

Beatrice Folino

# [Progetto di valutazione]

*EPICODE - CYBERSECURITY CLASS [W4D4 Prova Pratica]*

*17 novembre 2023*





La prova illustrata chiede di verificare le competenze finora acquisite iniziando da alcune premesse:

- **settaggio IP Kali Linus VM su 192.168.32.100**
- **settaggio IP Windows7 VM su 192.132.32.101**
- **HTTPS server attivo**
- **Servizio DNS per risoluzione nomi dominio attivo**

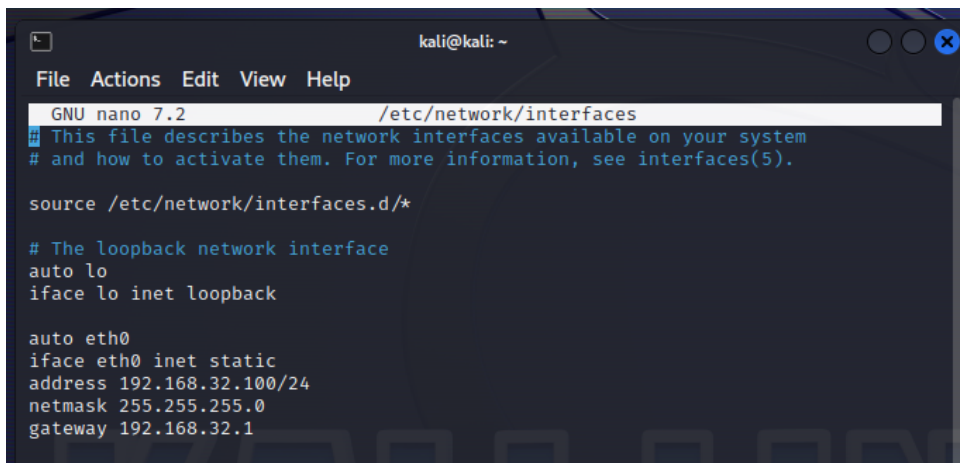
Occorre simulare, in un laboratorio virtuale un'architettura client server, in cui un client con indirizzo 192.168.32.101 (Windows 7) richieda tramite web browser una risorsa all'hostname epicode.internal che risponde all'indirizzo 192.168.32.100 (Kali).

Infine bisogna intercettare 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.

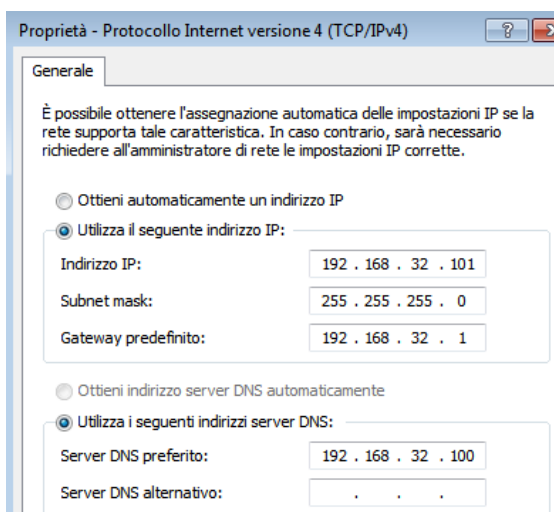
Intercettare nuovamente il traffico, evidenziando le eventuali differenze tra il traffico appena catturato in HTTP ed il traffico precedente in HTTPS. Spiegare, motivandole, le principali differenze se presenti.

## CONFIGURAZIONE IP KALI E WINDOWS7

Per permettere il corretto svolgimento della traccia, il primo passo è stato configurare gli indirizzi IP delle macchine virtuali come da premesse, e cioè:



```
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 eth0  
iface eth0 inet static  
address 192.168.32.100/24  
netmask 255.255.255.0  
gateway 192.168.32.1
```



Proprietà - Protocollo Internet versione 4 (TCP/IPv4)

Generale

È possibile ottenere l'assegnazione automatica delle impostazioni IP se la rete supporta tale caratteristica. In caso contrario, sarà necessario richiedere all'amministratore di rete le impostazioni IP corrette.

☐ Ottieni automaticamente un indirizzo IP

☒ Utilizza il seguente indirizzo IP:

Indirizzo IP: 192 , 168 , 32 , 101

Subnet mask: 255 , 255 , 255 , 0

Gateway predefinito: 192 , 168 , 32 , 1

☐ Ottieni indirizzo server DNS automaticamente

☒ Utilizza i seguenti indirizzi server DNS:

Server DNS preferito: 192 , 168 , 32 , 100

Server DNS alternativo: . , . , .

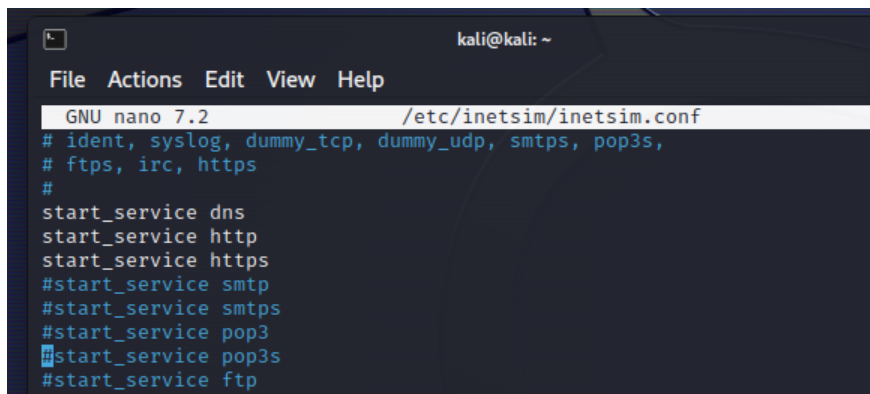
## CONFIGURAZIONE INETSIM.CONF

Successivamente, per soddisfare le premesse, è stato necessario recarsi all'interno di **inetsim.conf** in qualità di super-utente e in modalità scrittura, ovvero avviando il terminale di Kali e digitando **sudo nano /etc/inetsim/inetsim.conf**.

Nel caso non risultasse subito disponibile, la directory è raggiungibile digitando **cd /etc/inetsim** e successivamente **ls**, in questo modo ci verrà esplicitata la posizione del file e potremo procedere a digitare il comando di cui sopra per fare le modifiche di nostro interesse.

Una volta aperto il file, sarà necessario modificarlo avendo accortezza di inserire i valori come da immagini.

- 1) “Scommentiamo” i servizi che ci interessa mantenere attivi, in questo caso dns, http, https.



```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/inetsim/inetsim.conf  
# 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
```

- 2) Impostiamo il `service_bind_address` con l'indirizzo IP della macchina che dovrà fare da fake server, ovvero quello di Kali.

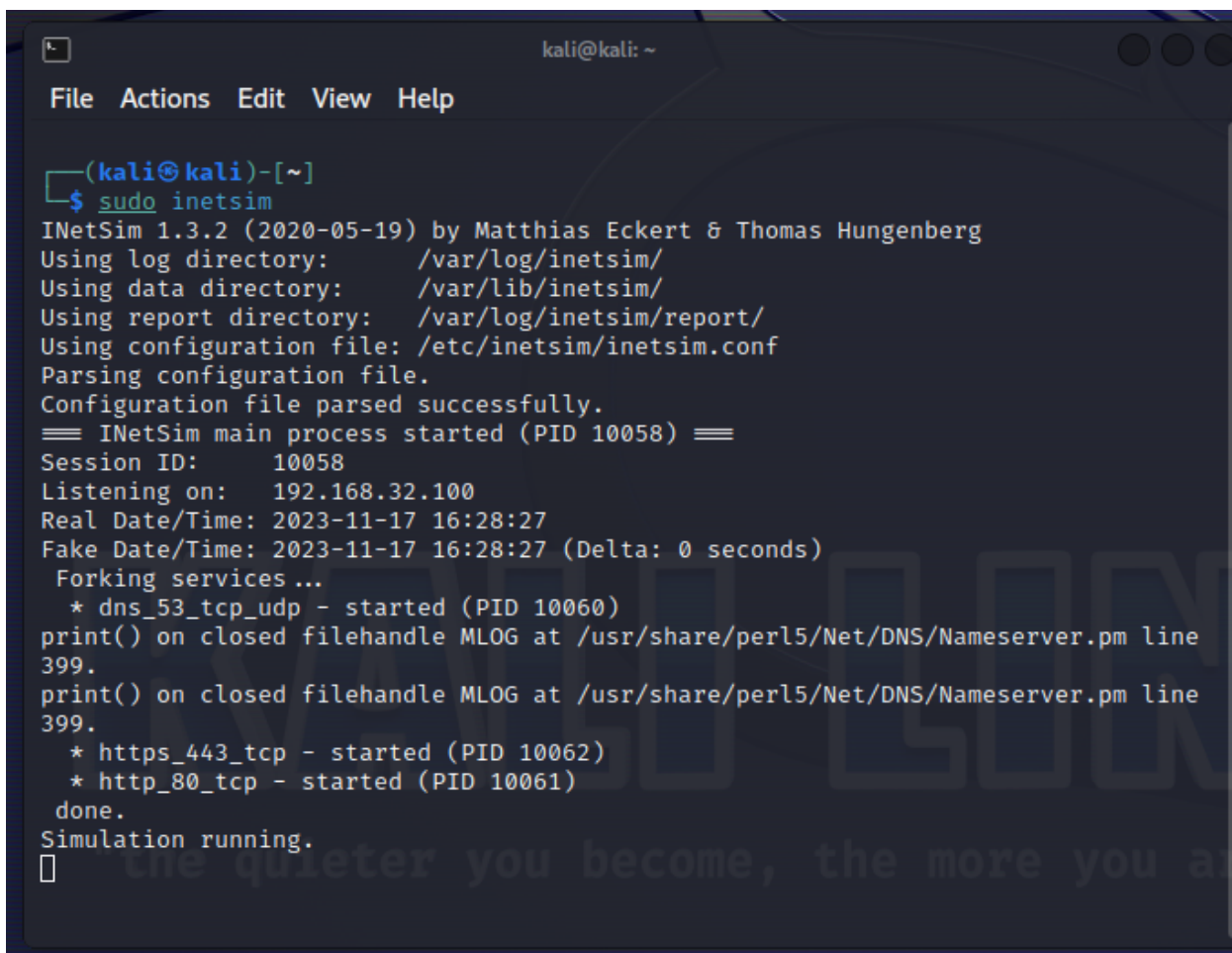
```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/inetsim/inetsim.conf  
  
#####  
# service_bind_address  
#  
# IP address to bind services to  
#  
# Syntax: service_bind_address <IP address>  
#  
# Default: 127.0.0.1  
#  
service_bind_address 192.168.32.100
```

- 3) Per risolvere l'hostname `epicode.internal`, andremo ad associare lo stesso all'indirizzo IP corrispondente al `service_bind_address`.

```
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 ftp.bar.net 10.10.20.30  
dns_static epicode.internal 192.168.32.100
```

## AVVIO SIMULAZIONE INETSIM

Dal terminale di Kali, digitando **sudo inetsim**, a questo punto partirà il client server da noi impostato.

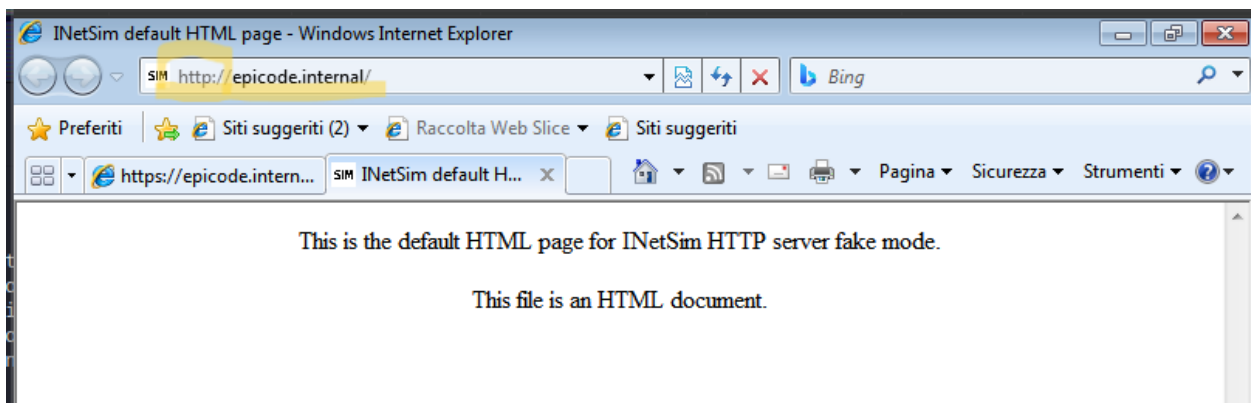


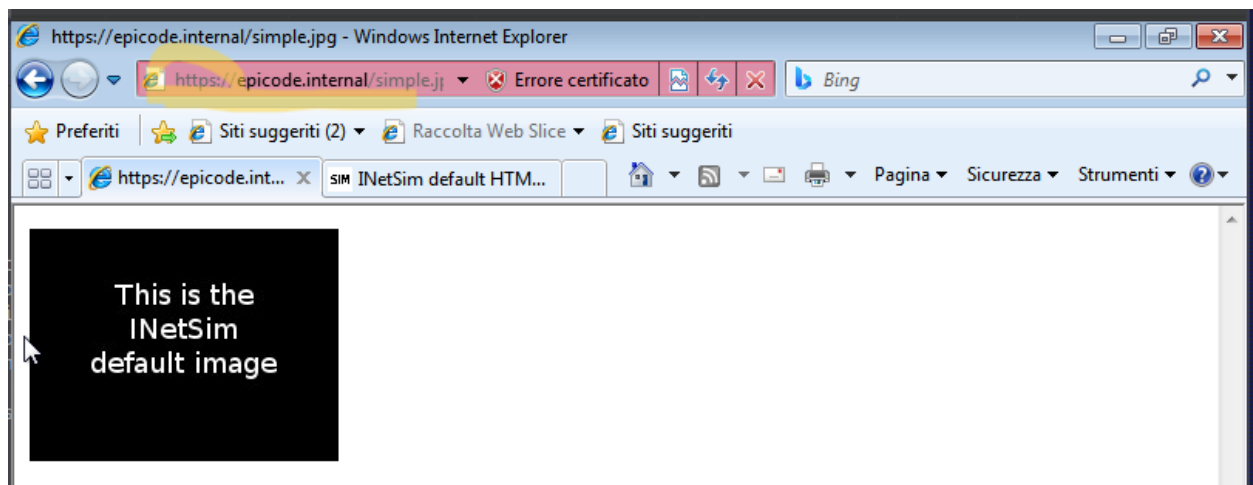
```
kali@kali: ~  
File Actions Edit View Help  
  
(kali@kali)-[~]  
$ sudo inetsim  
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg  
Using log directory: /var/log/inetsim/  
Using data directory: /var/lib/inetsim/  
Using report directory: /var/log/inetsim/report/  
Using configuration file: /etc/inetsim/inetsim.conf  
Parsing configuration file.  
Configuration file parsed successfully.  
=== INetSim main process started (PID 10058) ===  
Session ID: 10058  
Listening on: 192.168.32.100  
Real Date/Time: 2023-11-17 16:28:27  
Fake Date/Time: 2023-11-17 16:28:27 (Delta: 0 seconds)  
Forking services ...  
* dns_53_tcp_udp - started (PID 10060)  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm line 399.  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm line 399.  
* https_443_tcp - started (PID 10062)  
* http_80_tcp - started (PID 10061)  
done.  
Simulation running.  
█
```

Se avremo fatto tutto correttamente, sarà adesso possibile andare sul web browser di Windows7, in questo caso Internet Explorer, e provare sia con http che con https a collegarsi rispettivamente a:

- <https://epicode.internal>
- <http://epicode.internal>

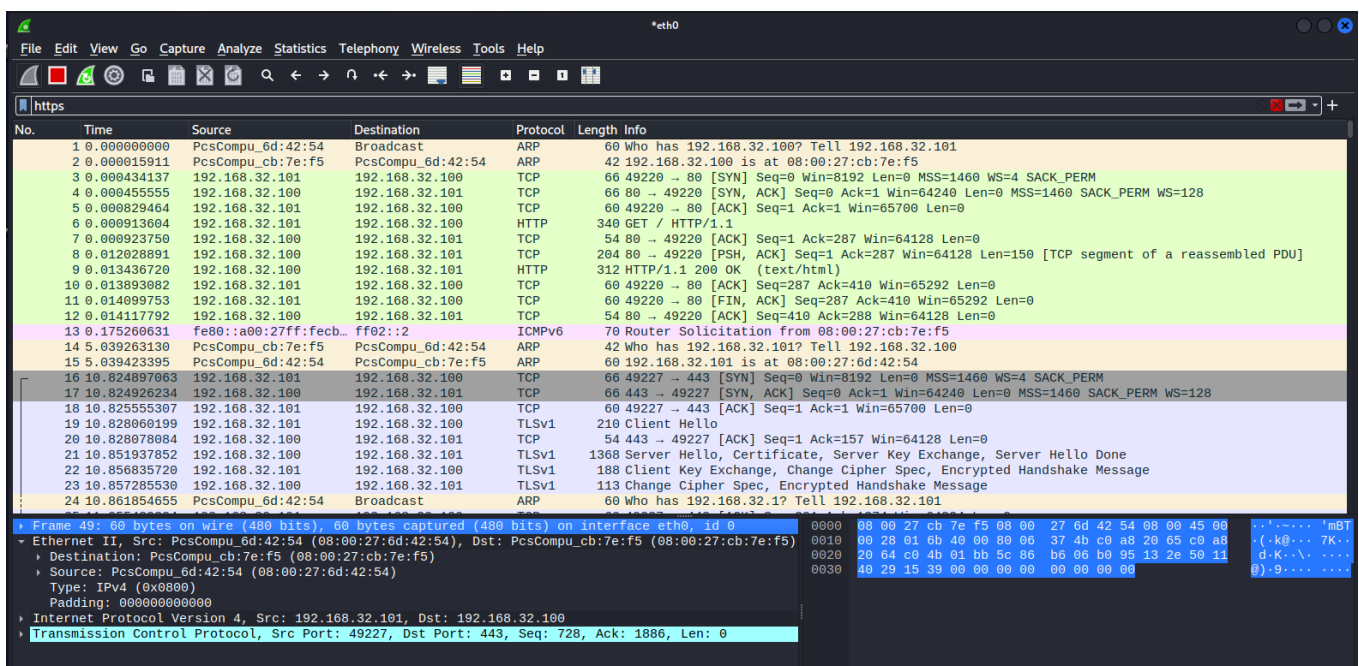
Il DNS risolverà correttamente la richiesta come da seguenti immagini:





## PACKET CAPTURE CON WIRESHARK

Il prossimo step consiste nel verificare la corretta trasmissione dei pacchetti tra Windows7 e Kali attraverso il client server appena creato, per cui andremo a rieseguire le query epicode.internal e a effettuare *packet capture* con il tool Wireshark preinstallato su Kali.







Dalla cattura, si potranno notare differenze sostanziali tra i protocolli di trasmissione https e http.

HTTP (Hypertext Transfer Protocol) e HTTPS (Hypertext Transfer Protocol Secure) sono protocolli utilizzati per la comunicazione tra un client (come un browser web) e un server web. La principale differenza tra i due è la sicurezza dei dati durante la trasmissione.

Quando si catturano pacchetti HTTP e HTTPS con Wireshark, si notano alcune differenze chiave:

### **1. Crittografia dei dati:**

- HTTP: I dati scambiati tra il client e il server sono trasferiti in chiaro, il che significa che sono leggibili da chiunque abbia accesso ai pacchetti catturati.

- HTTPS: I dati sono crittografati utilizzando un protocollo di sicurezza come TLS (Transport Layer Security) o SSL (Secure Sockets Layer). Questo significa che, anche se i pacchetti vengono catturati, i dati non sono facilmente leggibili senza la chiave di decrittografia corretta.

### **2. Porta di default:**

- HTTP: Solitamente utilizza la porta 80 per le comunicazioni non cifrate.

- HTTPS: Solitamente utilizza la porta 443 per le comunicazioni cifrate.



### **3. Protocollo utilizzato:**

- HTTP: Utilizza solo il protocollo HTTP per la comunicazione.
- HTTPS: Utilizza il protocollo HTTP all'interno di una sessione sicura fornita da TLS o SSL.

### **4. Intestazioni dei pacchetti:**

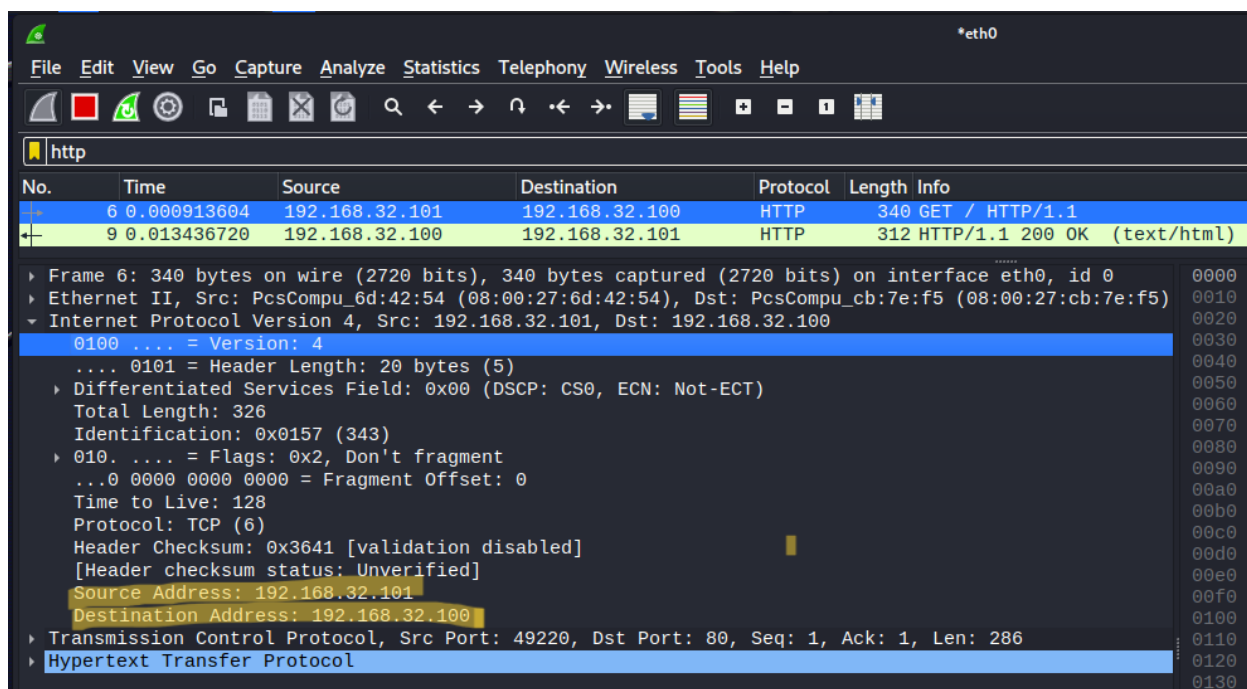
- HTTP e HTTPS: Contengono informazioni simili nelle intestazioni dei pacchetti, come i dettagli sulla richiesta, i cookie, gli user-agent, ecc. La differenza principale è che, mentre in HTTP queste informazioni sono trasmesse in chiaro, in HTTPS sono crittografate.

### **5. Metodi di sicurezza:**

- HTTP: Non offre meccanismi di sicurezza intrinseci.
- HTTPS: Offre un livello aggiuntivo di sicurezza tramite crittografia e autenticazione, garantendo che i dati trasmessi tra il client e il server siano sicuri e non manipolati.

## VERIFICA DEI MAC ADDRESS e SOURCE/DESTINATION

Analizzando i pacchetti catturati con Wireshark, vedremo come in tutti i casi gli indirizzi IP di source e destination saranno rispettivamente corrispondenti alla macchina Windows7 e alla Kali Linux.



Lo stesso varrà per i MAC address:

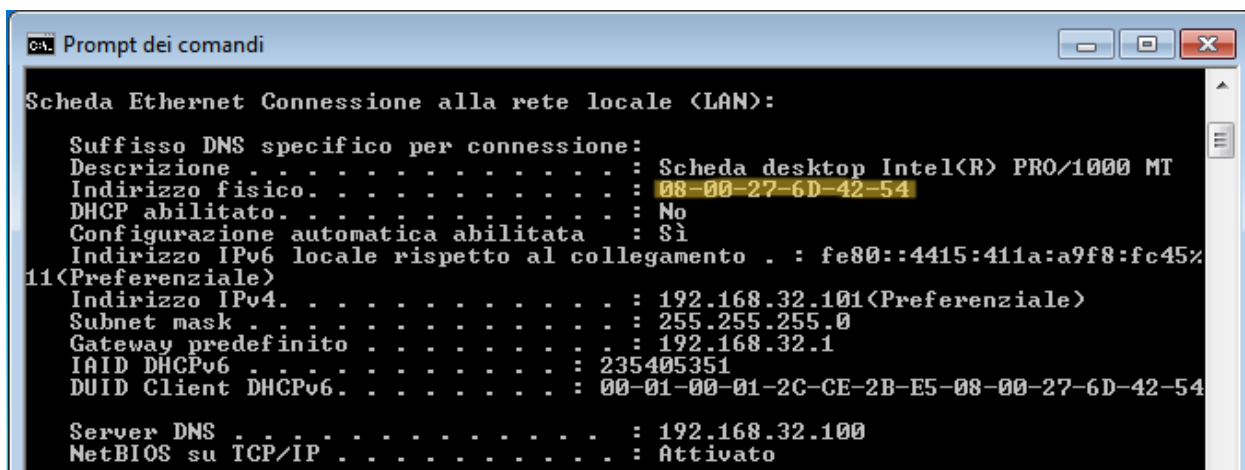
```

> Frame 49: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
- Ethernet II, Src: PcsCompu_6d:42:54 (08:00:27:6d:42:54), Dst: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5)
  Destination: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5)
  Source: PcsCompu_6d:42:54 (08:00:27:6d:42:54)
  Type: IPv4 (0x0800)
  Padding: 000000000000
> Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
> Transmission Control Protocol, Src Port: 49227, Dst Port: 443, Seq: 728, Ack: 1886, Len: 0

```

Per maggior riprova di quanto appreso, è possibile verificare i MAC address:

- attraverso Windows7 con la funzione `ipconfig /all`



```

C:\> Prompt dei comandi

Scheda Ethernet Connessione alla rete locale (LAN):

Suffisso DNS specifico per connessione:
Descrizione . . . . . : Scheda desktop Intel(R) PRO/1000 MT
Indirizzo fisico . . . . . : 08-00-27-6D-42-54
DHCP abilitato . . . . . : No
Configurazione automatica abilitata . . . . . : Sì
Indirizzo IPv6 locale rispetto al collegamento . . : fe80::4415:411a:a9f8:fc45%
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 . . . . . : 00-01-00-01-2C-CE-2B-E5-08-00-27-6D-42-54

Server DNS . . . . . : 192.168.32.100
NetBIOS su TCP/IP . . . . . : Attivato

```

- attraverso Kali con la finzione **ifconfig**

```

kali@kali: ~
File Actions Edit View Help
(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 0x20<link>
    ether 08:00:27:cb:7e:f5 txqueuelen 1000 (Ethernet)
    RX packets 364 bytes 32024 (31.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 586 bytes 55117 (53.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```