

Report M1W4D5

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:

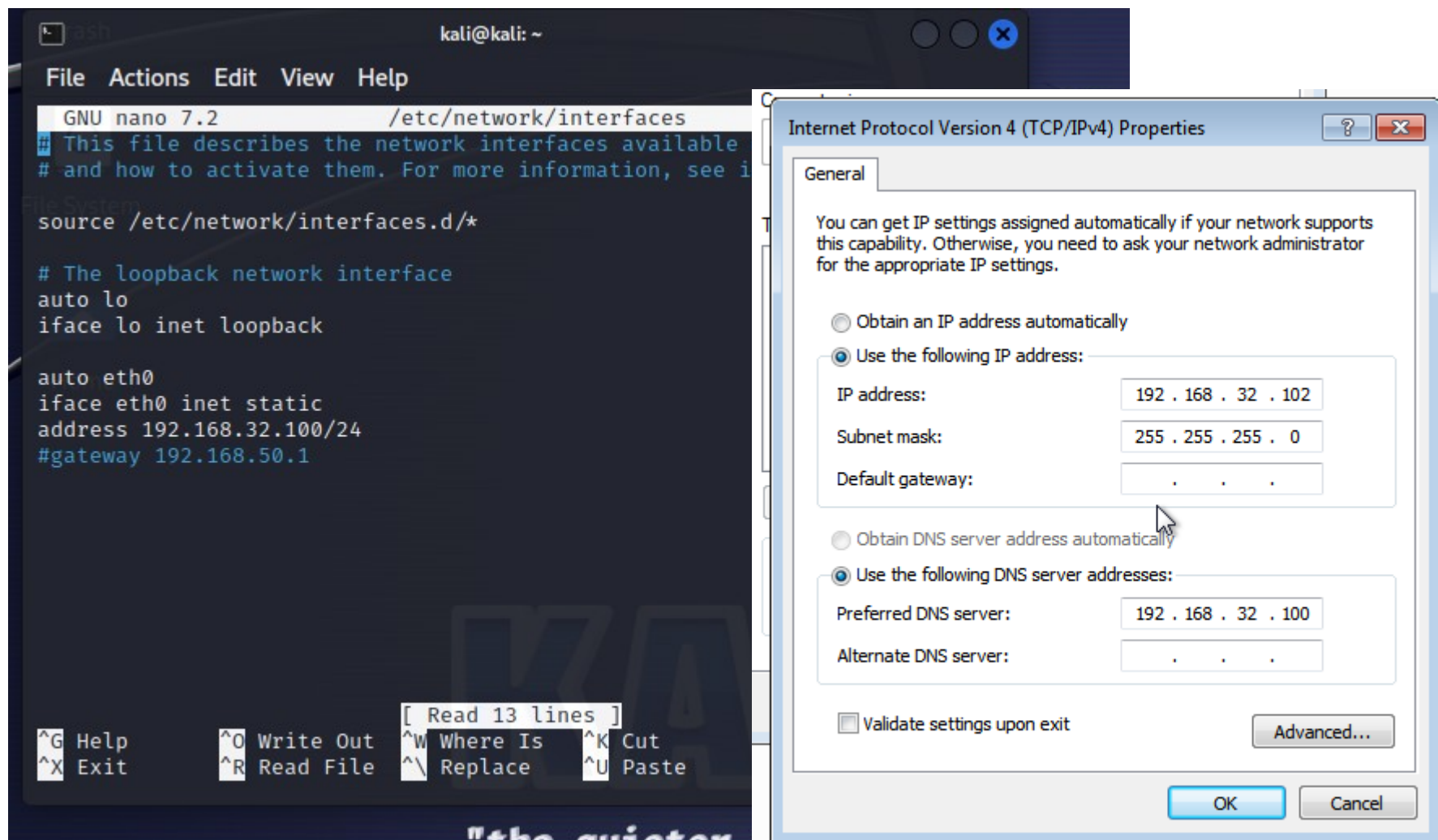
Simulare, in ambiente di laboratorio virtuale, un'architettura client server in cui un client con indirizzo 192.168.32.101 (Windows 7) richiede tramite web browser una risorsa all'hostname epicode.internal che risponde all'indirizzo 192.168.32.100 (Kali).

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.

- **CONFIGURAZIONI IP E INETSIM:**

Configuro le impostazioni di rete assegnando degli ip statici e facendo attenzione che l'indirizzo DNS di windows corrisponda all'indirizzo IP di kali per poter utilizzare il servizio di simulazione



In questo caso devo configurare anche il servizio DNS di inetsim

```
# time_udp, daytime_tcp, daytime_udp, echo_t
# echo_udp, discard_tcp, discard_udp, quotd_
# quotd_udp, chargen_tcp, chargen_udp, finge
# ident, syslog, dummy
# ftps, irc, https
#
start_service dns
start_service http
start_service https
start_service smtp
start_service smtps
start_service pop3
start_service pop3s
```

```
#####
# 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
```

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

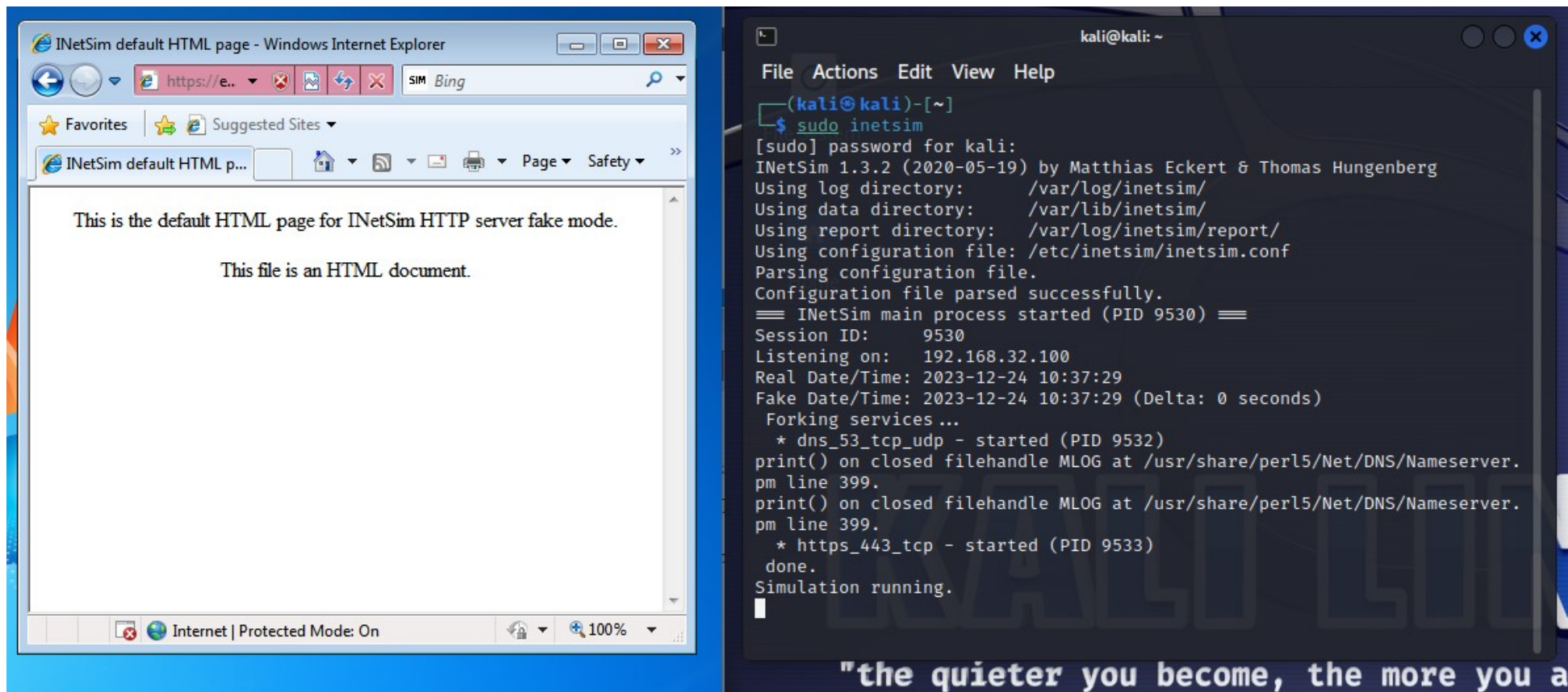
```
#####
# dns_default_domainname
#
# Default domain name to return with DNS
# Syntax: dns_default_domainname <domainname>
# Default: inetsim.org
#
dns_default_domainname epicode.internal
```

```
#####
# dns_static
#
# Static mappings for DNS
# Syntax: dns_static <fqdn hostname> <IP address>
# Default: none
#
dns_static epicode.internal 192.168.32.100
#dns_static ns1.foo.com 10.70.50.30
#dns_static ftp.bar.net 10.10.20.30
```

```
#####
#
# Syntax: https_default_fakefile <filename> <mimetype>
# Default: none
#
https_default_fakefile sample.html text/html
```

- **CATTURA PACCHETTI HTTPS CON WIRESHARK:**

Dopo aver avviato la simulazione Inetsim da terminale, apro wireshark su eth0 e carico la pagina <https://epicode.internal> su windows per catturarne i pacchetti



E quindi la cattura con wireshark:

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. 4), and the bottom pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.102	192.168.32.100	DNS	76	Standard query 0xd59c A epicode.internal
2	0.018762392	192.168.32.100	192.168.32.102	DNS	92	Standard query response 0xd59c A epicode.internal A 192.168.32.100
3	0.019610358	192.168.32.102	192.168.32.100	TCP	66	49170 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
4	0.019646652	192.168.32.100	192.168.32.102	TCP	66	443 → 49170 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK
5	0.019868711	192.168.32.102	192.168.32.100	TCP	60	49170 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
6	0.030111987	192.168.32.102	192.168.32.100	TLSv1	178	Client Hello
7	0.030130061	192.168.32.100	192.168.32.102	TCP	54	443 → 49170 [ACK] Seq=1 Ack=125 Win=64128 Len=0
8	0.124903765	192.168.32.100	192.168.32.102	TLSv1	1368	Server Hello, Certificate, Server Key Exchange, Server Hello Done
9	0.143725778	192.168.32.102	192.168.32.100	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Messa
10	0.143751103	192.168.32.100	192.168.32.102	TCP	54	443 → 49170 [ACK] Seq=1315 Ack=259 Win=64128 Len=0
11	0.144513317	192.168.32.100	192.168.32.102	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
12	0.226552604	fe80::1823:1acb:9ad...	ff02::1:3	LLMNR	84	Standard query 0x99b5 A wpad
13	0.226552813	192.168.32.102	224.0.0.252	LLMNR	64	Standard query 0x99b5 A wpad
14	0.335580223	fe80::1823:1acb:9ad...	ff02::1:3	LLMNR	84	Standard query 0x99b5 A wpad
15	0.335886952	192.168.32.102	224.0.0.252	LLMNR	64	Standard query 0x99b5 A wpad
16	0.349270280	192.168.32.100	192.168.32.102	TCP	113	[TCP Retransmission] 443 → 49170 [PSH, ACK] Seq=1315 Ack=259 Win=6
17	0.349591335	192.168.32.102	192.168.32.100	TCP	66	49170 → 443 [ACK] Seq=259 Ack=1374 Win=64324 Len=0 SLE=1315 SRE=13
18	0.539134331	192.168.32.102	192.168.32.255	NBNS	92	Name query NB WPAD<00>
19	1.288340721	192.168.32.102	192.168.32.255	NBNS	92	Name query NB WPAD<00>
20	2.039338561	192.168.32.102	192.168.32.255	NBNS	92	Name query NB WPAD<00>

Frame 4: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface
Ethernet II, Src: PcsCompu_cb:7e:f5 (08:00:27:cb:7e:f5), Dst: PcsCompu_7:
Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.102
0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 52
Identification: 0x0000 (0)
010. = Flags: 0x2, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: TCP (6)
Header Checksum: 0x78a9 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.32.100
Destination Address: 192.168.32.102
Transmission Control Protocol, Src Port: 443, Dst Port: 49170, Seq: 0, A

0000 08 00 27 71 e1 78 08 00 27 cb 7e f5 08 00 45 00 ...q.x...~...E
0010 00 34 00 00 40 00 40 06 78 a9 c0 a8 20 64 c0 a8 ...4..@.x...d..
0020 20 66 01 bb c0 12 c8 0a b7 14 8b ef 72 17 80 12 ...f.....r..
0030 fa f0 c2 41 00 00 02 04 05 b4 01 01 04 02 01 03 ...A.....
0040 03 07

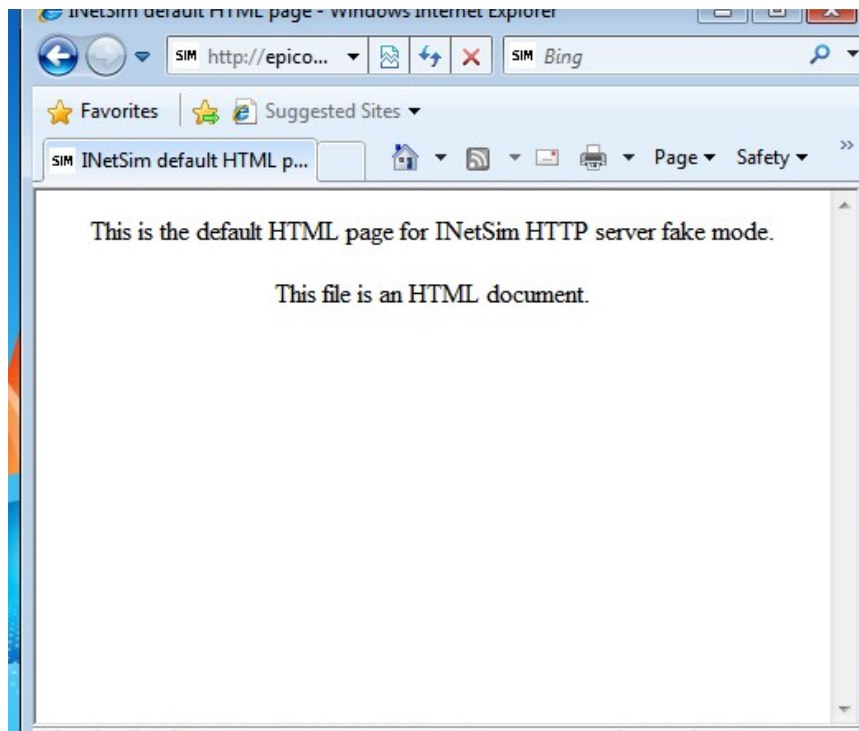
wireshark_eth06MX6F2.pcapng Packets: 115 · Displayed: 115 (100.0%) · Dropped: 0 (0.0%) Profile: Default

- **CATTURA PACCHETTI HTML CON WIRESHARK:**

Dopo aver modificato le impostazioni di simulazione ripeto il procedimento per l'indirizzo `http://epicode.internal`

```
# Available service names are:
# dns, http, smtp, pop3, tftp, ftp, ntp, tin
# time_udp, daytime_tcp, daytime_udp, echo_t
# echo_udp, discard_tcp, discard_udp, quotd
# quotd_udp, chargen_tcp, chargen_udp, finger
# ident, syslog, dummy_tcp, dummy_udp, smtps
# ftps, irc, https
#
start_service dns
start_service http
#start_service https
#start_service smtp
#start_service smtps
#start_service pop3
#start_service pop3s
```

```
#####
# http_default_fakefile
#
# The default fake file returned in fake mode if the f
# in the HTTP request does not match any of the extens
# defined above.
#
# The default fake file must be placed in <data-dir>/h
#
# Syntax: http_default_fakefile <filename> <mime-type>
#
# Default: none
#
http_default_fakefile sample.html text/html
```



```
File Actions Edit View Help
[sudo] password for kali:
(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 2195) ==
Session ID: 2195
Listening on: 192.168.32.100
Real Date/Time: 2023-12-24 10:50:44
Fake Date/Time: 2023-12-24 10:50:44 (Delta: 0 seconds)
Forking services...
* dns_53_tcp_udp - started (PID 2205)
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.
* http_80_tcp - started (PID 2206)
done.
Simulation running.
```

E quindi la cattura con wireshark:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.102	192.168.32.100	TCP	66	49204 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1
2	0.000025688	192.168.32.100	192.168.32.102	TCP	66	80 → 49204 [SYN, ACK] Seq=0 Ack=1 Win=64240
3	0.000269870	192.168.32.102	192.168.32.100	TCP	60	49204 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=
4	0.000433288	192.168.32.102	192.168.32.100	HTTP	381	GET /fwlink/?LinkId=69157 HTTP/1.1
5	0.000441716	192.168.32.100	192.168.32.102	TCP	54	80 → 49204 [ACK] Seq=1 Ack=328 Win=64128 Le
6	0.019048988	192.168.32.100	192.168.32.102	TCP	204	80 → 49204 [PSH, ACK] Seq=1 Ack=328 Win=641
7	0.021643554	192.168.32.100	192.168.32.102	HTTP	312	HTTP/1.1 200 OK (text/html)
8	0.021898941	192.168.32.102	192.168.32.100	TCP	60	49204 → 80 [ACK] Seq=328 Ack=410 Win=65292
9	0.027729849	192.168.32.102	192.168.32.100	TCP	60	49204 → 80 [FIN, ACK] Seq=328 Ack=410 Win=6
10	0.027750244	192.168.32.100	192.168.32.102	TCP	54	80 → 49204 [ACK] Seq=410 Ack=329 Win=64128
11	5.031233851	PcsCompu_cb:7e:f5	PcsCompu_71:e1:78	ARP	42	Who has 192.168.32.102? Tell 192.168.32.100
12	5.031558216	PcsCompu_71:e1:78	PcsCompu_cb:7e:f5	ARP	60	192.168.32.102 is at 08:00:27:71:e1:78
13	6.047728959	192.168.32.102	192.168.32.100	TCP	66	49205 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1
14	6.047755525	192.168.32.100	192.168.32.102	TCP	66	80 → 49205 [SYN, ACK] Seq=0 Ack=1 Win=64240
15	6.048019162	192.168.32.102	192.168.32.100	TCP	60	49205 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=
16	6.048189514	192.168.32.102	192.168.32.100	HTTP	472	GET / HTTP/1.1
17	6.048198331	192.168.32.100	192.168.32.102	TCP	54	80 → 49205 [ACK] Seq=1 Ack=419 Win=64128 Le
18	6.068021720	192.168.32.100	192.168.32.102	TCP	204	80 → 49205 [PSH, ACK] Seq=1 Ack=419 Win=641
19	6.070666020	192.168.32.100	192.168.32.102	HTTP	312	HTTP/1.1 200 OK (text/html)

Frame 16: 472 bytes on wire (3776 bits), 472 bytes captured (3776 bits) on interface

Ethernet II, Src: PcsCompu_71:e1:78 (08:00:27:71:e1:78), Dst: PcsCompu_cb:7e:f5 (08:

Internet Protocol Version 4, Src: 192.168.32.102, Dst: 192.168.32.100

Transmission Control Protocol, Src Port: 49205, Dst Port: 80, Seq: 1, Ack: 1, Len: 4

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n

Accept: application/x-ms-application, image/jpeg, application/xaml+xml, image/gif,

Accept-Language: en-US\r\n

User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0;

Accept-Encoding: gzip, deflate\r\n

Host: epicode.internal\r\n

Connection: Keep-Alive\r\n

\r\n

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

[HTTP request 1/1]

[Response in frame: 19]

Internet Protocol Version 4 (ip), 20 bytes

Packets: 64 · Displayed: 64 (100.0%)

Profile: Default

- **CONCLUSIONI:**

Si possono notare grandi differenze:

- **HTTPS:**
 - sono evidenti i pacchetti cifrati con protocollo TLSv1
 - i pacchetti LLMNR e NBNS sono utilizzati per la risoluzione degli host-name
 - viene richiesto dal browser un certificato di sicurezza che il simulatore non è in grado di fornire
- **HTTP:**
 - i pacchetti del protocollo HTTP non essendo cifrati mostrano in chiaro l'indirizzo richiesto