX packets 830 bytes 65639 (64.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 217 bytes 33106 (32.3 KiB)
    TX errors 0 dropped 19 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 8 bytes 480 (480.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 480 (480.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Kali Linux

CONFIGURAZIONE HTTP E HTTPS:

Passare ora alla configurazione di INetSim su Kali Linux digitando “sudo nano /etc/inetsim/inetsim.conf” per modificare la configurazione del file “inetsim.conf”. Bisognerà successivamente abilitare “start_service http” e “start_service https” rimuovendo gli hashtag che li precedono.



```
8.3 /etc/inetsim/inetsim.conf
#####
#NetSim 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
#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
#start_service time_tcp
#start_service time_udp
#start_service daytime_tcp
```

CONFIGURAZIONE HTTP E HTTPS:

Sarà necessario impostare “service_bind_address” con l'indirizzo IP della macchina che esegue Kali Linux.

Se non configurato correttamente, il valore predefinito sarà 127.0.0.1 (loopback), e in questo caso il servizio HTTPS non sarà raggiungibile dalla macchina Windows.

```
Mano 8.3 /etc/inetsim/inetsim.conf
start_service time_tcp /etc/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

#####
# service_run_as_user
#
# User to run services
#
# Syntax: service_run_as_user <username>
#
# Default: inetsim
#
#service_run_as_user nobody

#####
# service_max_childs
#
# Maximum number of child processes (parallel connections)
# for each service
#
```

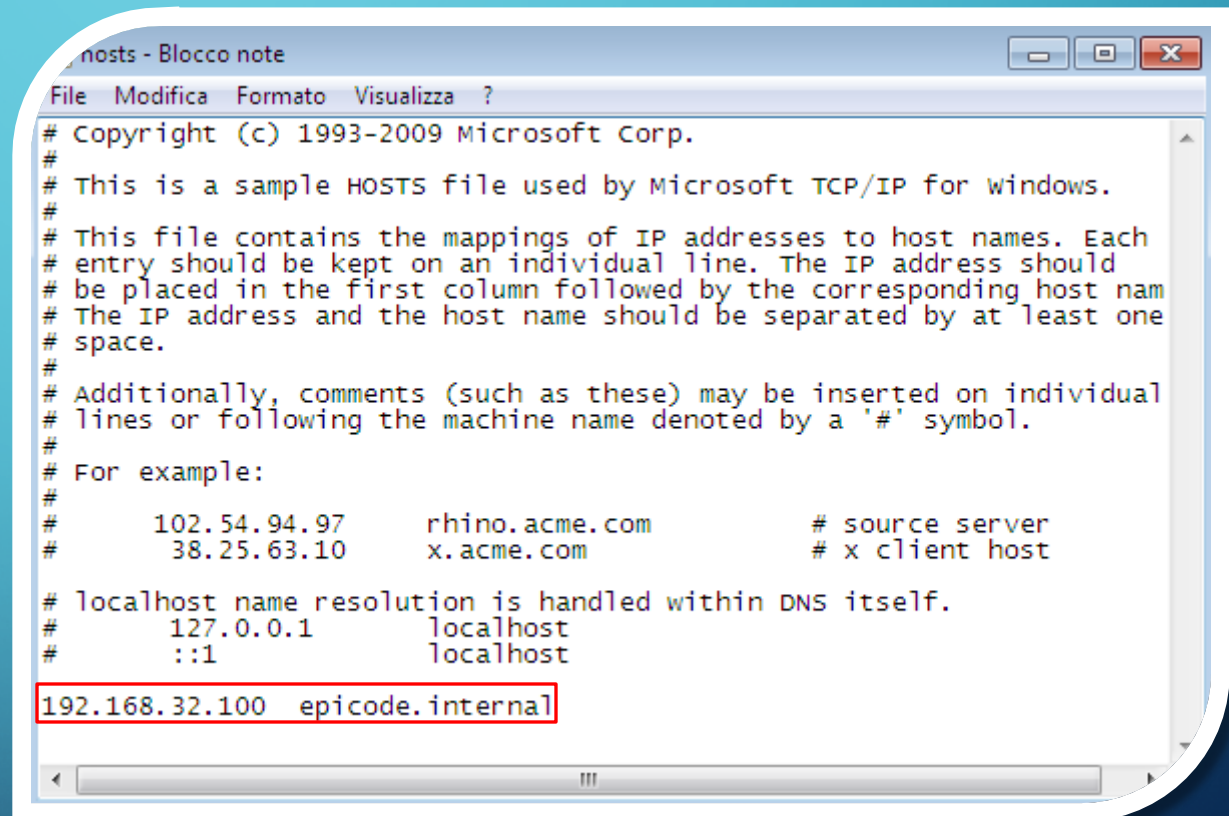
^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark
^X Exit ^R Read File ^N Replace ^U Paste ^J Justify ^_ Go To Line M-E Redo M-G Copy

CONFIGURAZIONE DNS:

Poiché su INetSim la configurazione dei DNS non funziona con la versione 2025.1, una soluzione può essere configurare manualmente il file “hosts”. È necessario aprire il Blocco note come amministratore e selezionare “File” e “Apri...”.

Successivamente, inserire la directory

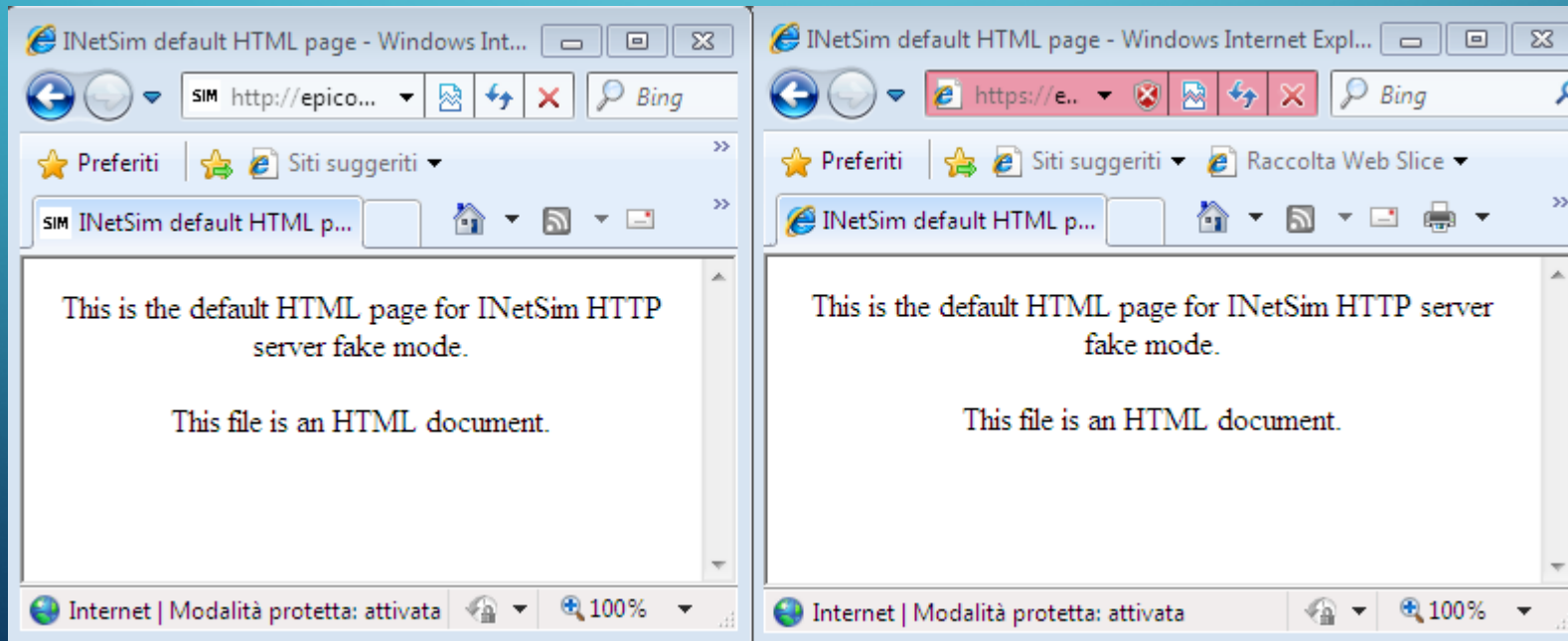
“C:\Windows\System32\drivers\etc”. A questo punto, aggiungere manualmente l’IP di Kali Linux (192.168.32.100) e il dominio che si desidera impostare (epicode.internal).



```
# hosts - Blocco note
File Modifica Formato Visualizza ?
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host nam
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com       # source server
#       38.25.63.10       x.acme.com           # x client host
#
# localhost name resolution is handled within DNS itself.
#       127.0.0.1         localhost
#       ::1               localhost
192.168.32.100 epicode.internal
```

TEST DELLA CONNESSIONE:

Dopo aver avviato INetSim su Kali Linux con il comando "sudo inetsim", aprire Internet Explorer su Windows e digitare <http://epicode.internal/> e <https://epicode.internal/>. Nel seguente screenshot, a sinistra è mostrato il collegamento alla pagina HTTP, mentre a destra quello alla pagina HTTPS.



SNIFFING HTTPS CON WIRESHARK:

I protocolli utilizzati sono: ARP, NBNS, TCP, TLSv1 e LLMNR.

29	459.849871066	PCSSystemtec_6e:13:...	PCSSystemtec_32:12:...	ARP	42	192.168.32.100 is at 08:00:27:6e:13:6e
30	459.850293187	192.168.32.101	192.168.32.100	TCP	66	49211 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
31	459.850315207	192.168.32.100	192.168.32.101	TCP	66	443 → 49211 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
32	459.850688885	192.168.32.101	192.168.32.100	TCP	60	49211 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
33	459.851920463	192.168.32.101	192.168.32.100	TLSv1	183	Client Hello (SNI=epicode.internal)
34	459.851931055	192.168.32.100	192.168.32.101	TCP	54	443 → 49211 [ACK] Seq=1 Ack=130 Win=64128 Len=0
35	459.876040679	192.168.32.100	192.168.32.101	TLSv1	1373	Server Hello, Certificate, Server Key Exchange, Server Hello Done
36	459.897562070	192.168.32.101	192.168.32.100	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
37	459.898152798	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
38	459.923360607	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
39	460.096639970	192.168.32.101	192.168.32.100	TCP	60	49211 → 443 [ACK] Seq=264 Ack=1379 Win=64320 Len=0
40	460.704846124	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
41	461.705962241	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
42	463.040525489	fe80::6cba:413c:1b6...	ff02::1:3	LLMNR	84	Standard query 0x735e A wpad
43	463.040575069	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x735e A wpad
44	463.142545324	fe80::6cba:413c:1b6...	ff02::1:3	LLMNR	84	Standard query 0x735e A wpad
45	463.142545471	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x735e A wpad
46	463.346393187	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
47	464.095983794	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
48	464.846893518	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
49	465.108184252	PCSSystemtec_6e:13:...	PCSSystemtec_32:12:...	ARP	42	Who has 192.168.32.101? Tell 192.168.32.100
50	465.108606874	PCSSystemtec_32:12:...	PCSSystemtec_6e:13:...	ARP	60	192.168.32.101 is at 08:00:27:32:12:5a
51	465.600869541	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
52	466.205635860	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
53	467.206530043	PCSSystemtec_32:12:...	Broadcast	ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
54	468.740992022	fe80::6cba:413c:1b6...	ff02::1:3	LLMNR	84	Standard query 0x0c8e A wpad
55	468.741074962	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x0c8e A wpad
56	468.846851023	fe80::6cba:413c:1b6...	ff02::1:3	LLMNR	84	Standard query 0x0c8e A wpad
57	468.847066487	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x0c8e A wpad
58	469.050139703	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
59	469.800691894	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
60	470.550389331	192.168.32.101	192.168.32.255	NBNS	92	Name query NB WPAD<00>
61	471.304105918	192.168.32.101	192.168.32.100	TCP	60	49211 → 443 [FIN, ACK] Seq=264 Ack=1379 Win=64320 Len=0
62	471.304272908	192.168.32.100	192.168.32.101	TLSv1	91	Encrypted Alert
63	471.304593401	192.168.32.101	192.168.32.100	TCP	60	49211 → 443 [RST, ACK] Seq=265 Ack=1416 Win=0 Len=0

SNIFFING HTTPS CON WIRESHARK:

- **ARP:**
Utilizzato per risolvere un indirizzo IP in un indirizzo MAC. È necessario per entrambe le connessioni HTTP e HTTPS, ma non dipende dal protocollo.
- **NBNS:**
Utilizzato per risolvere i nomi di dispositivi su una rete locale. Questo processo è indipendente da HTTP o HTTPS, ma può essere utile per entrambe le connessioni.
- **TCP:**
Gestisce la comunicazione tra dispositivi, garantendo che i dati arrivino correttamente. La connessione viene stabilita tramite il 3-way handshake (SYN, SYN-ACK, ACK). Questo processo è valido sia per HTTP che per HTTPS.
- **TLSv1 (Transport Layer Security):**
Utilizzato in HTTPS per garantire una comunicazione sicura tra client e server. TLS cifra i dati scambiati per proteggere la privacy e l'integrità delle informazioni. La connessione sicura viene stabilita attraverso un handshake simile a quello di TCP, ma con fasi aggiuntive per negoziare i parametri di sicurezza, come il tipo di cifratura.
- **LLMNR (Link-Local Multicast Name Resolution):**
Utilizzato in reti locali per risolvere i nomi dei dispositivi senza fare affidamento su un server DNS. Può comparire nei pacchetti di rete quando un dispositivo cerca di risolvere un nome che non può essere risolto tramite il DNS configurato. Sebbene non sia legato direttamente a HTTP o HTTPS, può essere utilizzato se la macchina non riesce a risolvere un nome tramite il DNS e tenta di farlo in maniera locale.

SNIFFING HTTPS CON WIRESHARK:

È importante notare che i MAC address di sorgente e destinazione si invertono a ogni passaggio, riflettendo le fasi del 3-way handshake. Ad esempio, il MAC address della macchina Windows termina con 6E:13:6E, mentre quello di Kali Linux termina con 32:12:5A.

```
33 459.851920463 192.168.32.101 192.168.32.100 TLSv1 183 Client Hello (SNI=epicode.internal)
UTC Arrival Time: Mar 23, 2025 11:24:34.880120916 UTC
Epoch Arrival Time: 1742729074.880120916
[Time shift for this packet: 0.000000000 seconds]
[Time delta from previous captured frame: 0.001231578 seconds]
[Time delta from previous displayed frame: 0.001231578 seconds]
[Time since reference or first frame: 459.851920463 seconds]
Frame Number: 33
Frame Length: 183 bytes (1464 bits)
Capture Length: 183 bytes (1464 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ethertype:ip:tcp:tls]
[Coloring Rule Name: TCP]
[Coloring Rule String: tcp]
↳ Ethernet II, Src: PCSSystemtec_32:12:5a (08:00:27:32:12:5a), Dst: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Destination: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Source: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
Type: IPv4 (0x0800)
[Stream index: 1]
↳ Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
↳ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
```

```
34 459.876040679 192.168.32.101 192.168.32.100 TLSv1 1373 Server Hello, Certificate, Server Key Exchange, Server Hello Done
UTC Arrival Time: Mar 23, 2025 11:24:34.904241132 UTC
Epoch Arrival Time: 1742729074.904241132
[Time shift for this packet: 0.000000000 seconds]
[Time delta from previous captured frame: 0.024109624 seconds]
[Time delta from previous displayed frame: 0.024109624 seconds]
[Time since reference or first frame: 459.876040679 seconds]
Frame Number: 35
Frame Length: 1373 bytes (10984 bits)
Capture Length: 1373 bytes (10984 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ethertype:ip:tcp:tls:x509sat:x509sat:x509sat:x509sat:x509sat:x509sat:x509ce:x509ce:x509ce]
[Coloring Rule Name: TCP]
[Coloring Rule String: tcp]
↳ Ethernet II, Src: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e), Dst: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
Destination: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
Source: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Type: IPv4 (0x0800)
[Stream index: 1]
↳ Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
0100 .... = Version: 4
```

```
36 459.897562070 192.168.32.101 192.168.32.100 TLSv1 188 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
UTC Arrival Time: Mar 23, 2025 11:24:34.925762523 UTC
Epoch Arrival Time: 1742729074.925762523
[Time shift for this packet: 0.000000000 seconds]
[Time delta from previous captured frame: 0.021521391 seconds]
[Time delta from previous displayed frame: 0.021521391 seconds]
[Time since reference or first frame: 459.897562070 seconds]
Frame Number: 36
Frame Length: 188 bytes (1504 bits)
Capture Length: 188 bytes (1504 bits)
[Frame is marked: False]
[Frame is ignored: False]
[Protocols in frame: eth:ethertype:ip:tcp:tls]
[Coloring Rule Name: TCP]
[Coloring Rule String: tcp]
↳ Ethernet II, Src: PCSSystemtec_32:12:5a (08:00:27:32:12:5a), Dst: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Destination: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
Source: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
Type: IPv4 (0x0800)
[Stream index: 1]
↳ Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
↳ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
```


SNIFFING HTTPS CON WIRESHARK:

È importante notare che nel Transport Layer Security (TLS) con HTTPS, i pacchetti sono criptati, cosa che non avviene con HTTP. Inoltre, come detto in precedenza, la connessione sicura viene stabilita attraverso un handshake simile a quello di TCP, ma con fasi aggiuntive per negoziare i parametri di sicurezza, come il tipo di cifratura.

37 459.898152798 192.168.32.100 192.168.32.101 TLSv1 113 Change Cipher Spec, Encrypted Handshake Message

```
▶ Frame 37: 113 bytes on wire (904 bits), 113 bytes captured (904 bits) on interface eth0, id 0
▶ Ethernet II, Src: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e), Dst: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
▶ Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
▶ Transmission Control Protocol, Src Port: 443, Dst Port: 49211, Seq: 1320, Ack: 264, Len: 59
▼ Transport Layer Security
  ▼ TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
    Content Type: Change Cipher Spec (20)
    Version: TLS 1.0 (0x0301)
    Length: 1
    Change Cipher Spec Message
  ▼ TLSv1 Record Layer: Handshake Protocol: Encrypted Handshake Message
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 48
    Handshake Protocol: Encrypted Handshake Message
```

SNIFFING HTTP CON WIRESHARK:

I protocolli utilizzati sono: ARP, NBNS, TCP e HTTP.

1	0.000000000	PCSSystemtec_32:12:...	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101
2	0.720504863	PCSSystemtec_32:12:...	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101
3	1.720395831	PCSSystemtec_32:12:...	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101
4	3.446253237	192.168.32.101	192.168.32.255	NBNS	92 Name query NB WWW.BING.COM<00>
5	4.189413102	192.168.32.101	192.168.32.255	NBNS	92 Name query NB WWW.BING.COM<00>
6	4.939406045	192.168.32.101	192.168.32.255	NBNS	92 Name query NB WWW.BING.COM<00>
7	28.224873814	PCSSystemtec_32:12:...	Broadcast	ARP	60 Who has 192.168.32.100? Tell 192.168.32.101
8	28.224884239	PCSSystemtec_6e:13:...	PCSSystemtec_32:12:...	ARP	42 192.168.32.100 is at 08:00:27:6e:13:6e
9	28.225237826	192.168.32.101	192.168.32.100	TCP	66 49210 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
10	28.225257135	192.168.32.100	192.168.32.101	TCP	66 80 → 49210 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
11	28.225570506	192.168.32.101	192.168.32.100	TCP	60 49210 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
12	28.225570603	192.168.32.101	192.168.32.100	HTTP	444 GET / HTTP/1.1
13	28.225604543	192.168.32.100	192.168.32.101	TCP	54 80 → 49210 [ACK] Seq=1 Ack=391 Win=64128 Len=0
14	28.246268216	192.168.32.100	192.168.32.101	TCP	204 80 → 49210 [PSH, ACK] Seq=1 Ack=391 Win=64128 Len=150 [TCP PDU reassembled in 15]
15	28.248981652	192.168.32.100	192.168.32.101	HTTP	312 HTTP/1.1 200 OK (text/html)
16	28.249334183	192.168.32.101	192.168.32.100	TCP	60 49210 → 80 [ACK] Seq=391 Ack=410 Win=65292 Len=0
17	28.249480526	192.168.32.101	192.168.32.100	TCP	60 49210 → 80 [FIN, ACK] Seq=391 Ack=410 Win=65292 Len=0
18	28.249492252	192.168.32.100	192.168.32.101	TCP	54 80 → 49210 [ACK] Seq=410 Ack=392 Win=64128 Len=0
19	33.483454088	PCSSystemtec_6e:13:...	PCSSystemtec_32:12:...	ARP	42 Who has 192.168.32.101? Tell 192.168.32.100
20	33.483891248	PCSSystemtec_32:12:...	PCSSystemtec_6e:13:...	ARP	60 192.168.32.101 is at 08:00:27:32:12:5a

SNIFFING HTTP CON WIRESHARK:

- **ARP:**
Utilizzato per risolvere un indirizzo IP in un indirizzo MAC. È necessario per entrambe le connessioni HTTP e HTTPS, ma non dipende dal protocollo.
- **NBNS:**
Utilizzato per risolvere i nomi di dispositivi su una rete locale. Questo processo è indipendente da HTTP o HTTPS, ma può essere utile per entrambe le connessioni.
- **TCP:**
Gestisce la comunicazione tra dispositivi, garantendo che i dati arrivino correttamente. La connessione viene stabilita tramite il 3-way handshake (SYN, SYN-ACK, ACK). Questo processo è valido sia per HTTP che per HTTPS.
- **HTTP:**
Protocollo utilizzato per la comunicazione tra client e server. In Wireshark, si possono vedere richieste come GET / HTTP/1.1 e risposte come HTTP/1.1 200 OK. Il codice 200 OK indica che la richiesta è stata completata con successo.

SNIFFING HTTP CON WIRESHARK:

È importante notare come il traffico di pacchetti non è criptato con HTTP, a differenza di HTTPS, rendendo possibile visualizzare il contenuto di ciascuna richiesta inviata dal client al web server.

```
12 28.225570603 192.168.32.101 192.168.32.100 HTTP 444 GET / HTTP/1.1
13 28.225604543 192.168.32.100 192.168.32.101 TCP 54 80 → 49210 [ACK] Seq=1 Ack=391 Win=64128 Len=0
14 28.246268216 192.168.32.100 192.168.32.101 TCP 204 80 → 49210 [PSH, ACK] Seq=1 Ack=391 Win=64128 Len=150 [TCP PDU reassemble
15 28.248981652 192.168.32.100 192.168.32.101 HTTP 312 HTTP/1.1 200 OK (text/html)
16 28.249334183 192.168.32.101 192.168.32.100 TCP 60 49210 → 80 [ACK] Seq=391 Ack=410 Win=65292 Len=0
17 28.249480526 192.168.32.101 192.168.32.100 TCP 60 49210 → 80 [FIN, ACK] Seq=391 Ack=410 Win=65292 Len=0
18 28.249492252 192.168.32.100 192.168.32.101 TCP 54 80 → 49210 [ACK] Seq=410 Ack=392 Win=64128 Len=0
19 33.483454088 PCSSystemtec_6e:13: PCSSystemtec_32:12: ARP 42 Who has 192.168.32.101? Tell 192.168.32.100
20 33.483891248 PCSSystemtec_32:12: PCSSystemtec_6e:13: ARP 60 192.168.32.101 is at 08:00:27:32:12:5a
21 164.664159228 fe80::6cba:413c:1b6... ff02::1:2 DHCPv6 146 Solicit XID: 0x908898 CID: 000100012f5055cf08002732125a

> Frame 12: 444 bytes on wire (3552 bits), 444 bytes captured (3552 bits) on interface eth0, id 0
> Ethernet II, Src: PCSSystemtec_32:12:5a (08:00:27:32:12:5a), Dst: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e)
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49210, Dst Port: 80, Seq: 1, Ack: 1, Len: 390
> Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
  Accept: image/jpeg, application/x-ms-application, image/gif, application/xaml+xml, image/pjpeg, application/x-ms-xbap, */*\r\n
  Accept-Language: it-IT\r\n
  User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729)\r\n
  Accept-Encoding: gzip, deflate\r\n
  Host: epicode.internal\r\n
  Connection: Keep-Alive\r\n
  \r\n
  [Response in frame: 15]
  [Full request URI: http://epicode.internal/]
```

```
15 28.248981652 192.168.32.100 192.168.32.101 HTTP 312 HTTP/1.1 200 OK (text/html)
16 28.249334183 192.168.32.101 192.168.32.100 TCP 60 49210 → 80 [ACK] Seq=391 Ack=410
17 28.249480526 192.168.32.101 192.168.32.100 TCP 60 49210 → 80 [FIN, ACK] Seq=391 Ac
18 28.249492252 192.168.32.100 192.168.32.101 TCP 54 80 → 49210 [ACK] Seq=410 Ack=392
19 33.483454088 PCSSystemtec_6e:13: PCSSystemtec_32:12: ARP 42 Who has 192.168.32.101? Tell 192
20 33.483891248 PCSSystemtec_32:12: PCSSystemtec_6e:13: ARP 60 192.168.32.101 is at 08:00:27:32
21 164.664159228 fe80::6cba:413c:1b6... ff02::1:2 DHCPv6 146 Solicit XID: 0x908898 CID: 00010

> Frame 15: 312 bytes on wire (2496 bits), 312 bytes captured (2496 bits) on interface eth0, id 0
> Ethernet II, Src: PCSSystemtec_6e:13:6e (08:00:27:6e:13:6e), Dst: PCSSystemtec_32:12:5a (08:00:27:32:12:5a)
> Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
> Transmission Control Protocol, Src Port: 80, Dst Port: 49210, Seq: 151, Ack: 391, Len: 258
> [2 Reassembled TCP Segments (408 bytes): #14(150), #15(258)]
> Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
  Connection: Close\r\n
  Date: Sun, 23 Mar 2025 11:17:23 GMT\r\n
  > Content-Length: 258\r\n
  Server: INetSim HTTP Server\r\n
  Content-Type: text/html\r\n
  \r\n
  [Request in frame: 12]
  [Time since request: 0.023411049 seconds]
  [Request URI: /]
  [Full request URI: http://epicode.internal/]
  File Data: 258 bytes
  > Line-based text data: text/html (10 lines)
```

DIFFERENZE TRA HTTP ED HTTPS:

Le principali differenze tra HTTP e HTTPS sono le seguenti:

1. Cifratura dei dati e sicurezza:

Con HTTP, i dati sono trasmessi in chiaro, mentre con HTTPS i dati sono cifrati tramite SSL/TLS, garantendo una maggiore sicurezza. In caso di attacchi MITM (Man in the Middle), una connessione HTTPS risulta più sicura.

2. Porta di comunicazione:

La porta utilizzata da HTTP è la 80, mentre HTTPS utilizza la porta 443, dedicata alle connessioni sicure.