

Epicode

17-11-2023

Paul Alarcon



Esercizio

Traccia e requisiti

Nell'esercizio di oggi metteremo insieme le competenze acquisite finora.
Lo studente verrà valutato sulla base della risoluzione al problema seguente.

Requisiti e servizi:

- Kali Linux ☐ IP 192.168.32.100
- Windows 7 ☐ IP 192.168.32.101
- HTTPS server: attivo
- Servizio DNS per risoluzione nomi di dominio: attivo

Traccia:

Simulare, in ambiente di laboratorio virtuale, un'architettura client server in cui un client con indirizzo 192.168.32.101 (Windows 7) 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.

2

Per raggiungere gli obiettivi, è necessario prima configurare le due macchine sulla rete. Per la macchina Kali Linux, che ha come IP 192.168.50.100/24, questo dovrà essere sostituito con quello indicato nell'esercizio, ovvero 192.168.32.100/24. Per quanto riguarda la macchina Windows 7, l'IP 192.168.50.101/24 dovrà essere cambiato in 192.169.32.101/24. Come da consegna, imposterò anche l'IP per le connessioni DNS sulla macchina Kali. Dopo la configurazione degli IP, si può passare alla configurazione dei vari servizi che la macchina Kali Linux dovrà mettere a disposizione per la macchina Windows 7. Per la configurazione dei servizi, userò il tool inetsim e, per concludere l'esercizio, farò un'analisi del traffico di rete con Wireshark.

Configurazione IP su una macchina Kali linux

Per configurare l'indirizzo IP su una macchina Linux, ho aperto il terminale e utilizzato i seguenti comandi:

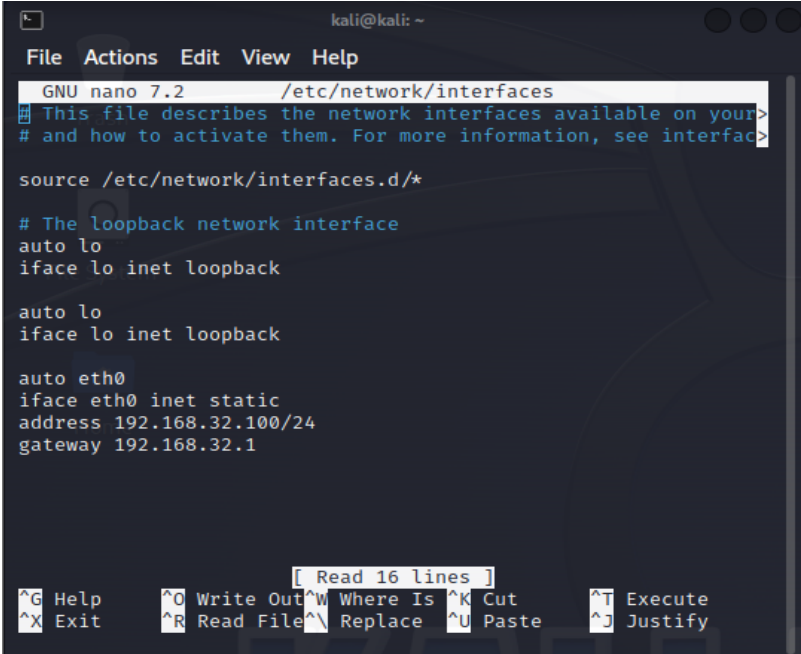
- `sudo nano /etc/network/interfaces`

Questo comando apre il file di configurazione di rete, all'interno del quale è possibile impostare l'indirizzo IP della macchina.

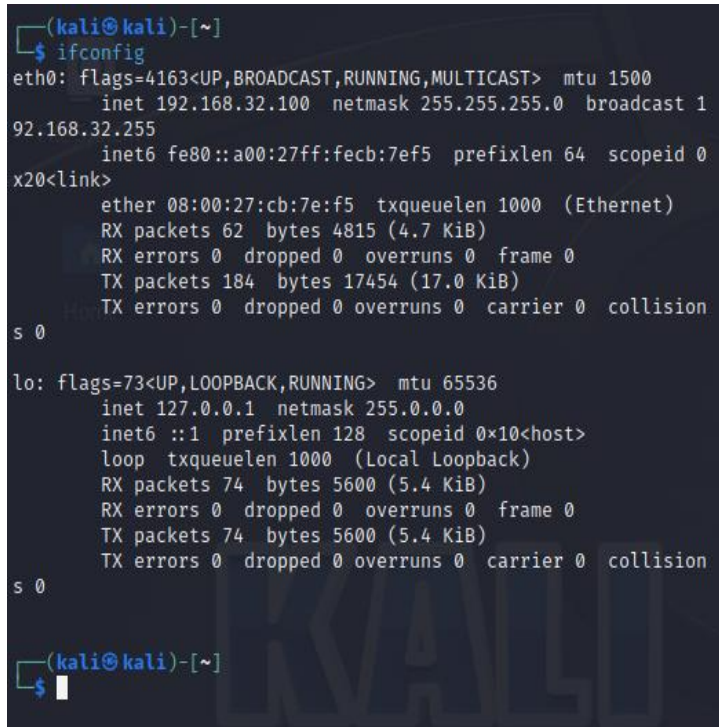
Dopo aver apportato le modifiche, è necessario riavviare l'interfaccia di rete per caricare le modifiche. Questo può essere fatto con il comando:

- `sudo systemctl restart networking` comando

- `Sudo systemctl restart networking`



```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/network/interfaces  
# This file describes the network interfaces available on your system  
# and how to activate them. For more information, see interfaces(5)  
  
source /etc/network/interfaces.d/*  
  
# The loopback network interface  
auto lo  
iface lo inet loopback  
  
auto lo  
iface lo inet loopback  
  
auto eth0  
iface eth0 inet static  
address 192.168.32.100/24  
gateway 192.168.32.1  
  
[ Read 16 lines ]  
^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  
^X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify
```



```
(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:feeb:7ef5 prefixlen 64 scopeid 0 x20<link>  
        ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)  
        RX packets 62 bytes 4815 (4.7 KiB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 184 bytes 17454 (17.0 KiB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collision s 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 74 bytes 5600 (5.4 KiB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 74 bytes 5600 (5.4 KiB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collision s 0  
  
(kali@kali)-[~]  
$
```

Per verificare se le modifiche sono state applicate correttamente, ho utilizzato il comando:

ifconfig

Questo comando restituisce le informazioni di configurazione di rete. Come si può vedere dall'immagine, il nuovo indirizzo IP è stato impostato correttamente.

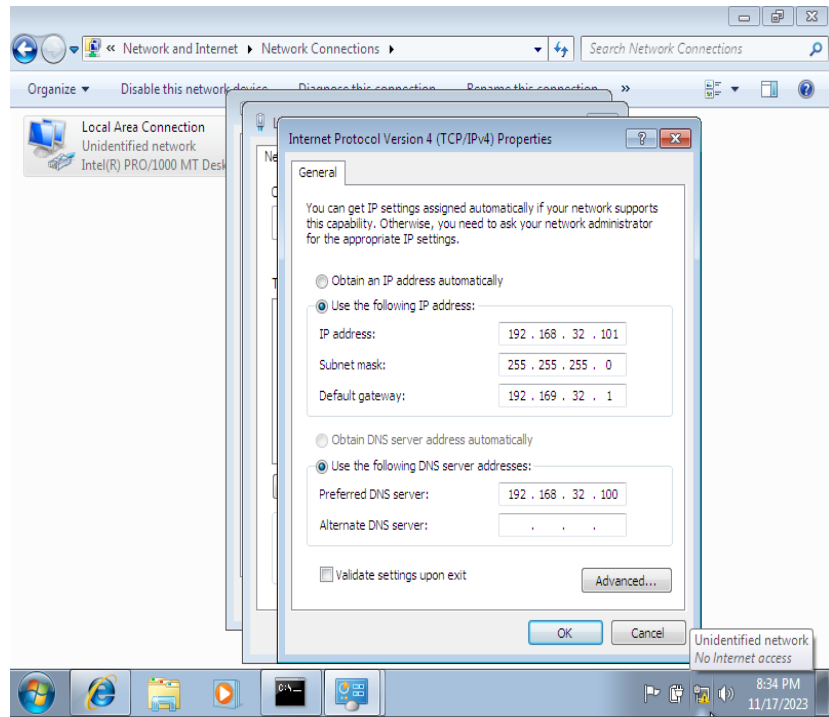
Configurazione IP su una macchina Windows 7

Impostare IP

Per impostare l'indirizzo IP della macchina **Windows 7**, ho acceduto al pannello di controllo, all'interno della configurazione di rete, e ho modificato l'IP del protocollo IPv4 nei connettori.

Nella stessa interfaccia, è possibile impostare il server per la connessione DNS. In questo campo, ho inserito l'IP della macchina **Kali Linux**, che conterrà il servizio DNS della rete.

Per concludere, ho disattivato il connettore e lo ho riattivato.



```
C:\Windows\system32\cmd.exe
Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\paul>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::deb0:ea55:147f:d49d%11
    IPv4 Address. . . . . : 192.168.32.101
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.169.32.1

Tunnel adapter isatap.{2E83C6EC-013B-4CF9-BE7B-E8A52BD30360}:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . : 

C:\Users\paul>SS
```

Controllo del nuovo IP

Con il comando `ipconfig`, hai confermato le modifiche. Dall'immagine, si può vedere che l'operazione è andata a buon fine. Ora puoi continuare con la configurazione dei servizi all'interno della macchina **Kali Linux**.

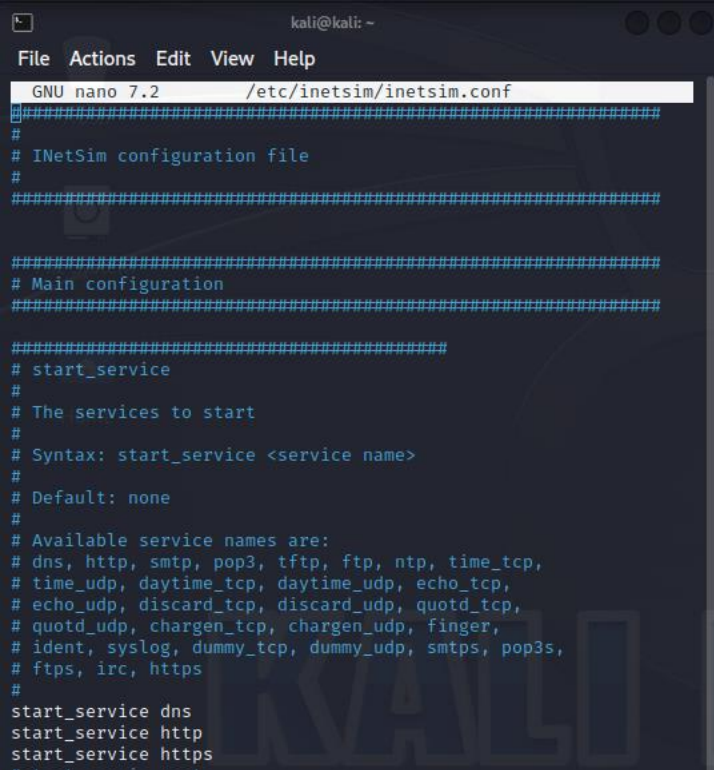
Configurazione di inetsim

Attivare i servizi

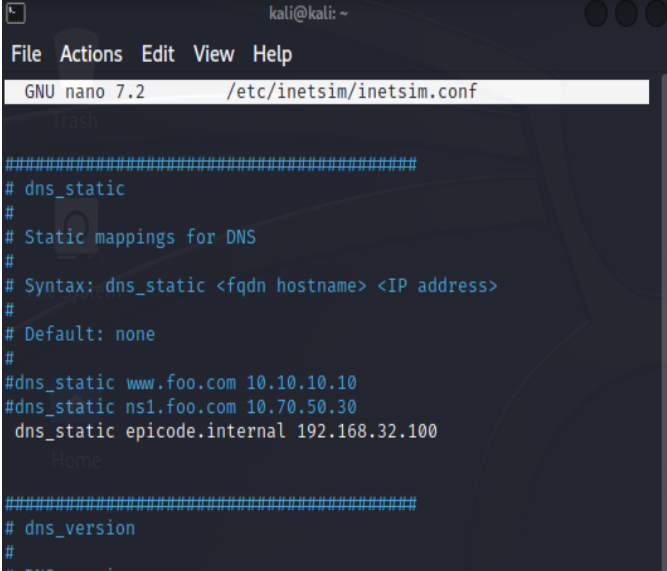
Per configurare inetsim, è necessario modificare il file inetsim.conf, abilitando i servizi richiesti dalla consegna (HTTPS, HTTP, DNS). Per accedere al file di configurazione di inetsim, userai il tool nano, che permette la lettura e la scrittura di file all'interno del terminale di Kali Linux. Puoi accedere al file di configurazione di inetsim con il seguente comando:

- `sudo nano /etc/inetsim/inetsim.conf`

All'interno del file di configurazione, troverai un elenco di servizi. Per questo esercizio, hai bisogno solo dei servizi DNS, HTTPS e HTTP. Per attivare questi servizi, basta rimuovere il '#' prima del comando 'start_service'.



```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /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
```



```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/inetsim/inetsim.conf  
#####  
# dns_static  
#  
# Static mappings for DNS  
#  
# Syntax: dns_static <fqdn hostname> <IP address>  
#  
# Default: none  
#  
#dns_static www.foo.com 10.10.10.10  
#dns_static ns1.foo.com 10.70.50.30  
dns_static epicode.internal 192.168.32.100  
#  
#####  
# dns_version  
#  
# DNS version
```

Configurazione DNS

I servizi HTTPS e HTTP sono pronti per l'uso. Per il servizio DNS, invece, è necessario impostare i domini che desideri associare all'interno della rete con un IP statico. In altre parole, devi indicare il nome della pagina e in quale macchina si può trovare il servizio. Nella rete che hai configurato, l'IP della macchina che contiene il servizio web coincide con quello del server DNS, quindi hai inserito l'IP dell'host Kali Linux.

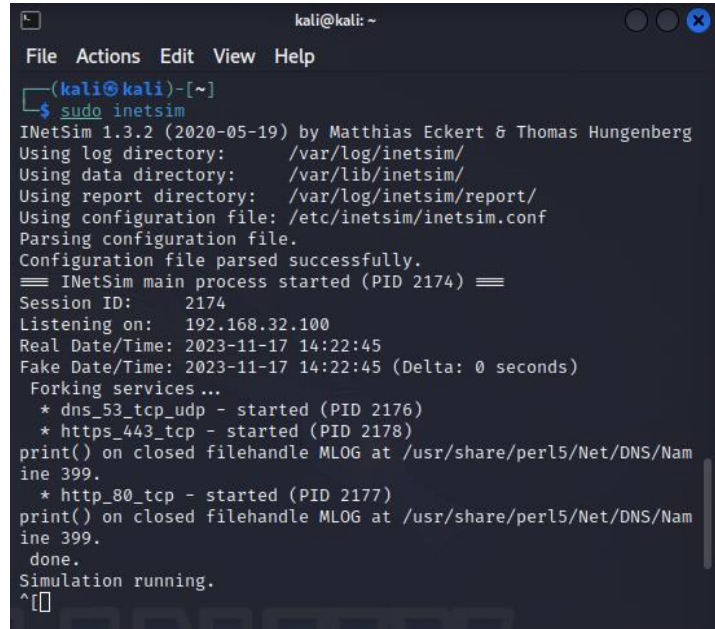
Per concludere, basta salvare il file e attivare Inetsim.

Attivaza Inetsim

Per attivare Inetsim, è sufficiente eseguire il comando:

- `sudo inetsim`

Sul terminale inizierà a visualizzare i dati della 'Simulation running' di Inetsim. A questo punto, puoi procedere a controllare la connessione su **Windows 7**.



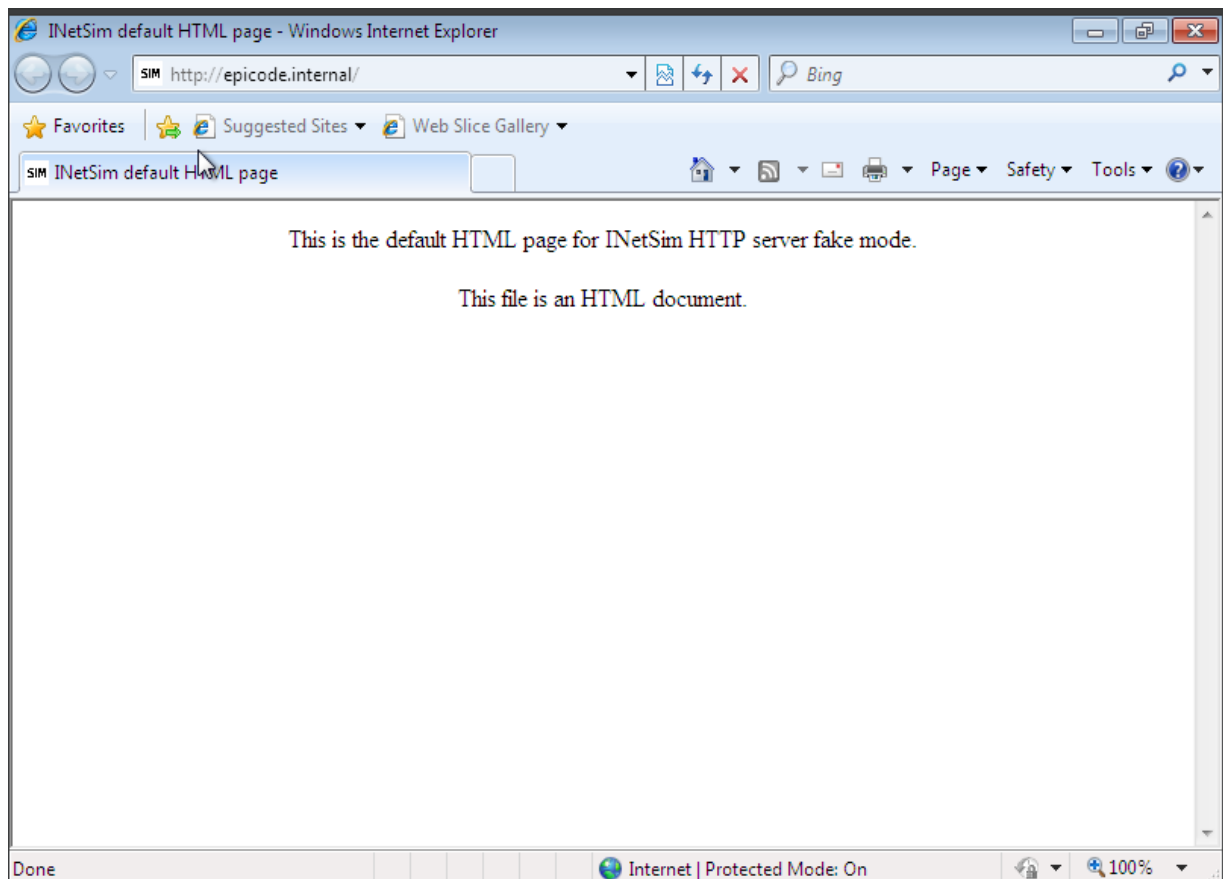
```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo inetsim  
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg  
Using log directory: /var/log/inetsim/  
Using data directory: /var/lib/inetsim/  
Using report directory: /var/log/inetsim/report/  
Using configuration file: /etc/inetsim/inetsim.conf  
Parsing configuration file.  
Configuration file parsed successfully.  
== INetSim main process started (PID 2174) ==  
Session ID: 2174  
Listening on: 192.168.32.100  
Real Date/Time: 2023-11-17 14:22:45  
Fake Date/Time: 2023-11-17 14:22:45 (Delta: 0 seconds)  
Forking services ...  
* dns_53_tcp_udp - started (PID 2176)  
* https_443_tcp - started (PID 2178)  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nam  
ine 399.  
* http_80_tcp - started (PID 2177)  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nam  
ine 399.  
done.  
Simulation running.  
^[]
```

Concessione e controllo del corretto funzionamento della rete.

Connessione http

Dopo aver configurato gli IP sulle macchine e i servizi Web e DNS sull'host Kali Linux, hai proceduto a fare una richiesta HTTP dal browser di Windows 7 con il dominio del servizio web ('epicode.internal' come richiesto).

Dall'immagine, si può vedere che la richiesta del servizio HTTP è andata a buon fine e il server DNS sta funzionando perfettamente perché è riuscito a interpretare il dominio 'epicode.internal'. Questo è un ottimo risultato!



Analisi della rete http

Utilizzando il tool Wireshark, è possibile visualizzare il traffico dei pacchetti e come il PC Windows 7 effettua una richiesta GET HTTP al server Kali. Questo è evidente dall'IP di origine (che coincide con quello del PC Kali Linux) e dall'IP destinatario (che coincide con quello del PC Windows 7).

È interessante notare che la comunicazione HTTP avviene in chiaro, mostrando tutti i dati della pagina. Questo è un aspetto importante da considerare per questioni di sicurezza e privacy.

The image shows a Wireshark network traffic capture. The top pane displays a list of captured packets. The middle pane shows the details of the selected packet (No. 6), which is an HTTP GET request. The bottom pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	PcsCompu_e4:5a:e6	192.168.32.100	ARP	62	Who has 192.168.32.100? Tell 192.168.32.101
2	0.000018579	PcsCompu_cb:7e:f5	192.168.32.100	ARP	44	192.168.32.100 is at 08:00:27:cb:7e:f5
3	0.000538878	192.168.32.101	192.168.32.100	TCP	68	49212 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
4	0.000570992	192.168.32.100	192.168.32.101	TCP	68	80 → 49212 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
5	0.001573832	192.168.32.101	192.168.32.100	TCP	62	49212 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
6	0.001574172	192.168.32.101	192.168.32.100	HTTP	356	GET / HTTP/1.1
7	0.001625349	192.168.32.100	192.168.32.101	TCP	56	80 → 49212 [ACK] Seq=1 Ack=301 Win=64128 Len=0
8	0.014311205	192.168.32.100	192.168.32.101	TCP	206	80 → 49212 [PSH, ACK] Seq=1 Ack=301 Win=64128 Len=150 [TCP segment of a reassembled PDU]
9	0.016230658	192.168.32.100	192.168.32.101	HTTP	314	HTTP/1.1 200 OK (text/html)
10	0.017354228	192.168.32.101	192.168.32.100	TCP	62	49212 → 80 [ACK] Seq=301 Ack=410 Win=65292 Len=0
11	0.017735960	192.168.32.101	192.168.32.100	TCP	62	49212 → 80 [FIN, ACK] Seq=301 Ack=410 Win=65292 Len=0
12	0.017753546	192.168.32.100	192.168.32.101	TCP	56	80 → 49212 [ACK] Seq=410 Ack=302 Win=64128 Len=0
13	5.164813324	PcsCompu_cb:7e:f5	192.168.32.101	ARP	44	Who has 192.168.32.101? Tell 192.168.32.100
14	5.165683131	PcsCompu_e4:5a:e6	192.168.32.101	ARP	62	192.168.32.101 is at 08:00:27:e4:5a:e6
15	177.662321016	PcsCompu_e4:5a:e6	192.168.32.101	ARP	62	Who has 192.168.32.100? Tell 192.168.32.101
16	177.662356742	PcsCompu_cb:7e:f5	192.168.32.101	ARP	44	192.168.32.100 is at 08:00:27:cb:7e:f5
17	177.662849942	192.168.32.101	192.168.32.100	DNS	78	Standard query 0xb8c7 A time.windows.com
18	177.679814228	192.168.32.100	192.168.32.101	DNS	94	Standard query response 0xb8c7 A time.windows.com A 127.0.0.1
19	182.828732540	PcsCompu_cb:7e:f5	192.168.32.100	ARP	44	Who has 192.168.32.101? Tell 192.168.32.100
20	182.829580719	PcsCompu_e4:5a:e6	192.168.32.101	ARP	62	192.168.32.101 is at 08:00:27:e4:5a:e6

Frame 6: 356 bytes on wire (2848 bits), 356 bytes captured (2848 bits) on interface a

Linux cooked capture v1

Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100

Transmission Control Protocol, Src Port: 49212, Dst Port: 80, Seq: 1, Ack: 1, Len: 30

Hypertext Transfer Protocol

GET / HTTP/1.1

Accept: */*

Accept-Language: en-US

User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; Accept-Encoding: gzip, deflate)

Host: epicode.internal

Connection: Keep-Alive

Full request URI: http://epicode.internal/

[HTTP request 1/1]

[Response in frame: 9]

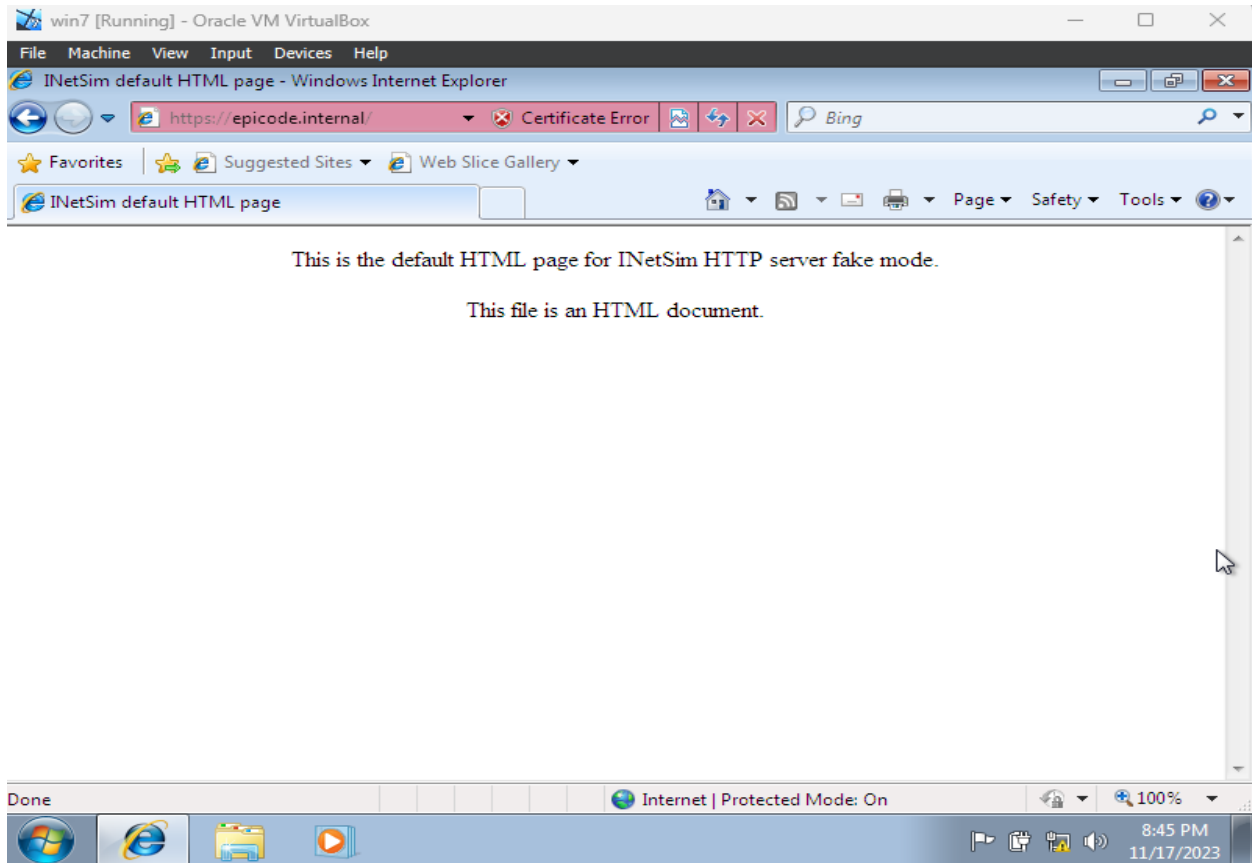
Bytes 274-305: Accept Encoding (http.accept_encoding)

Packets: 79 · Displayed: 79 (100.0%)

Profile: Default

Connessione HTTPS

Eseguito lo stesso test ma con il protocollo HTTPS, il browser riesce a connettersi anche in questo caso al dominio 'epicode.internal'. Si può notare che non trova un certificato valido, ma questo potrebbe essere dovuto al fatto che stiamo usando un servizio di provenienza ignota per il browser.



Analisi della rete HTTPS

Da Wireshark, si può notare che la situazione è simile a quella del protocollo HTTP. Ora, però, i pacchetti utilizzano il protocollo TCP e i dati non sono più in chiaro. Inoltre, ci sono più passaggi di connessione dovuti all'handshake tra le due macchine per impostare la chiave di crittografia. Questo è un aspetto fondamentale del protocollo HTTPS, che garantisce una comunicazione sicura tra client e server.

The image shows a Wireshark network capture of an HTTPS session. The packet list on the left shows several packets, with packet 141 selected. The packet details pane on the right shows the structure of the selected packet, which is an encrypted application data (TLSv1, Application Data). The packet bytes pane on the right shows the raw data of the packet, which is encrypted. The status bar at the bottom indicates that the payload is encrypted application data (tls.app_data), 432 bytes.

No.	Time	Source	Destination	Protocol	Length	Info
131	845.383883655	192.168.32.100	192.168.32.101	TLSv1	93	Encrypted Alert
132	845.3834492932	192.168.32.101	192.168.32.100	TCP	62	49230 → 443 [RST, ACK] Seq=297 Ack=1416 Win=0 Len=0
133	845.314634764	192.168.32.101	192.168.32.100	TCP	68	49230 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
134	845.314666901	192.168.32.100	192.168.32.101	TCP	68	443 → 49230 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
135	845.315169572	192.168.32.101	192.168.32.100	TCP	62	49230 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
136	845.316615534	192.168.32.101	192.168.32.100	TLSv1	217	Client Hello
137	845.316633622	192.168.32.100	192.168.32.101	TCP	56	443 → 49230 [ACK] Seq=1 Ack=162 Win=64128 Len=0
138	845.346269860	192.168.32.100	192.168.32.101	TLSv1	1375	Server Hello, Certificate, Server Key Exchange, Server Hello Done
139	845.357184843	192.168.32.101	192.168.32.100	TLSv1	190	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
140	845.357631224	192.168.32.100	192.168.32.101	TLSv1	115	Change Cipher Spec, Encrypted Handshake Message
141	845.362633170	192.168.32.101	192.168.32.100	TLSv1	432	Application Data
142	845.381169395	192.168.32.100	192.168.32.101	TLSv1	237	Application Data
143	845.384128287	192.168.32.100	192.168.32.101	TLSv1	386	Application Data, Encrypted Alert
144	845.385322522	192.168.32.101	192.168.32.100	TCP	62	49230 → 443 [ACK] Seq=733 Ack=1891 Win=65700 Len=0
145	845.386114317	192.168.32.101	192.168.32.100	TCP	62	49230 → 443 [FIN, ACK] Seq=733 Ack=1891 Win=65700 Len=0
146	845.386134216	192.168.32.100	192.168.32.101	TCP	56	443 → 49230 [ACK] Seq=1891 Ack=734 Win=64128 Len=0
147	845.501071490	192.168.32.101	192.168.32.100	TCP	68	49231 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
148	845.501132866	192.168.32.100	192.168.32.101	TCP	68	443 → 49231 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
149	845.502165127	192.168.32.101	192.168.32.100	TCP	62	49231 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
150	845.512376215	192.168.32.101	192.168.32.100	TLSv1	185	Client Hello

Frame 141: 493 bytes on wire (3944 bits), 493 bytes captured (3944 bits) on interface
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
Transmission Control Protocol, Src Port: 49230, Dst Port: 443, Seq: 296, Ack: 1379, L
Transport Layer Security
TLSv1 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
Content Type: Application Data (23)
Version: TLS 1.0 (0x0301)
Length: 432
Encrypted Application Data: a16cfbb7e7ab2f433eaa9accd3210d8564108533d3dee678b0d0
[Application Data Protocol: Hypertext Transfer Protocol]

Payload is encrypted application data (tls.app_data), 432 bytes

Packets: 170 · Displayed: 170 (100.0%)

Profile: Default

Indirizzo MAC

Un'informazione interessante che possiamo osservare all'interno del traffico di rete è l'indirizzo MAC della macchina che effettua la richiesta del servizio, in questo caso della macchina Windows.

Dall'immagine, si può notare l'indirizzo MAC della macchina Windows 7 all'interno dei pacchetti TCP per la connessione HTTPS. Questo dettaglio può essere utile per l'analisi del traffico di rete e per identificare specifici dispositivi all'interno della rete.

The screenshot displays a network traffic analysis using Wireshark. The main window shows a packet capture of an HTTPS connection. The packet list on the left shows a packet from 192.168.32.101 to 192.168.32.100. The packet details pane shows the source MAC address as 08:00:27:cb:7e:f5. The packet bytes pane shows the raw data of the packet.

Packet List:

No.	Time	Source	Destination	Protocol
131	845.303883655	192.168.32.100	192.168.32.101	TLSv1
132	845.304492932	192.168.32.101	192.168.32.100	TCP
133	845.314634764	192.168.32.101	192.168.32.100	TCP
134	845.314666881	192.168.32.100	192.168.32.101	TCP
135	845.315169572	192.168.32.101	192.168.32.100	TCP
136	845.316615534	192.168.32.101	192.168.32.100	TLSv1
137	845.316633622	192.168.32.100	192.168.32.101	TCP
138	845.346269860	192.168.32.100	192.168.32.101	TLSv1
139	845.357184843	192.168.32.101	192.168.32.100	TLSv1
140	845.357631224	192.168.32.100	192.168.32.101	TLSv1
141	845.369308478	192.168.32.101	192.168.32.100	TLSv1
142	845.381160395	192.168.32.100	192.168.32.101	TLSv1
143	845.384128287	192.168.32.100	192.168.32.101	TLSv1
144	845.385322522	192.168.32.101	192.168.32.100	TCP
145	845.386114317	192.168.32.101	192.168.32.100	TCP
146	845.386134216	192.168.32.100	192.168.32.101	TCP
147	845.501071490	192.168.32.101	192.168.32.100	TCP
148	845.501132866	192.168.32.100	192.168.32.101	TCP
149	845.502165127	192.168.32.101	192.168.32.100	TCP
150	845.512376215	192.168.32.100	192.168.32.100	TLSv1

Packet Details (Frame 138):

- Linux cooked capture v1
- Packet type: Sent by us (4)
- Link-layer address type: Ethernet (1)
- Link-layer address length: 6
- Source: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5)
- Unused: 0000
- Protocol: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
- Transmission Control Protocol, Src Port: 443, Dst Port: 49230, Seq: 1
- Transport Layer Security
 - TLSv1 Record Layer: Handshake Protocol: Server Hello
 - TLSv1 Record Layer: Handshake Protocol: Certificate
 - TLSv1 Record Layer: Handshake Protocol: Server Key Exchange
 - TLSv1 Record Layer: Handshake Protocol: Server Hello Done

Packet Bytes:

```
00e0 04 0a 0c 07 49 4e 65 74 53 69 6d 31 14 30 12 06 ... INet Sim1 0...
00f0 03 55 04 0b 0c 0b 44 65 76 65 6c 6f 70 6d 65 6e ... U... De velopen
0100 74 31 14 30 12 06 03 55 04 03 0c 0b 69 6e 65 74 ... t1 0... U... inet
0110 73 69 6d 2e 6f 72 07 30 1e 17 0d 32 33 30 38 32 ... sim.org0 ... 23082
0120 31 31 30 35 38 33 37 5a 17 0d 33 30 30 38 31 30 ... 1105037Z ... 330818
0130 31 38 35 38 33 37 5a 30 3e 31 10 30 0e 06 03 55 ... 1858377Z >1 0... U
0140 04 0a 0c 07 49 4e 65 74 53 69 6d 31 14 30 12 06 ... INet Sim1 0...
0150 03 55 04 0b 0c 0b 44 65 76 65 6c 6f 70 6d 65 6e ... U... De velopen
```