

ESERCITAZIONE W4D4 :

Testo esercizio:

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.

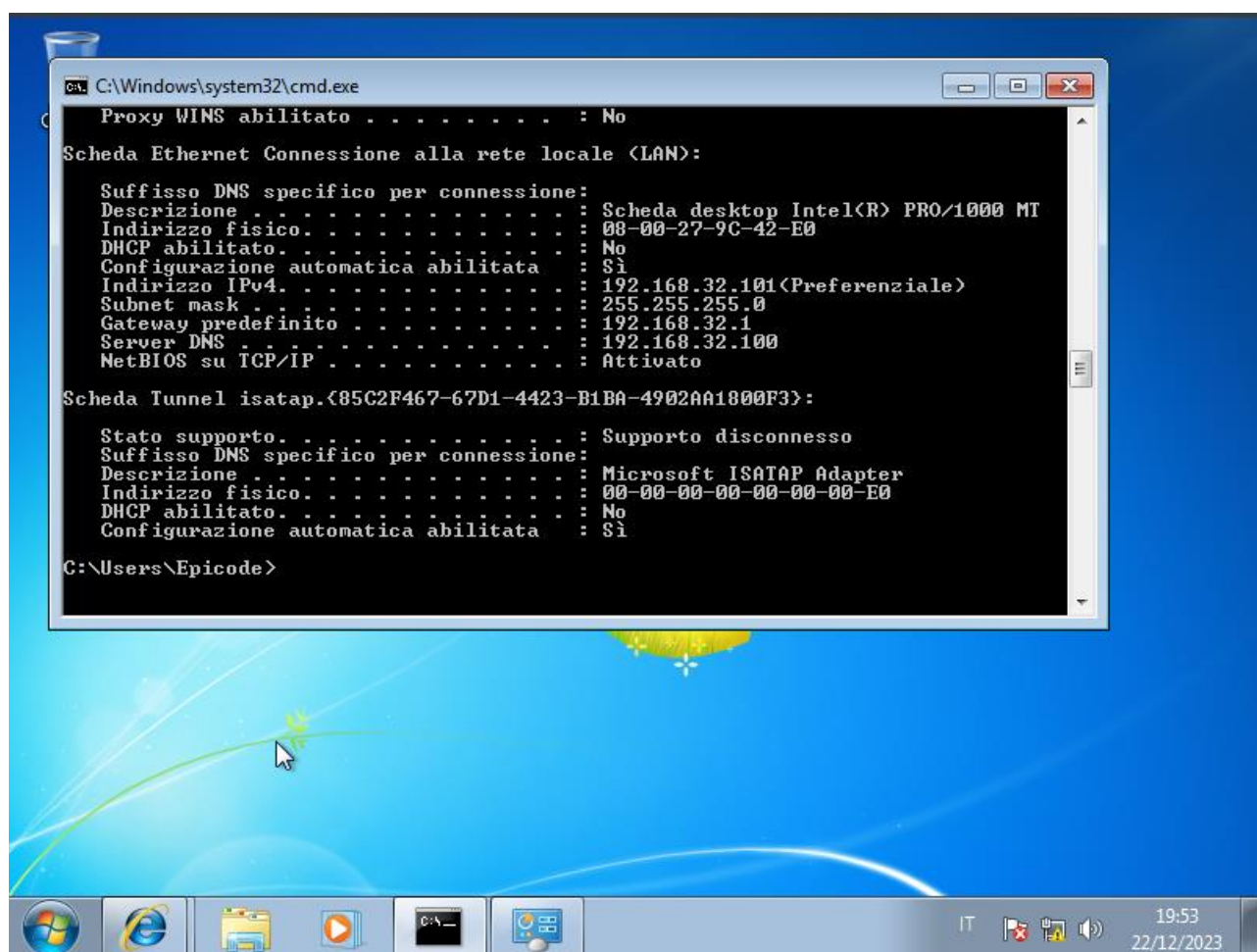
PREPARAZIONE AMBIENTE DI TEST:

Settaggio Windows

In questa fase ho settato l'ip statico di windows che come da traccia deve essere : 192.168.32.101

Importante settare gateway e anche dns per l'esercizio che dobbiamo svolgere

In questa schermata possiamo trovare anche il mac address che ci tornerà utile più avanti

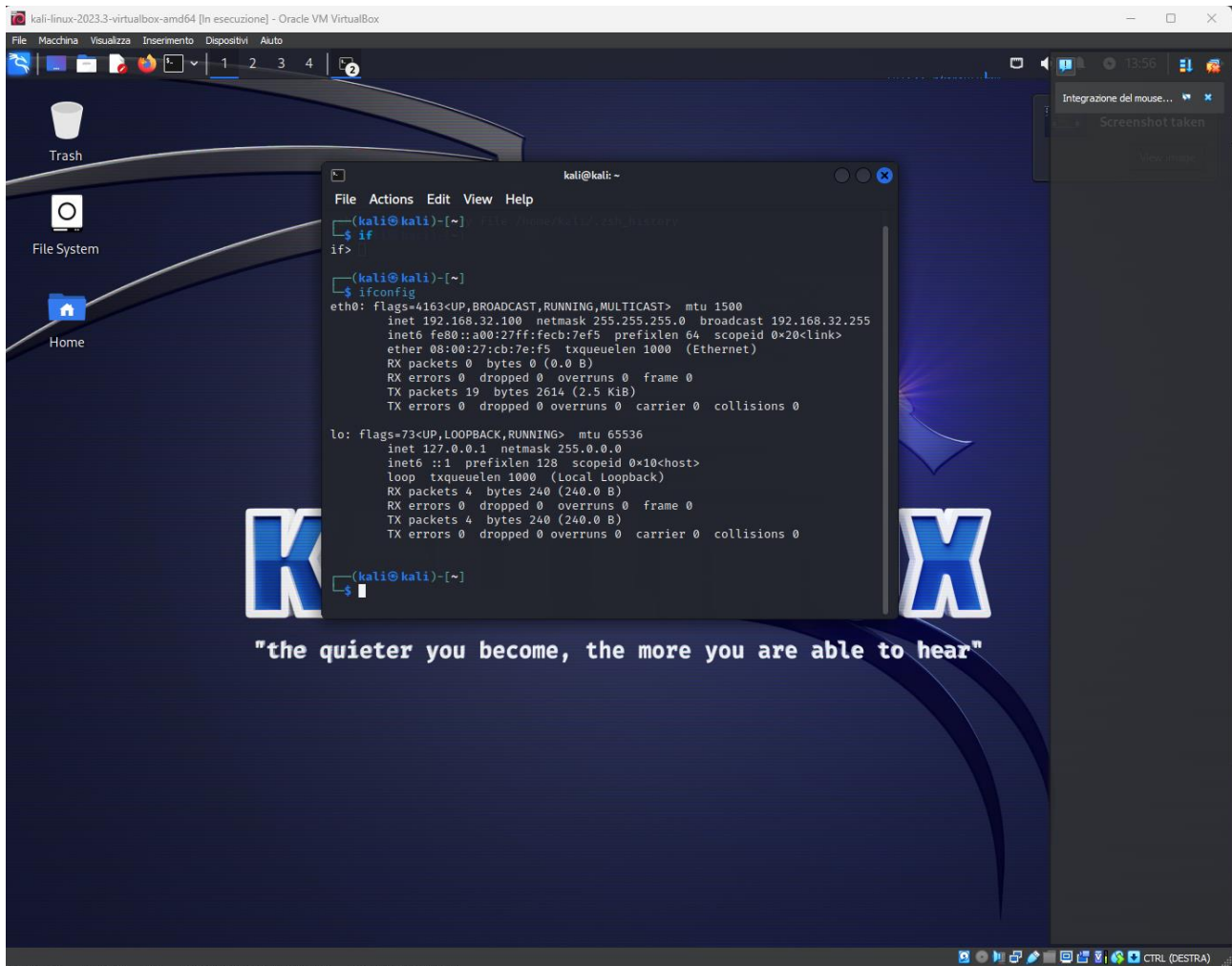


Settaggio linux:

In questa fase ho settato l'ip statico di kali che come da traccia deve essere : 192.168.32.100

Importante settare gateway

In questa schermata possiamo trovare anche il mac address che ci tornerà utile più avanti

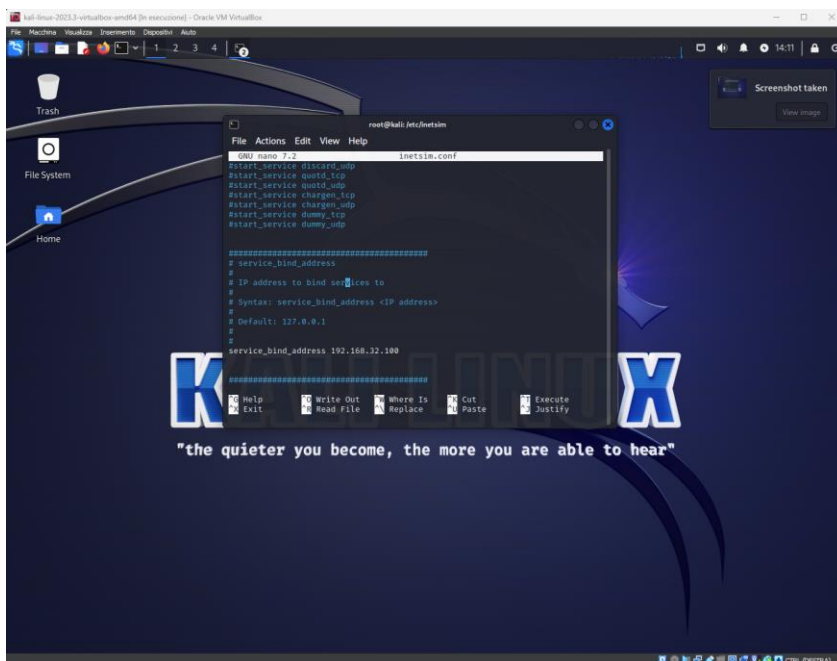
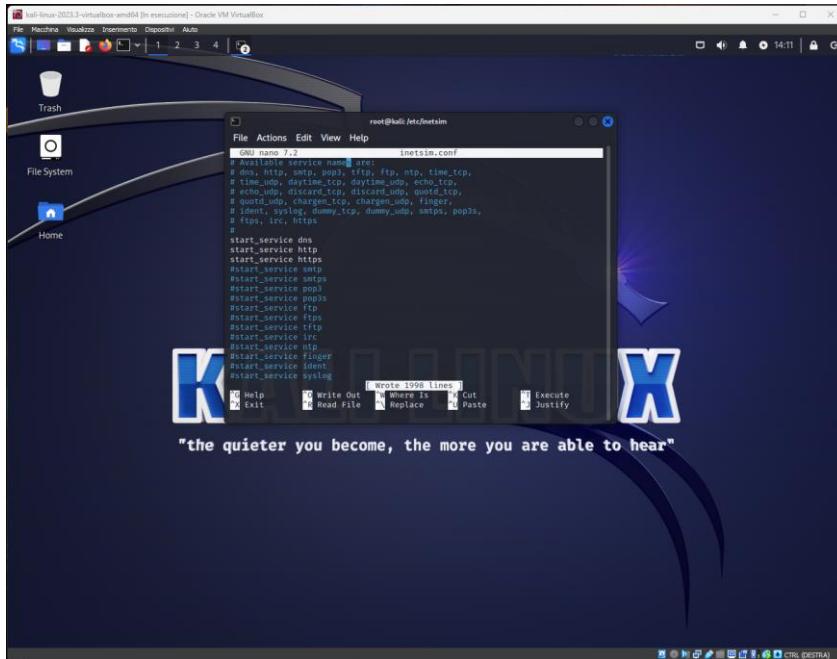


```
kali@kali: ~  
$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.32.100 netmask 255.255.255.0 broadcast 192.168.32.255  
    inet6 fe80::a00:27ff:feeb:7ef5 prefixlen 64 scopeid 0<link>  
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 19 bytes 2614 (2.5 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
```

ATTIVAZIONE SERVIZI http, HTTPS E DNS:

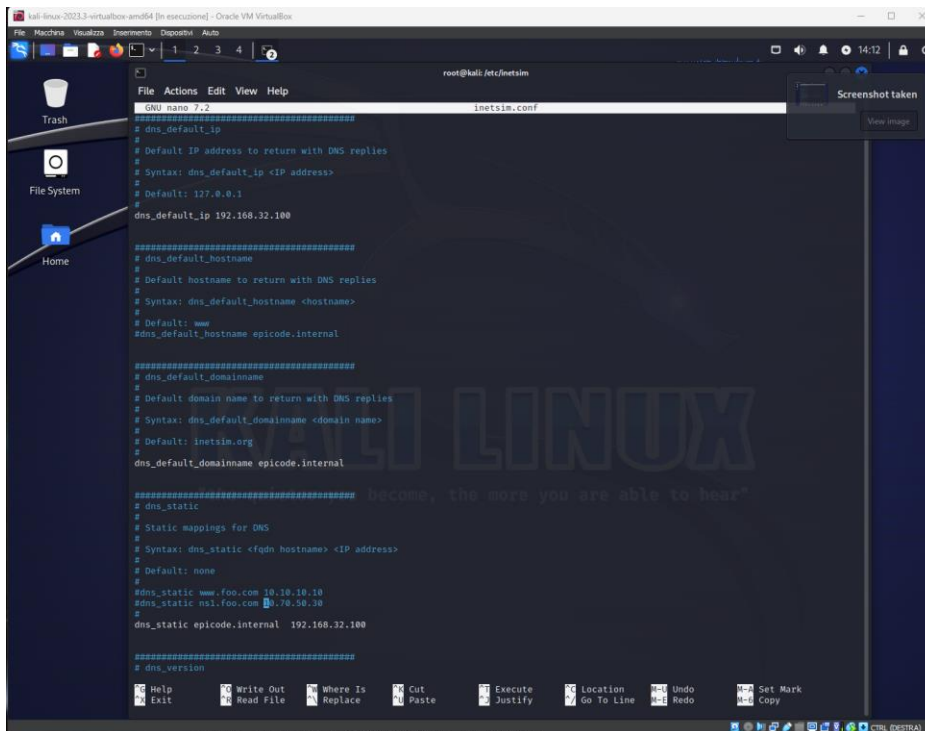
In queste slide possiamo vedere quali siano i servizi da tenere attivi su inetsim.

Quando andremo ad utilizzare l'http bisogna andare a disattivare il servizio di https e viceversa.



SETTINGS DNS:

In questa fase andiamo a dire ad inetsim qual è l'indirizzo ip del dns (in questo caso lo stesso linux) dopo di che si va a specificare qual è il dns default domain che come da testo è epicode.internal ed infine andiamo ad associare ip e domain name così che si possa raggiungere il sito tramite il dominio epicode.internal



The screenshot shows a Kali Linux terminal window with the nano text editor open to the file inetsim.conf. The user is root@kali: /etc/inetsim. The configuration file contains the following settings:

```
#####
# dns_default_ip
#
# Default IP address to return with DNS replies
# Syntax: dns_default_ip <IP address>
#
# Default: 127.0.0.1
dns_default_ip 192.168.32.100

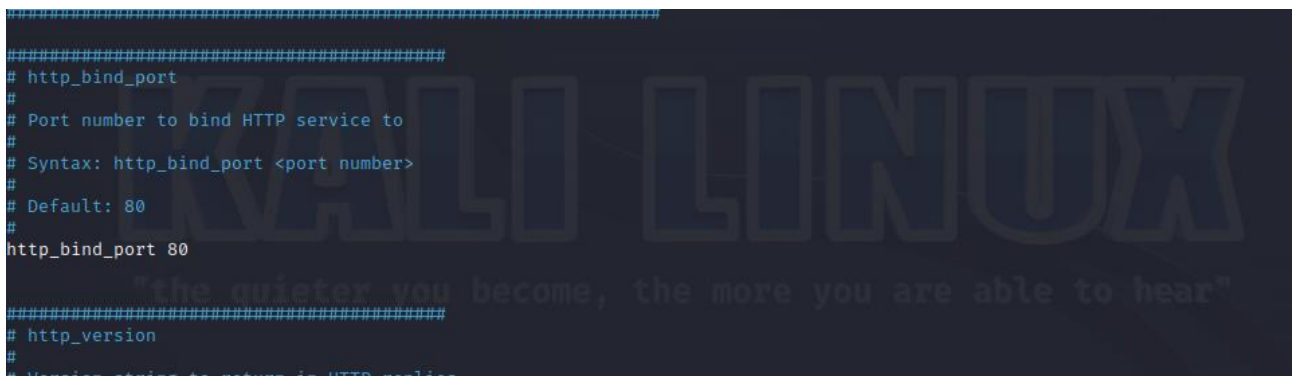
#####
# dns_default_hostname
#
# Default hostname to return with DNS replies
# Syntax: dns_default_hostname <hostname>
#
# Default: www
#dns_default_hostname epicode.internal

#####
# dns_default_domainname
#
# Default domain name to return with DNS replies
# Syntax: dns_default_domainname <domain name>
#
# Default: inetsim.org
#
dns_default_domainname epicode.internal

#####
# 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 10.10.10.10
#
dns_static epicode.internal 192.168.32.100

#####
# dns_version
```

SETUP HTTP:



The screenshot shows a Kali Linux terminal window with the nano text editor open to the file http.conf. The user is root@kali: /etc/httpd. The configuration file contains the following settings:

```
#####
# http_bind_port
#
# Port number to bind HTTP service to
# Syntax: http_bind_port <port number>
#
# Default: 80
#
http_bind_port 80

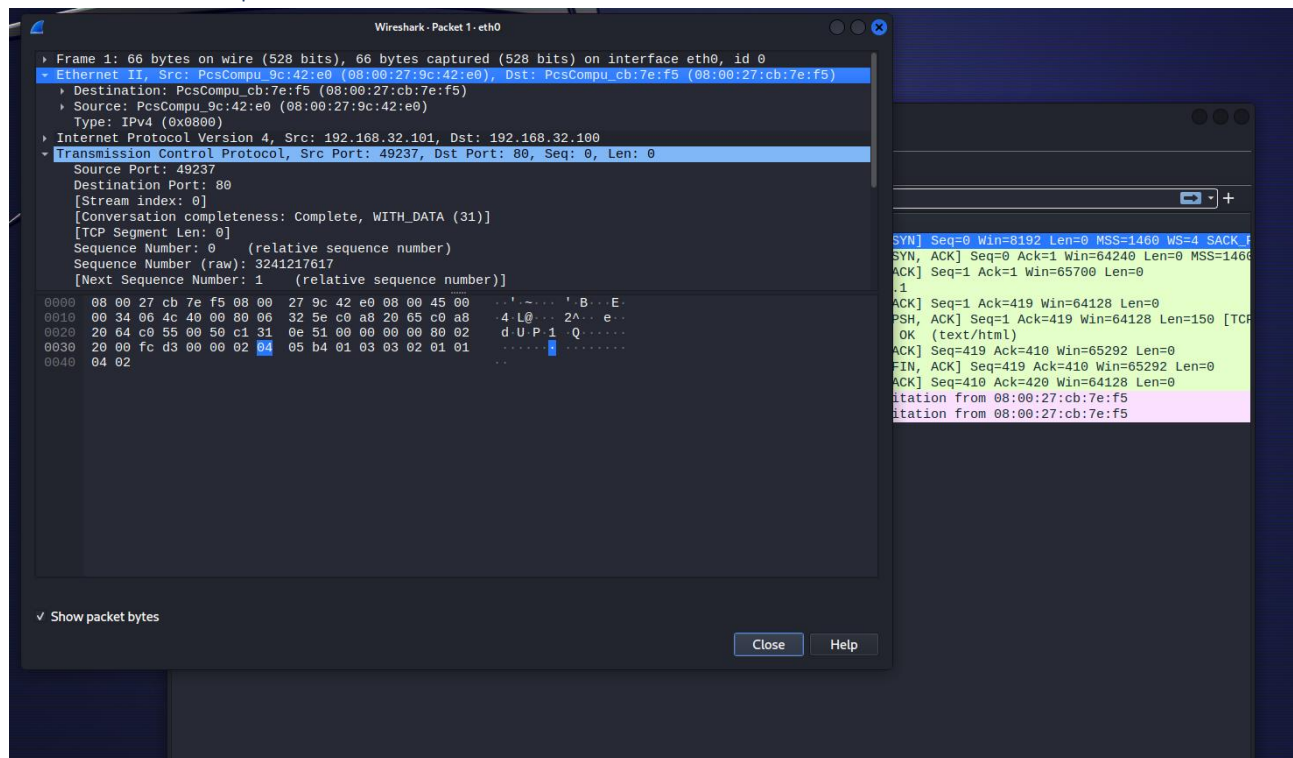
#####
# http_version
#
# Version string to return in HTTP replies
```

SETUP HTTPS:

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

Effettuate queste modifiche possiamo lanciare inetsim così da mettere in ascolto il server

MAC ADDRESS http:



The image shows a Wireshark packet capture window titled "Wireshark - Packet 1: eth0". The packet list on the left shows a single packet: "Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0, id 0". The packet details pane on the right shows the following information:

- Ethernet II, Src: PcsCompu_9c:42:e0 (08:00:27:9c:42:e0), Dst: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5)
- Destination: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5)
- Source: PcsCompu_9c:42:e0 (08:00:27:9c:42:e0)
- Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
- Transmission Control Protocol, Src Port: 49237, Dst Port: 80, Seq: 0, Len: 0

The packet bytes pane at the bottom shows the raw data of the packet, which is a SYN packet. The packet is 66 bytes long, and the raw data is displayed in hexadecimal and ASCII. The packet is a SYN packet, and the raw data is displayed in hexadecimal and ASCII.

On the right side of the image, there is a list of captured packets, showing a sequence of SYN and ACK packets. The list includes the following information for each packet:

- SYN, Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_F
- SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
- ACK] Seq=1 Ack=1 Win=65700 Len=0
- ACK] Seq=1 Ack=419 Win=64128 Len=0
- PSH, ACK] Seq=1 Ack=419 Win=64128 Len=150 [TCP
- OK (text/html)
- ACK] Seq=419 Ack=410 Win=65292 Len=0
- FIN, ACK] Seq=419 Ack=410 Win=65292 Len=0
- ACK] Seq=410 Ack=420 Win=64128 Len=0
- itation from 08:00:27:cb:7e:f5
- itation from 08:00:27:cb:7e:f5

HTTPS CONNESSIONE :

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Display a display filter: «eth0»

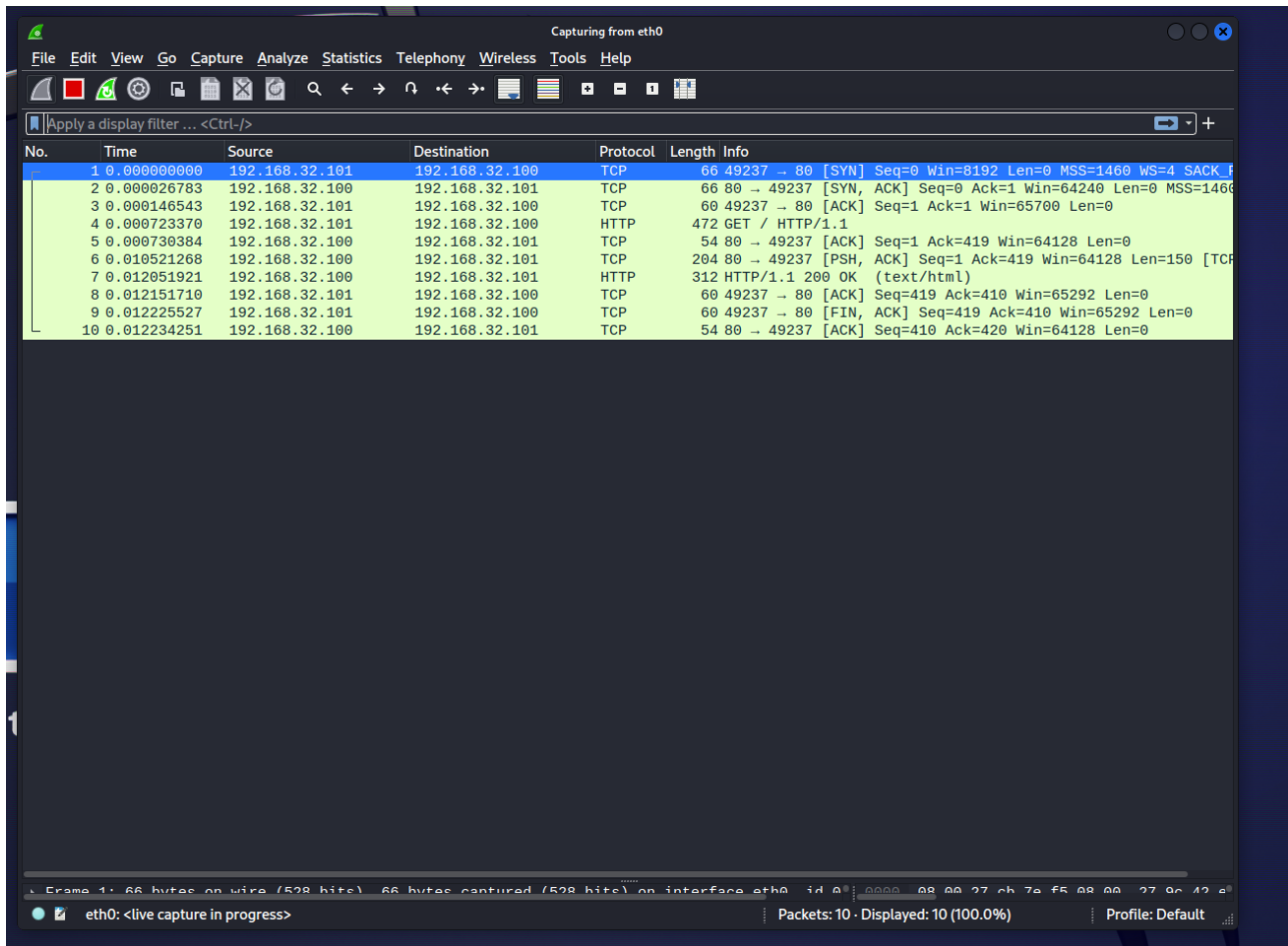
Capturing from eth0

1 2 3 4

14:37

No.	Time	Source	Destination	Protocol	Length	Info
2.0	0.00002460	192.168.32.101	192.168.32.101	TCP	60	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
3.0	0.00014817	192.168.32.101	192.168.32.101	TCP	60	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
4.0	0.00157989	192.168.32.101	192.168.32.101	TLSv1	176	Client Hello
5.0	0.00164826	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0
6.0	0.00009306	192.168.32.101	192.168.32.101	TLSv1	1368	Server Hello, Certificate, Server Key Exchange, Server Hello Done
7.0	0.00009950	192.168.32.101	192.168.32.101	TLSv1	1368	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
8.0	0.00010739	192.168.32.101	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
9.0	0.00061575	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
10.0	0.00060957	192.168.32.101	192.168.32.101	TCP	60	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
11.0	0.00121726	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
12.0	1.00101321	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
13.0	1.17970368	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x0724 A wad
14.0	1.28074480	192.168.32.101	224.0.0.252	LLMNR	92	Name query MB WPA-D0 wad
15.0	1.48994907	192.168.32.101	192.168.32.255	MNMG	92	Name query MB WPA-D0 wad
16.0	1.23899153	192.168.32.101	192.168.32.255	MNMG	92	Name query MB WPA-D0 wad
17.0	1.00034839	192.168.32.101	192.168.32.101	TCP	60	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
18.0	5.74119543	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
19.0	6.00074203	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
20.0	7.60773211	PcsCompu_9c42:e0	Broadcast	AMP	60	Who has 192.168.32.17 Tell 192.168.32.101
21.0	8.78939238	192.168.32.101	224.0.0.252	LLMNR	64	Standard query 0x005F A wad
22.0	8.98032123	192.168.32.101	224.0.0.252	LLMNR	92	Standard query 0x005F A wad
23.0	9.18959489	192.168.32.101	192.168.32.255	MNMG	92	Name query MB WPA-D0 wad
24.0	9.00050409	192.168.32.101	192.168.32.255	MNMG	92	Name query MB WPA-D0 wad
25.0	10.08935732	192.168.32.101	192.168.32.255	MNMG	92	Name query MB WPA-D0 wad
26.0	11.00034839	192.168.32.101	192.168.32.101	TCP	50	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
27.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
28.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
29.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
30.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
31.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
32.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
33.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 60 [RST] Seq=433 [Win=0] Seq=158182 Len=0 MSS=1460 WS=4 SACK_PERM
34.0	11.00034839	192.168.32.101	192.168.32.101	TCP	54	60 → 6

http CONNESSIONE:



Wireshark packet capture showing an HTTP connection. The interface is 'eth0' and the capture is in progress. The packet list shows 10 packets. The first packet is a SYN from 192.168.32.101 to 192.168.32.100. The second packet is a SYN, ACK from 192.168.32.100 to 192.168.32.101. The third packet is an ACK from 192.168.32.101 to 192.168.32.100. The fourth packet is a GET request from 192.168.32.101 to 192.168.32.100. The fifth packet is an ACK from 192.168.32.100 to 192.168.32.101. The sixth packet is a PSH, ACK from 192.168.32.100 to 192.168.32.101. The seventh packet is an HTTP/1.1 200 OK response from 192.168.32.100 to 192.168.32.101. The eighth packet is an ACK from 192.168.32.101 to 192.168.32.100. The ninth packet is a FIN, ACK from 192.168.32.101 to 192.168.32.100. The tenth packet is an ACK from 192.168.32.100 to 192.168.32.101.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.101	192.168.32.100	TCP	66	49237 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK=0
2	0.000026783	192.168.32.100	192.168.32.101	TCP	66	80 → 49237 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
3	0.000146543	192.168.32.101	192.168.32.100	TCP	60	49237 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
4	0.000723370	192.168.32.101	192.168.32.100	HTTP	472	GET / HTTP/1.1
5	0.000730384	192.168.32.100	192.168.32.101	TCP	54	80 → 49237 [ACK] Seq=1 Ack=419 Win=64128 Len=0
6	0.010521268	192.168.32.100	192.168.32.101	TCP	204	80 → 49237 [PSH, ACK] Seq=1 Ack=419 Win=64128 Len=150 [TCP
7	0.012051921	192.168.32.100	192.168.32.101	HTTP	312	HTTP/1.1 200 OK (text/html)
8	0.012151710	192.168.32.101	192.168.32.100	TCP	60	49237 → 80 [ACK] Seq=419 Ack=410 Win=65292 Len=0
9	0.012225527	192.168.32.101	192.168.32.100	TCP	60	49237 → 80 [FIN, ACK] Seq=419 Ack=410 Win=65292 Len=0
10	0.012234251	192.168.32.100	192.168.32.101	TCP	54	80 → 49237 [ACK] Seq=410 Ack=420 Win=64128 Len=0

MACRO DIFFERENZE:

Come possiamo vedere le differenze principali tra un pacchetto https e http sono la quantità di scambi che vi sono, dato che nel https il server e il client devono mettersi d'accordo su quali chiavi utilizzare per criptare e decriptare i pacchetti che si scambiano.

Si può notare che nel http possiamo vedere tutto il contenuto dei pacchetti che server e client si sono scambiati nell'arco di tutta la comunicazione, al contrario l'https tramite il tls crea un tunnel criptato nel quale non è possibile entrare in possesso di informazioni che vengono scambiate tra client e server.