

Traccia esercizio:

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.

Parte 1: setup dei requisiti e servizi

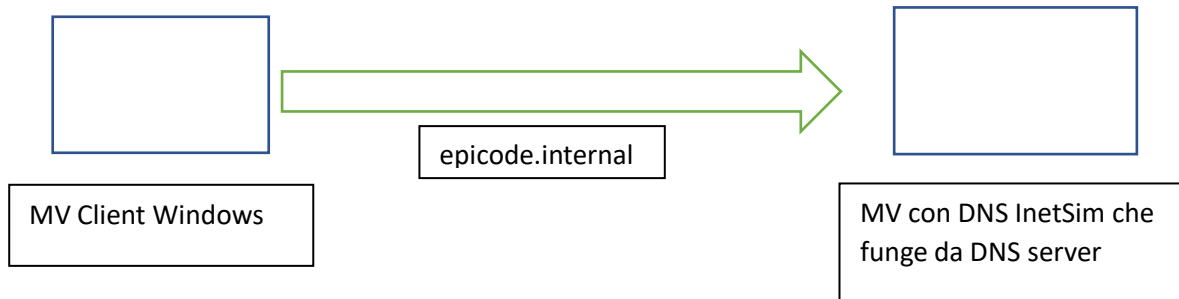
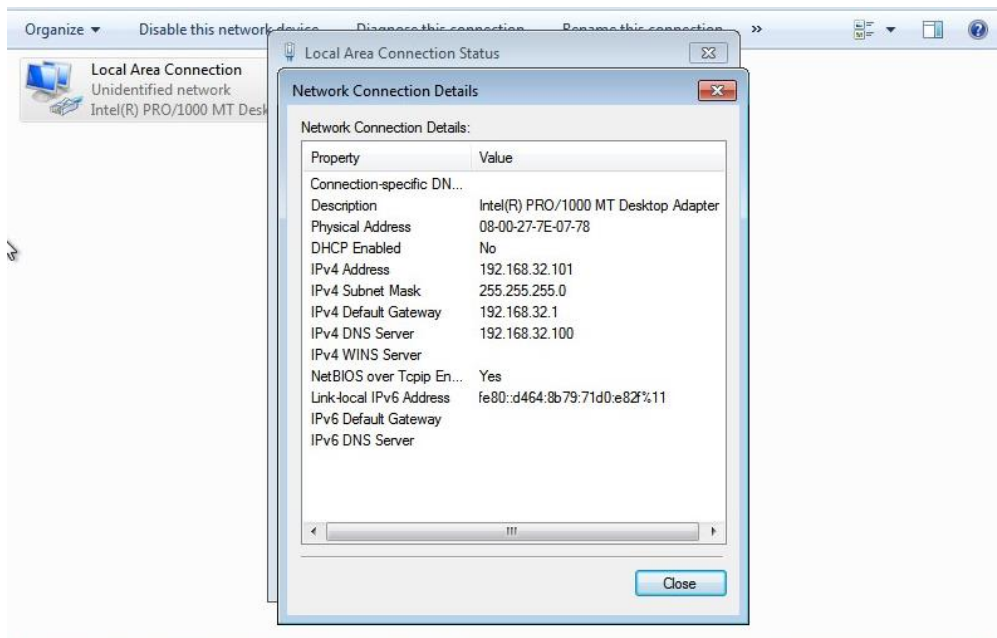
Kali Linux:

```
(davide@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:fe7c:ab04 prefixlen 64 scopeid 0x20<link>  
    ether 08:00:27:7c:ab:04 txqueuelen 1000 (Ethernet)  
    RX packets 547 bytes 46257 (45.1 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 290 bytes 37589 (36.7 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

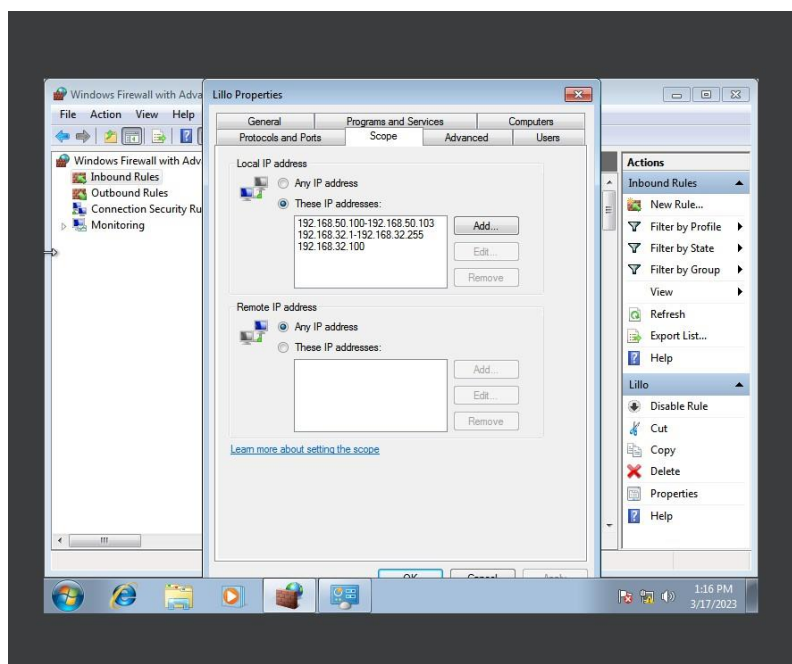
InetSim:

```
Configuration file parsed successfully.  
== INetSim main process started (PID 5327) ==  
Session ID: 5327  
Listening on: 0.0.0.0  
Real Date/Time: 2023-03-17 13:11:15  
Fake Date/Time: 2023-03-17 13:11:15 (Delta: 0 seconds)  
Forking services ...  
  * dns_53_tcp_udp - started (PID 5333)  
  * http_80_tcp - started (PID 5334)  
  * https_443_tcp - started (PID 5335)  
done.  
Simulation running.  
^C * https_443_tcp - stopped (PID 5335)  
  * http_80_tcp - stopped (PID 5334)  
  * dns_53_tcp_udp - stopped (PID 5333)  
Simulation stopped.  
Report written to '/var/log/inetsim/report/report.5327.txt' (28 lines)  
== INetSim main process stopped (PID 5327) ==
```

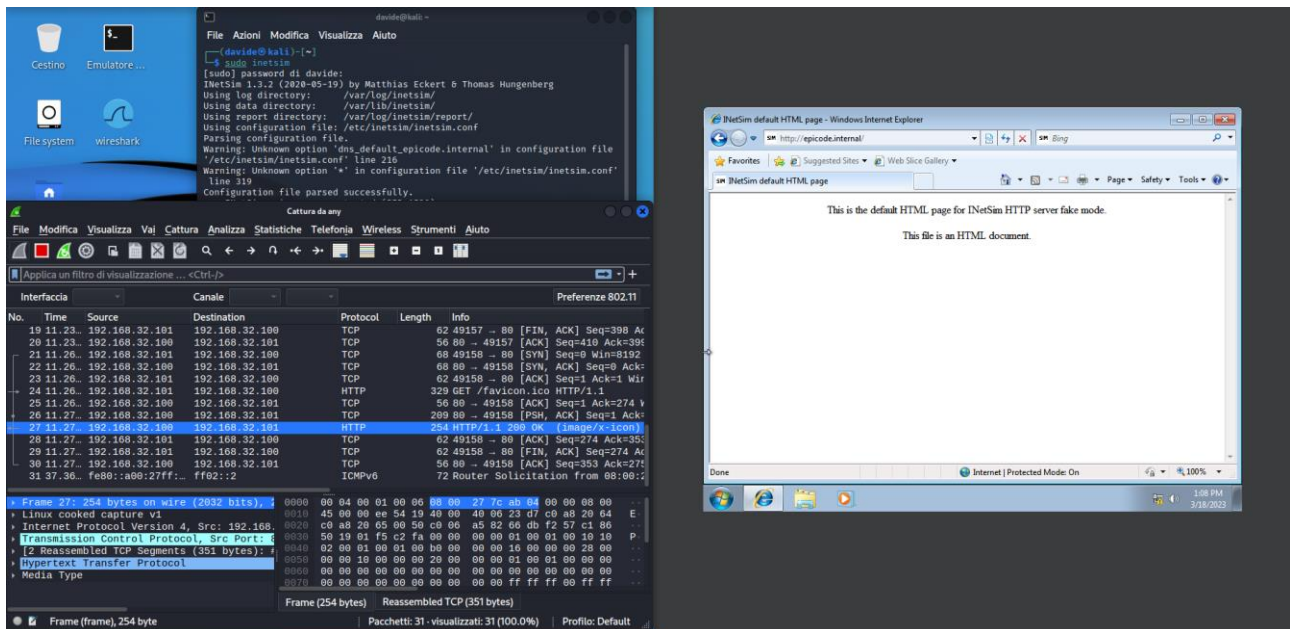
Windows:



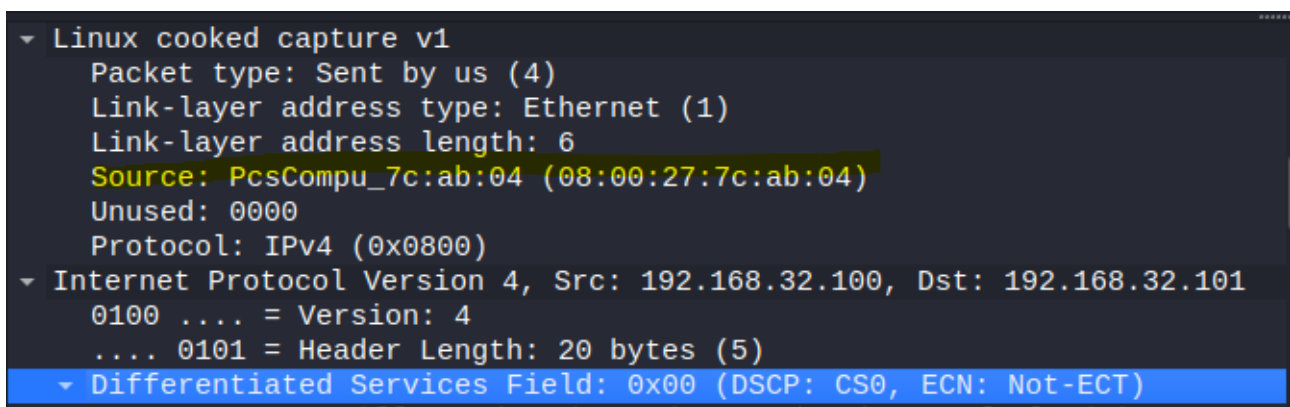
Setup firewall:



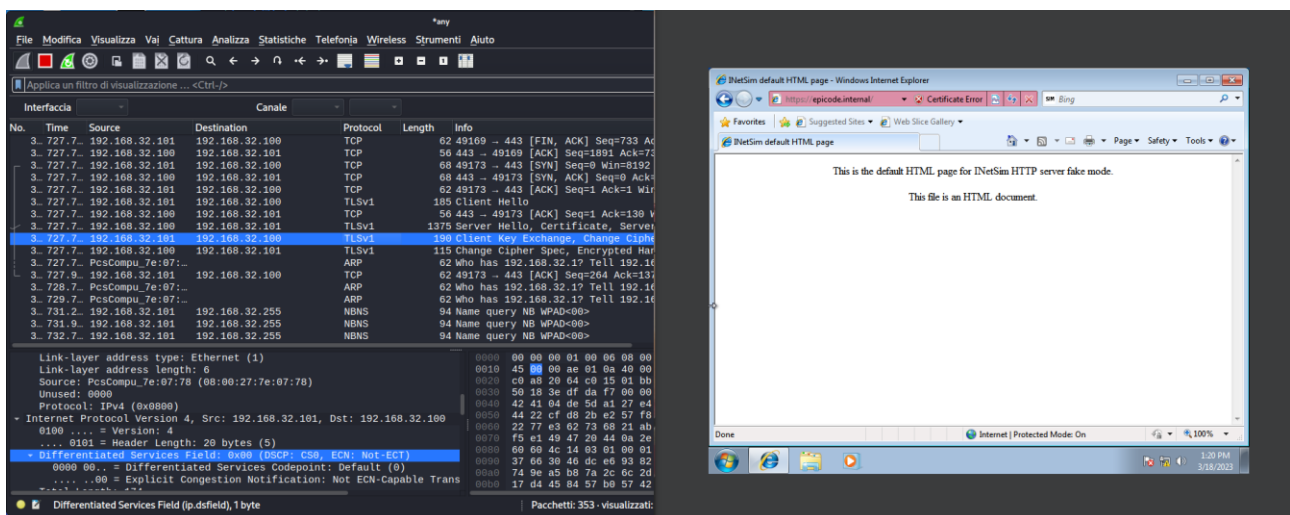
Verifica del traffico http con Wireshark in HTTP



Indirizzo MAC source



Verifica del traffico http con Wireshark in HTTPS



Dettaglio indirizzo MAC in HTTPS (uguale)

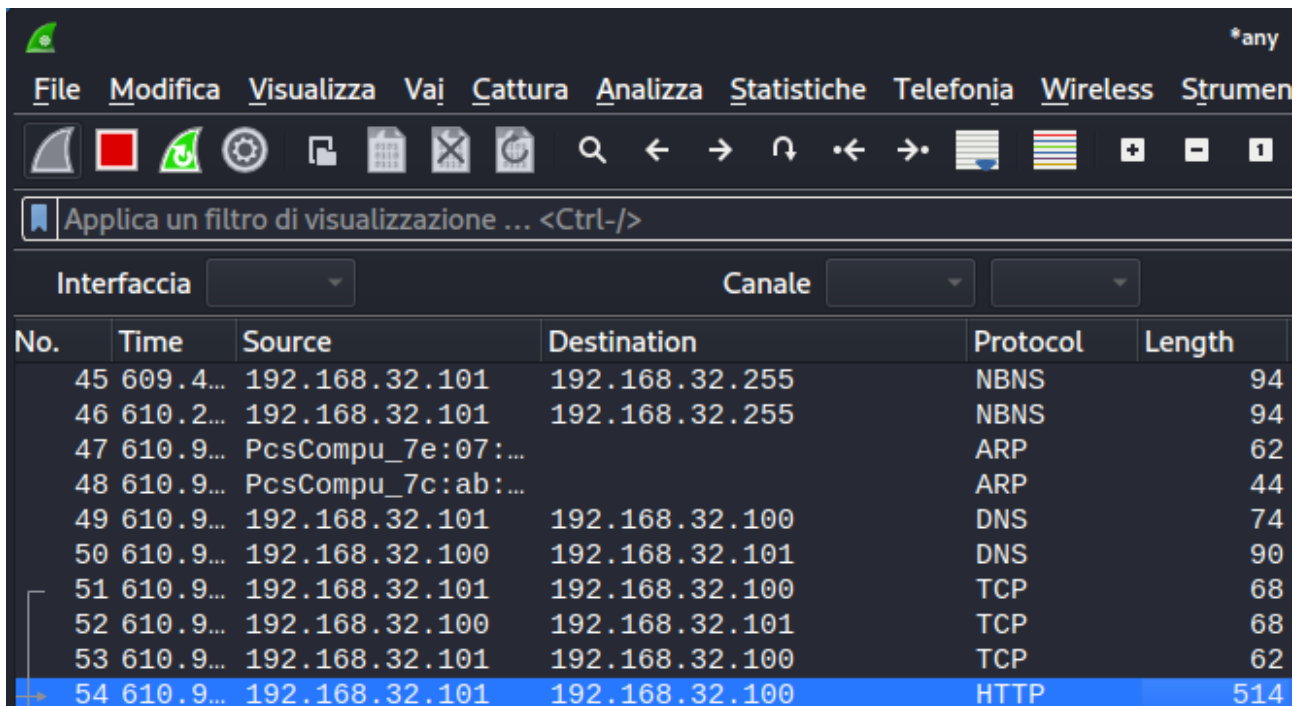
```

Link-layer address type: Ethernet (1)
Link-layer address length: 6
Source: PcsCompu_7e:07:78 (08:00:27:7e:07:78)
Unused: 0000
Protocol: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differenziated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
0000 00.. = Differentiated Services Codepoint: Default (0)
.... ..00 = Explicit Congestion Notification: Not ECN-Capable Trans

```

Somiglianze e differenze

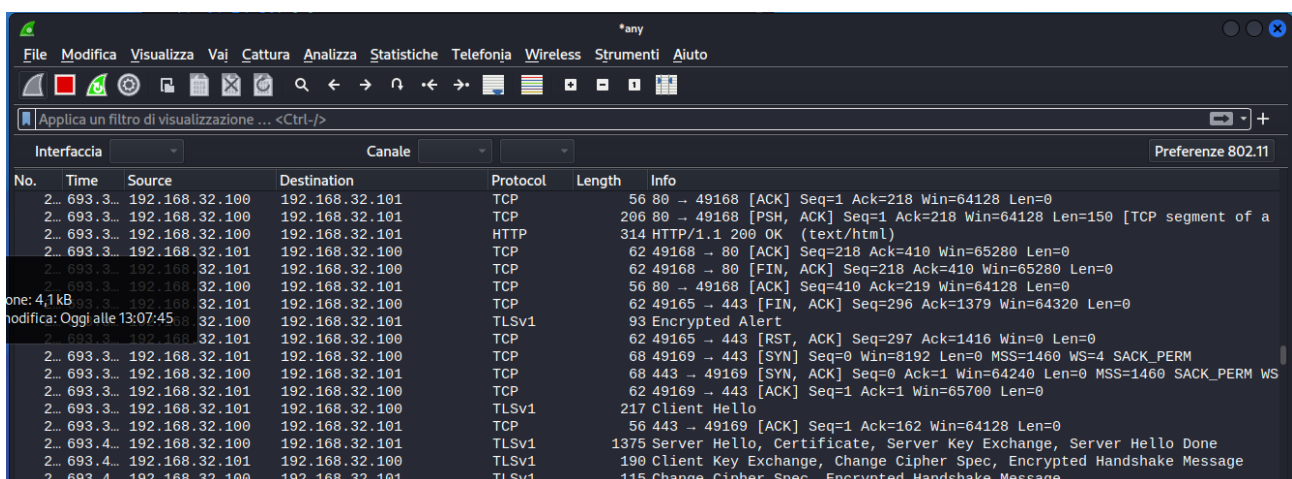
http:



Wireshark interface showing a list of network packets. The 'Filter' bar is empty. The 'Interfaccia' is set to 'any' and 'Canale' is set to 'any'. The packet list shows a sequence of packets, with packet 54 highlighted in blue, indicating it is selected. Packet 54 is an HTTP request from 192.168.32.101 to 192.168.32.100.

No.	Time	Source	Destination	Protocol	Length
45	609.4...	192.168.32.101	192.168.32.255	NBNS	94
46	610.2...	192.168.32.101	192.168.32.255	NBNS	94
47	610.9...	PcsCompu_7e:07:...		ARP	62
48	610.9...	PcsCompu_7c:ab:...		ARP	44
49	610.9...	192.168.32.101	192.168.32.100	DNS	74
50	610.9...	192.168.32.100	192.168.32.101	DNS	90
51	610.9...	192.168.32.101	192.168.32.100	TCP	68
52	610.9...	192.168.32.100	192.168.32.101	TCP	68
53	610.9...	192.168.32.101	192.168.32.100	TCP	62
54	610.9...	192.168.32.101	192.168.32.100	HTTP	514

https:



Wireshark interface showing a list of network packets. The 'Filter' bar is empty. The 'Interfaccia' is set to 'any' and 'Canale' is set to 'any'. The packet list shows a sequence of packets, with packet 2 highlighted in blue, indicating it is selected. Packet 2 is a TCP segment from 192.168.32.100 to 192.168.32.101.

No.	Time	Source	Destination	Protocol	Length	Info
2	693.3...	192.168.32.100	192.168.32.101	TCP	56	80 → 49168 [ACK] Seq=1 Ack=218 Win=64128 Len=0
2	693.3...	192.168.32.100	192.168.32.101	TCP	206	80 → 49168 [PSH, ACK] Seq=1 Ack=218 Win=64128 Len=150 [TCP segment of a
2	693.3...	192.168.32.100	192.168.32.101	HTTP	314	HTTP/1.1 200 OK (text/html)
2	693.3...	192.168.32.101	192.168.32.100	TCP	62	49168 → 80 [ACK] Seq=218 Ack=410 Win=65280 Len=0
2	693.3...	192.168.32.101	192.168.32.100	TCP	62	49168 → 80 [FIN, ACK] Seq=218 Ack=410 Win=65280 Len=0
2	693.3...	192.168.32.100	192.168.32.101	TCP	56	80 → 49168 [ACK] Seq=410 Ack=219 Win=64128 Len=0
2	693.3...	192.168.32.101	192.168.32.100	TCP	62	49165 → 443 [FIN, ACK] Seq=296 Ack=1379 Win=64320 Len=0
2	693.3...	192.168.32.101	192.168.32.100	TLSv1	93	Encrypted Alert
2	693.3...	192.168.32.101	192.168.32.100	TCP	62	49165 → 443 [RST, ACK] Seq=297 Ack=1416 Win=0 Len=0
2	693.3...	192.168.32.101	192.168.32.100	TCP	68	49169 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
2	693.3...	192.168.32.100	192.168.32.101	TCP	68	443 → 49169 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS
2	693.3...	192.168.32.101	192.168.32.100	TCP	62	49169 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
2	693.3...	192.168.32.101	192.168.32.100	TLSv1	217	Client Hello
2	693.3...	192.168.32.100	192.168.32.101	TCP	56	443 → 49169 [ACK] Seq=1 Ack=162 Win=64128 Len=0
2	693.4...	192.168.32.100	192.168.32.101	TLSv1	1375	Server Hello, Certificate, Server Key Exchange, Server Hello Done
2	693.4...	192.168.32.101	192.168.32.100	TLSv1	190	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
2	693.4...	192.168.32.100	192.168.32.101	TLSv1	115	Change Cipher Spec, Encrypted Handshake Message

Come è possibile notare dalle due schermate, entrambe le richieste viaggiano su protocollo TCP, stabilendo fra client e server un three-way handshake (SYN, SYN ACK, ACK).

La differenza fra i due protocolli sta di fatto nella “S” finale. Essendo l’HTTPS un protocollo secured, è necessaria la creazione di un canale di scambio criptato per l’invio della chiave di decodifica dell’algoritmo di cifratura dei dati (nell’immagine, il protocollo TLSv1 – Transport Layer Security)