

Configurazione IP statico e client/server DSN

Kali Linux 192.168.32.100

```
PS> kali@kali: /home/kali

File Actions Edit View Help

PowerShell 7.2.6
Copyright (c) Microsoft Corporation.

https://aka.ms/powershell
Type 'help' to get help.

(kali@kali)-[/home/kali]
PS> 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:fec7:e136 prefixlen 64 scopeid 0<link>
    ether 08:00:27:c7:e1:36 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 16 bytes 2414 (2.3 KiB)
    TX errors 0 dropped 0 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 0<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Windows 7 192.168.32.101

```
ca. C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\vbouser>ipconfig /all

Windows IP Configuration

Host Name . . . . . : Windows7
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . :
Description . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
Physical Address. . . . . : 08-00-27-2F-48-B8
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::8df0:bc24:6d35:101f%11(Preferred)
IPv4 Address. . . . . : 192.168.32.101(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.32.1
DHCPv6 IAID . . . . . : 235405351
DHCPv6 Client DUID. . . . . : 00-01-00-01-2C-17-D7-B4-08-00-27-2F-48-B8

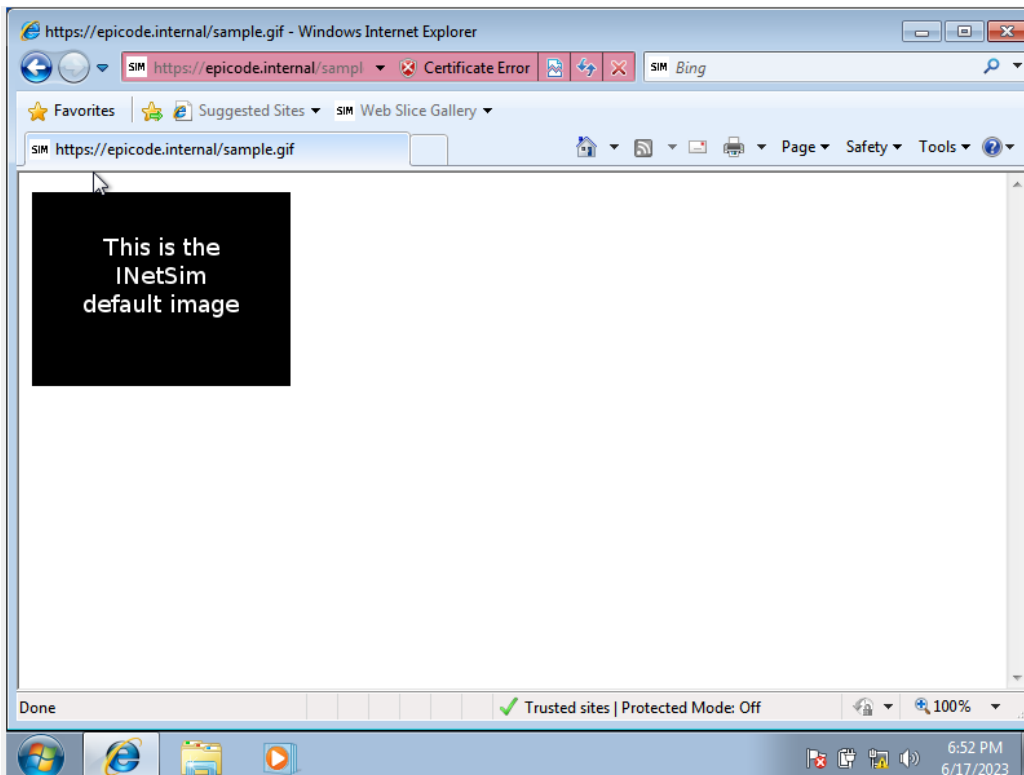
DNS Servers . . . . . : 192.168.32.100
NetBIOS over Tcpip. . . . . : Enabled

Tunnel adapter isatap.{B857AEDD-AFFD-47F1-A6AF-A76DC6B5C81D}:

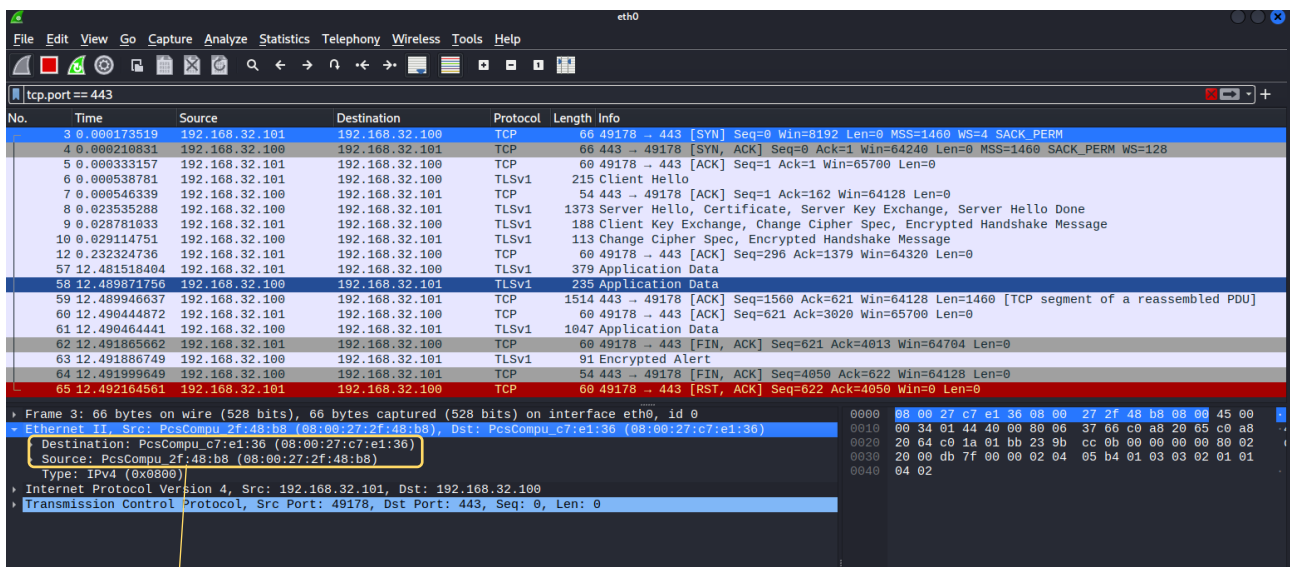
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
Description . . . . . : Microsoft ISATAP Adapter
Physical Address. . . . . : 00-00-00-00-00-00-E0
DHCP Enabled. . . . . : No
Autoconfiguration Enabled . . . . : Yes

C:\Users\vbouser>
```

Richiesta HTTPS da Windows 7 a epicode.internal/sample.gif



Pacchetti catturati con Wireshark (richiesta HTTPS)

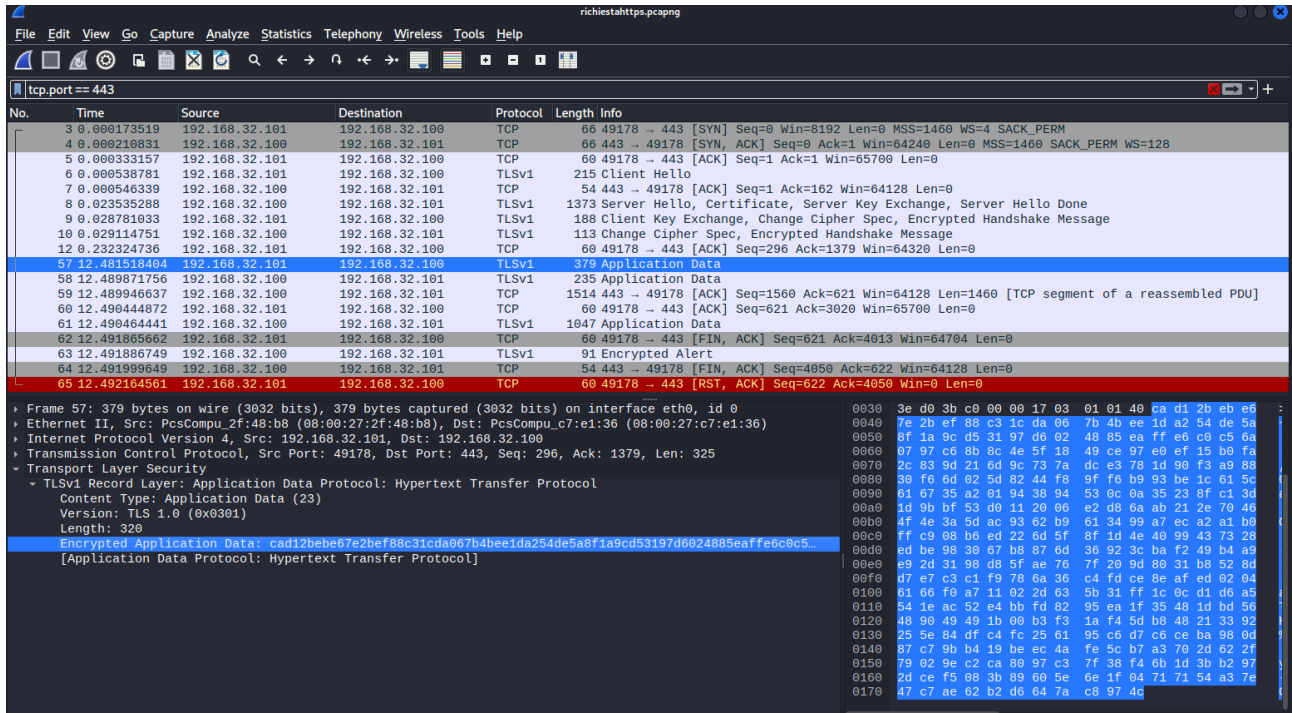


indirizzo MAC sorgente: 08:00:27:2f:48:b8

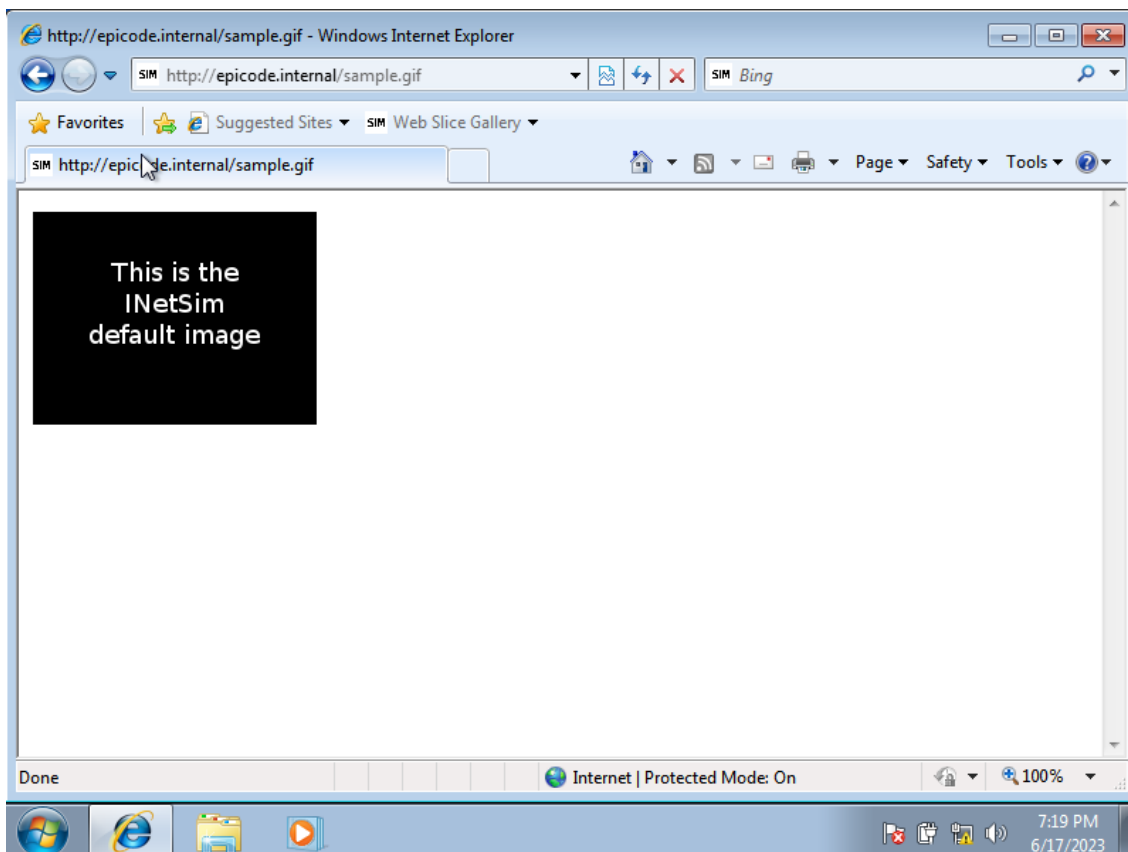
indirizzo MAC destinazione: 08:00:27:c7:e1:36

Contenuto della richiesta HTTPS

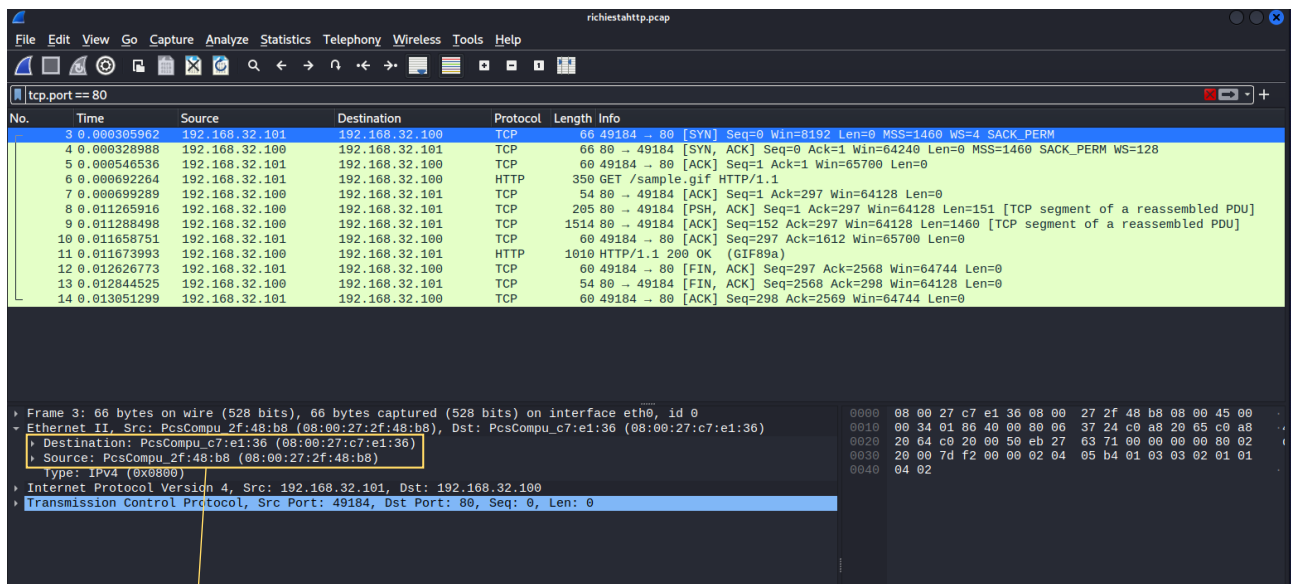
Trattandosi di richiesta HTTPS (o HTTP secure), il contenuto è criptato



Richiesta HTTP da Windows 7 a epicode.internal/sample.gif



Pacchetti catturati con Wireshark (richiesta HTTP)

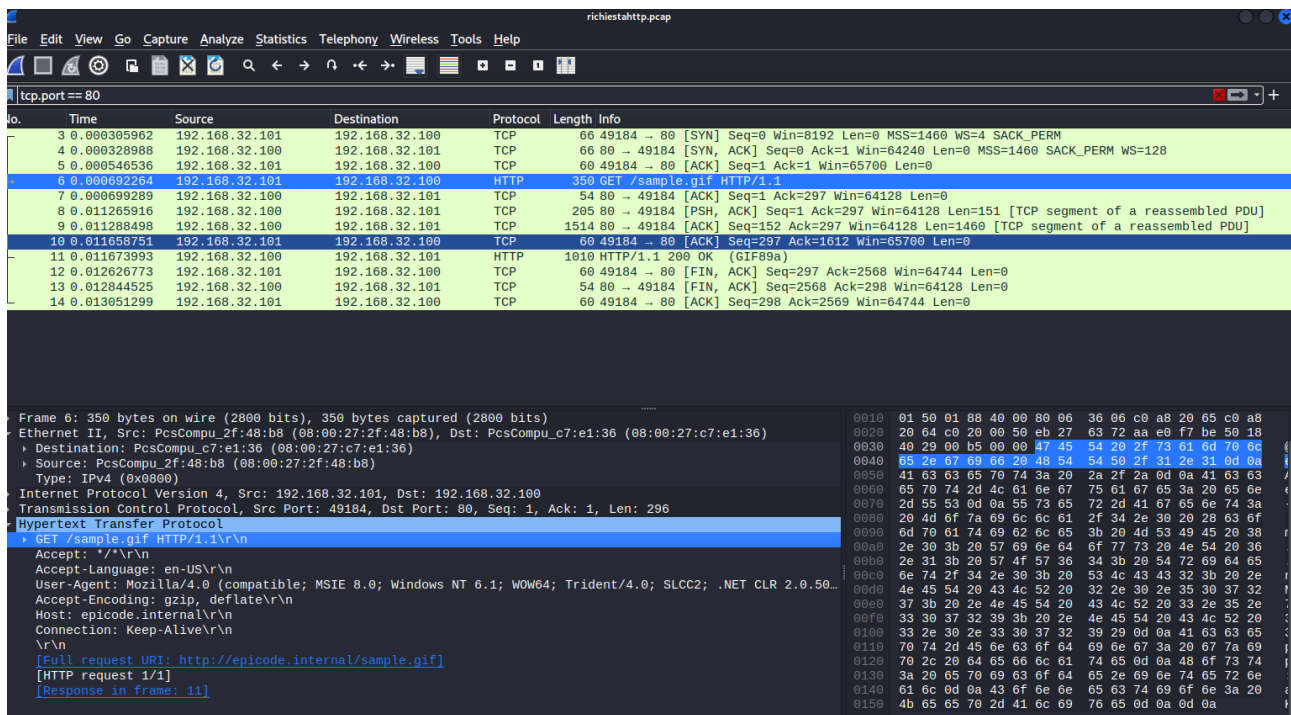


indirizzo MAC sorgente: 08:00:27:2f:48:b8

indirizzo MAC destinazione: 08:00:27:c7:e1:36

Contenuto della richiesta HTTP

In questo caso il contenuto della richiesta è in chiaro:



Differenze tra HTTPS e HTTP

Le richieste HTTPS e HTTP utilizzano entrambe il protocollo TCP per il trasporto su porte differenti.

Il protocollo HTTPS utilizza la porta 443

```

> Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: PcsCompu_2f:48:b8 (08:00:27:2f:48:b8), Dst: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49178, Dst Port: 443, Seq: 0, Len: 0

```

mentre il protocollo HTTP utilizza la porta 80

```

> Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: PcsCompu_2f:48:b8 (08:00:27:2f:48:b8), Dst: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49184, Dst Port: 80, Seq: 0, Len: 0

```

La differenza principale tra i due protocolli però sta nel layer applicazione.

Nel caso di HTTP, il protocollo utilizzato è HTTP e la richiesta è in chiaro.

Nel layer applicazione vediamo che viene utilizzato l'Hypertext Transfer Protocol. In questo caso la richiesta utilizza il metodo GET e il file "sample.gif" che è stato richiesto è visibile in chiaro nel pacchetto.

```

> Transmission Control Protocol, Src Port: 49184, Dst Port: 80, Seq: 1, Ack: 1, Len: 296
  Hypertext Transfer Protocol
    GET /sample.gif HTTP/1.1\r\n
    [Expert Info (Char/Sequence): GET /sample.gif HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /sample.gif
      Request Version: HTTP/1.1
    Accept: */*\r\n
    Accept-Language: en-US\r\n
    User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; ...)\r\n
    Accept-Encoding: gzip, deflate\r\n
    Host: epicode.internal\r\n
    Connection: Keep-Alive\r\n
    \r\n
    [Full request URI: http://epicode.internal/sample.gif]
    [HTTP request 1/1]
    [Response in frame: 11]

```

Vediamo nel pacchetto successivo di risposta dal server che l'esito della richiesta è OK (codice stato 200).

```
Ethernet II, Src: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36), Dst: PcsCompu_2f:48:b8 (08:00:27:c7:48:b8)
  Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
  Transmission Control Protocol, Src Port: 80, Dst Port: 49184, Seq: 1612, Ack: 297, Len: 1
  [3 Reassembled TCP Segments (2567 bytes): #8(151), #9(1460), #11(956)]
  Hypertext Transfer Protocol
    HTTP/1.1 200 OK\r\n
    Content-Length: 2416\r\n
    Date: Sat, 17 Jun 2023 17:21:28 GMT\r\n
    Server: INetSim HTTP Server\r\n
    Connection: Close\r\n
    Content-Type: image/gif\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 0.010982000 seconds]
    [Request in frame: 6]
    [Request URI: http://epicode.internal/sample.gif]
    File Data: 2416 bytes
  Compuserve GIF, Version: GIF89a
```

Nel caso di HTTPS, il contenuto è criptato utilizzando un protocollo cifrato (TLSv1 in questo caso). Al protocollo “Hypertext Transfer Protocol” che abbiamo nel caso di HTTP, nella richiesta HTTPS viene sovrapposto il protocollo “Transport Layer Security”. I pacchetti vengono scambiati in formato cifrato dopo una comunicazione client/server che comprende lo scambio della chiave di sicurezza:

No.	Time	Source	Destination	Protocol	Length	Info
3	0.000173	192.168.32.101	192.168.32.100	TCP	66	49178 → 443 [SYN]
4	0.000210	192.168.32.100	192.168.32.101	TCP	66	443 → 49178 [SYN]
5	0.000333	192.168.32.101	192.168.32.100	TCP	60	49178 → 443 [ACK]
6	0.000538	192.168.32.101	192.168.32.100	TLSv1	215	Client Hello
7	0.000546	192.168.32.100	192.168.32.101	TCP	54	443 → 49178 [ACK]
8	0.023535	192.168.32.100	192.168.32.101	TLSv1	1373	Server Hello, C
9	0.028781	192.168.32.101	192.168.32.100	TLSv1	188	Client Key Exch
10	0.029114	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher S
12	0.232324	192.168.32.101	192.168.32.100	TCP	60	49178 → 443 [ACK]
57	12.481518	192.168.32.101	192.168.32.100	TLSv1	379	Application Dat
58	12.489871	192.168.32.100	192.168.32.101	TLSv1	235	Application Dat
59	12.489946	192.168.32.100	192.168.32.101	TCP	1514	443 → 49178 [ACK]
60	12.490444	192.168.32.101	192.168.32.100	TCP	60	49178 → 443 [ACK]
61	12.490464	192.168.32.100	192.168.32.101	TLSv1	1047	Application Dat
62	12.491865	192.168.32.101	192.168.32.100	TCP	60	49178 → 443 [FIN]
63	12.491886	192.168.32.100	192.168.32.101	TLSv1	91	Encrypted Alert
64	12.491999	192.168.32.100	192.168.32.101	TCP	54	443 → 49178 [FIN]
65	12.492164	192.168.32.101	192.168.32.100	TCP	60	49178 → 443 [RST]

```
.....
  Frame 6: 215 bytes on wire (1720 bits), 215 bytes captured (1720 bits)
  Ethernet II, Src: PcsCompu_2f:48:b8 (08:00:27:2f:48:b8), Dst: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36)
  Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
  Transmission Control Protocol, Src Port: 49178, Dst Port: 443, Seq: 1, Ack: 1, Len: 161
  Transport Layer Security
    TLSv1 Record Layer: Handshake Protocol: Client Hello
```



```

> Frame 8: 1373 bytes on wire (10984 bits), 1373 bytes captured (10984 bits)
> Ethernet II, Src: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36), Dst: PcsCompu_2f:48:b8 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
> Transmission Control Protocol, Src Port: 443, Dst Port: 49178, Seq: 1, Ack: 162, Len: 1319
> 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

```

```

> Frame 9: 188 bytes on wire (1504 bits), 188 bytes captured (1504 bits)
> Ethernet II, Src: PcsCompu_2f:48:b8 (08:00:27:2f:48:b8), Dst: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49178, Dst Port: 443, Seq: 162, Ack: 1320, Len: 1319
> Transport Layer Security
  > TLSv1 Record Layer: Handshake Protocol: Client Key Exchange
  > TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
  > TLSv1 Record Layer: Handshake Protocol: Encrypted Handshake Message

```

```

> Frame 10: 113 bytes on wire (904 bits), 113 bytes captured (904 bits)
> Ethernet II, Src: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36), Dst: PcsCompu_2f:48:b8 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
> Transmission Control Protocol, Src Port: 443, Dst Port: 49178, Seq: 1320, Ack: 296, Len: 5
> Transport Layer Security
  > TLSv1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
  > TLSv1 Record Layer: Handshake Protocol: Encrypted Handshake Message

```

fino ad arrivare alla richiesta vera e propria, che viene trasmessa in formato criptato e non è quindi direttamente leggibile nel pacchetto.

Time	Source	Destination	Protocol	Details
3 0.000173	192.168.32.101	192.168.32.100	TCP	66 49178 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460
4 0.000210	192.168.32.100	192.168.32.101	TCP	66 443 → 49178 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
5 0.000333	192.168.32.101	192.168.32.100	TCP	60 49178 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
6 0.000538	192.168.32.101	192.168.32.100	TLSv1	215 Client Hello
7 0.000546	192.168.32.100	192.168.32.101	TCP	54 443 → 49178 [ACK] Seq=1 Ack=162 Win=64128 Len=0
8 0.023535	192.168.32.100	192.168.32.101	TLSv1	1373 Server Hello, Certificate, Server Key Exchange, Server Hello Done
9 0.028781	192.168.32.101	192.168.32.100	TLSv1	188 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
10 0.029114	192.168.32.100	192.168.32.101	TLSv1	113 Change Cipher Spec, Encrypted Handshake Message
12 0.232324	192.168.32.101	192.168.32.100	TCP	60 49178 → 443 [ACK] Seq=296 Ack=1379 Win=64320 Len=0
57 12.481518	192.168.32.101	192.168.32.100	TLSv1	379 Application Data
58 12.489871	192.168.32.100	192.168.32.101	TLSv1	235 Application Data
59 12.489946	192.168.32.100	192.168.32.101	TCP	1514 443 → 49178 [ACK] Seq=1560 Ack=621 Win=64128 Len=0
60 12.490444	192.168.32.101	192.168.32.100	TCP	60 49178 → 443 [ACK] Seq=621 Ack=3020 Win=65700 Len=0
61 12.490464	192.168.32.100	192.168.32.101	TLSv1	1047 Application Data
62 12.491865	192.168.32.101	192.168.32.100	TCP	60 49178 → 443 [FIN, ACK] Seq=621 Ack=4013 Win=0 Len=0
63 12.491886	192.168.32.100	192.168.32.101	TLSv1	91 Encrypted Alert
64 12.491999	192.168.32.100	192.168.32.101	TCP	54 443 → 49178 [FIN, ACK] Seq=4050 Ack=622 Win=0 Len=0
65 12.492164	192.168.32.101	192.168.32.100	TCP	60 49178 → 443 [RST, ACK] Seq=622 Ack=4050 Win=0 Len=0

```

> Frame 57: 379 bytes on wire (3032 bits), 379 bytes captured (3032 bits)
> Ethernet II, Src: PcsCompu_2f:48:b8 (08:00:27:2f:48:b8), Dst: PcsCompu_c7:e1:36 (08:00:27:c7:e1:36)
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49178, Dst Port: 443, Seq: 296, Ack: 1379, Len: 325
> Transport Layer Security
  > TLSv1 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
    Content Type: Application Data (23)
    Version: TLS 1.0 (0x0301)
    Length: 320
    Encrypted Application Data: cad12bebe67e2bef88c31cda067b4bee1da254de5a8f1a9cd53197d6024885eaffec0c5...
    [Application Data Protocol: Hypertext Transfer Protocol]

```