

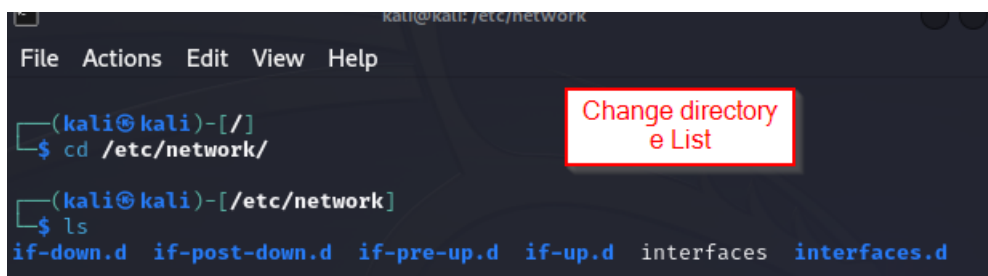
ESERCIZIO W4D4

L'esercizio seguente verte sull'utilizzo delle conoscenze acquisite durante le settimane di studio. Innanzitutto, prepariamo le macchine secondo i requisiti specificati in piattaforma:

- Kali Linux IP 192.168.32.100
- Windows 7 IP 192.168.32.101
- HTTPS server: attivato
- Servizio DNS per risoluzione nomi di dominio.

Impostazione indirizzo IP su Kali Linux

Avviamo la macchina virtuale Kali Linux per andare a settare l'indirizzo richiesto. Una volta caricato il sistema operativo, apriamo il terminal emulator. Ci sposteremo tra le varie cartelle fino anche non troveremo il file interfaces; Per fare ciò, useremo il comando `cd`(change directory) con annesso il percorso `/etc/network/`. Per essere sicuri di trovarci nel posto giusto, la direttiva `ls`(list) ci elencherà il contenuto della cartella network.



```
kali@kali: /etc/network
File Actions Edit View Help
(kali@kali)-[/]
$ cd /etc/network/
(kali@kali)-[/etc/network]
$ ls
if-down.d  if-post-down.d  if-pre-up.d  if-up.d  interfaces  interfaces.d
```

A red box highlights the text "Change directory e List" next to the `cd` command.

A questo punto, non ci resta che entrare nel file di testo interfaces. Normalmente esso non consente la scrittura ad una semplice interfaccia utente, per cui è necessario conferirci i diritti amministrativi. Accanto al comando editor text chiamato nano andrà affiancato dunque `sudo` per ricevere i privilegi. Il comando uscente sarà `sudo nano interfaces`. Inseriamo la password di default e siamo dentro.

```
kali@kali: /etc/network 80x24
GNU nano 7.2 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 192.168.32.100
```

Interfaccia configurazione Kali

Ci ritroveremo davanti una schermata del genere. Ovviamente il pannello di default necessiterà di un riempimento: Copriremo gli spazi vuoti con i dati in figura, andando a specificare l'indirizzo voluto, il range privato concessoci dalla netmask ed il default gateway per l'indirizzamento dei pacchetti sulla rete. Una volta scritto tutto, sovrascriviamo il file con ctrl+o e usciamo con ctrl+z. Già che stiamo ancora su Kali, cogliamo l'occasione per settare bene i servizi DNS e HTTP/HTTPS su inetsim.

Configurazione inetsim

```
(kali@kali)-[/etc/network]
$ cd ..

(kali@kali)-[/etc]
$ cd inetsim

(kali@kali)-[/etc/inetsim]
$ sudo nano inetsim.conf
```

Passaggio per trovare il file inetsim.conf

Come da figura, torniamo alla cartella /etc/ in modo da trovare la cartella inetsim. cd .. in questo caso sta per "un passo avanti nella gerarchia". Attiviamo nuovamente sudo nano mettendoci vicino il file che ci interessa: Inetsim.conf.

```
GNU nano 7.2                                inetsim.conf
# Default: none
#
# Available service names are:
# dns, http, smtp, pop3, tftp, ftp, ntp, time_tcp,
# time_udp, daytime_tcp, daytime_udp, echo_tcp,
# echo_udp, discard_tcp, discard_udp, quotd_tcp,
# quotd_udp, chargen_tcp, chargen_udp, finger,
# 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
#start_service ftps
#start_service tftp
#start_service irc
```

Configurazione Inetsim
start services

inetsim è un software utilizzato per simulare i comuni servizi di internet, come il caricamento di una pagina web, Appena entrati avremo un lungo corollario di funzioni con cui modificarlo. Cominciamo col rimuovere la griglia dai servizi che vogliamo emulare. Se un protocollo non ci interessa basterà mantenere il carattere per annullarlo. Lasciamo DNS, HTTP, HTTPS e scendiamo ancora più sotto.

```
#####
# 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
#####
# service_run_as_user
#
# User to run services
#
# Syntax: service_run_as_user <username>
#
```

Service_bin_address
configurazione

In questa sezione andremo invece ad inserire la macchina a cui legare i servizi di inetsim. Potremmo inserire 0.0.0.0 per indicare tutti gli indirizzi della rete, ma alla fine, è meglio specificare l'IP dell'host sul quale viene eseguito inetsim.

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

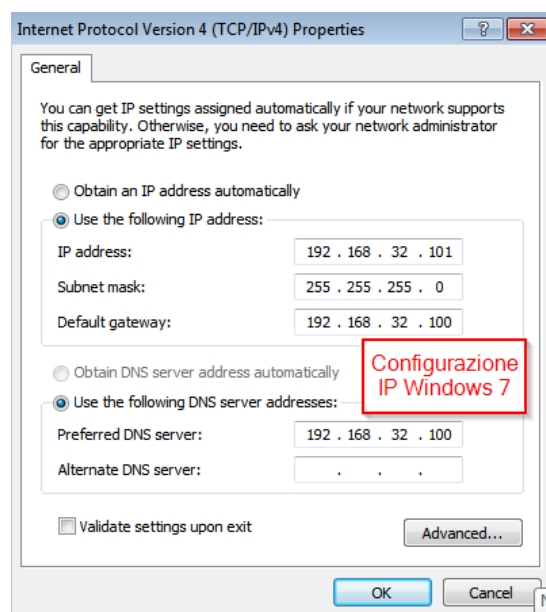
#####
# dns_version
#
# DNS version
#
```

dns_static
configurazione

Scorrendo ancora più in basso, è il momento di decidere a quale indirizzo assegnare il nome di dominio. Scegliamo l'accoppiata epicode.internal e 192.168.32.100. Come configurazione dovrebbe bastare, salviamo sempre con `ctrl+o` e usciamo con `ctrl+x`. Kali è pronto, ora tocca a Windows 7.

Impostazione indirizzo IP su Windows 7

Caricato il sistema operativo virtuale, usiamo la search bar di home per entrare in Network and Sharing Center → Local Area Connection. Successivamente clicchiamo su Properties per accedere alla lista dei protocolli. Facciamo doppio click su Internet Protocol Version 4 e caricheremo la finestra sottostante.



Il processo è simile a quello di Kali Linux. Inseriamo l'IP specificato per Windows 7, senza dimenticare di mettere lo stesso indirizzo sia per il gateway che per il DNS server. Abbiamo terminato i requisiti essenziali, ora tocca alle tracce seguenti.

Tracce esercizio W4D4

Traccia_1: 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).

Traccia_2: Si intercetti poi la comunicazione con Wireshark, evidenziando i MAC address di sorgente e destinazione ed il contenuto della richiesta HTTPS.

Traccia_3: 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.

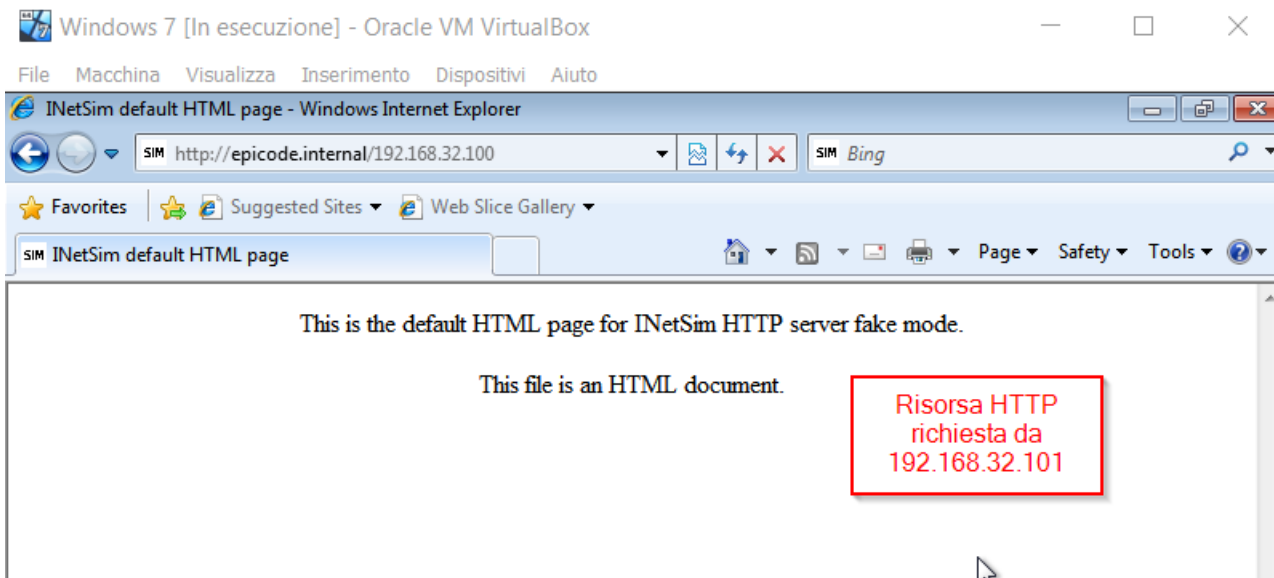
Traccia_1: Simulazione servizio web

Dopo aver settato gli IP e inetsim, usiamo il terminale su Kali per attivare inetsim. Il comando che ci servirà sarà `sudo inetsim`. Se tutto funziona non dovrebbe comparire nemmeno una scritta rossa di alert, ma solo started.

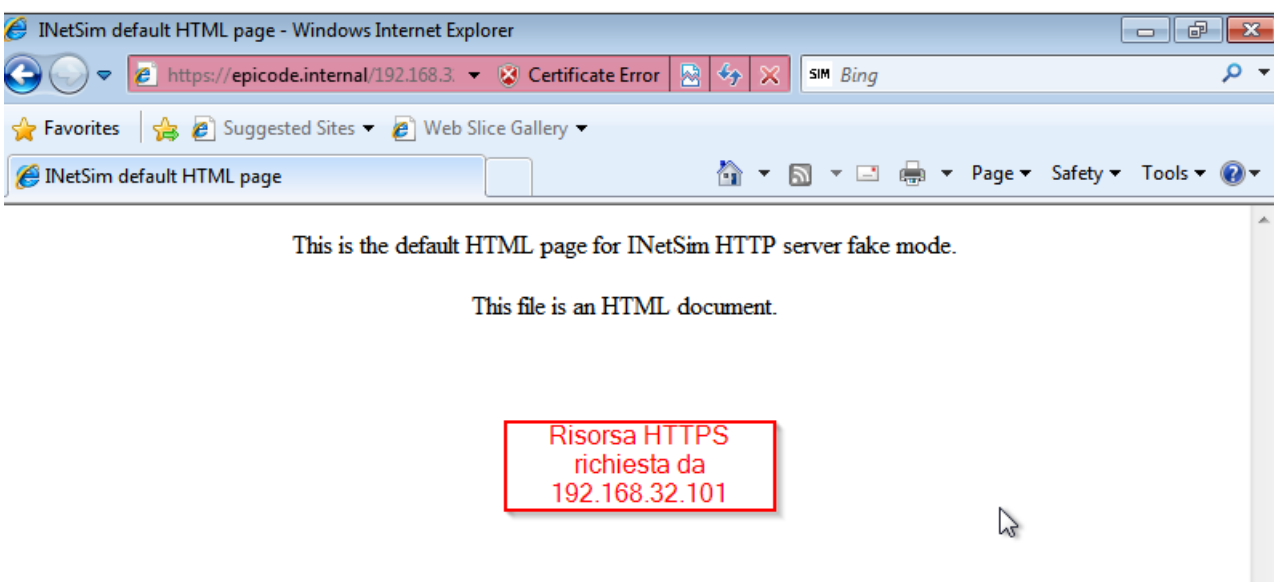
```
(kali@kali)-[/etc/network]
$ sudo inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory: /var/log/inetsim/
Using data directory: /var/lib/inetsim/
Using report directory: /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
== INetSim main process started (PID 8435) ==
Session ID: 8435
Listening on: 192.168.32.100
Real Date/Time: 2023-11-18 09:21:43
Fake Date/Time: 2023-11-18 09:21:43 (Delta: 0 seconds)
Forking services ...
* dns_53_tcp_udp - started (PID 8445)
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm l
ine 399.
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm l
ine 399.
* https_443_tcp - started (PID 8447)
* http_80_tcp - started (PID 8446)
done.
Simulation running.
```

Attivazione
servizi web

Come si evince dallo screenshot, il servizio è partito senza nessun problema. Tuttavia, bisogna vedere se Windows 7 riesce a risolvere la richiesta HTTP e ricevere una risposta HTTP da parte della macchina Kali. Facciamo partire il browser internet explorer su Windows ed inseriamo <http://epicode.internal/> come url.

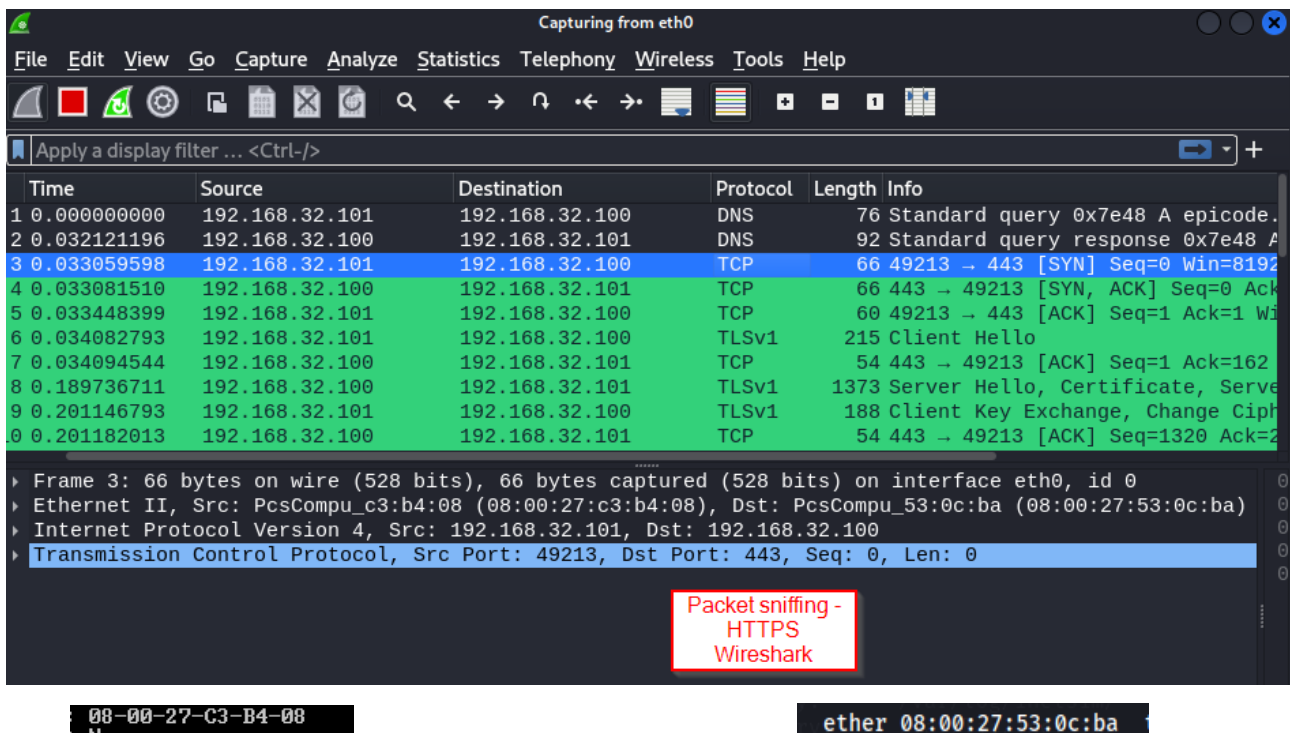


Constatiamo che la pagina viene caricata con successo. I due computer comunicano e si scambiano risorse. Risolta la Traccia_1, ci spostiamo alla Traccia_2. Carichiamo da Windows una pagina HTTPS nel frattempo.



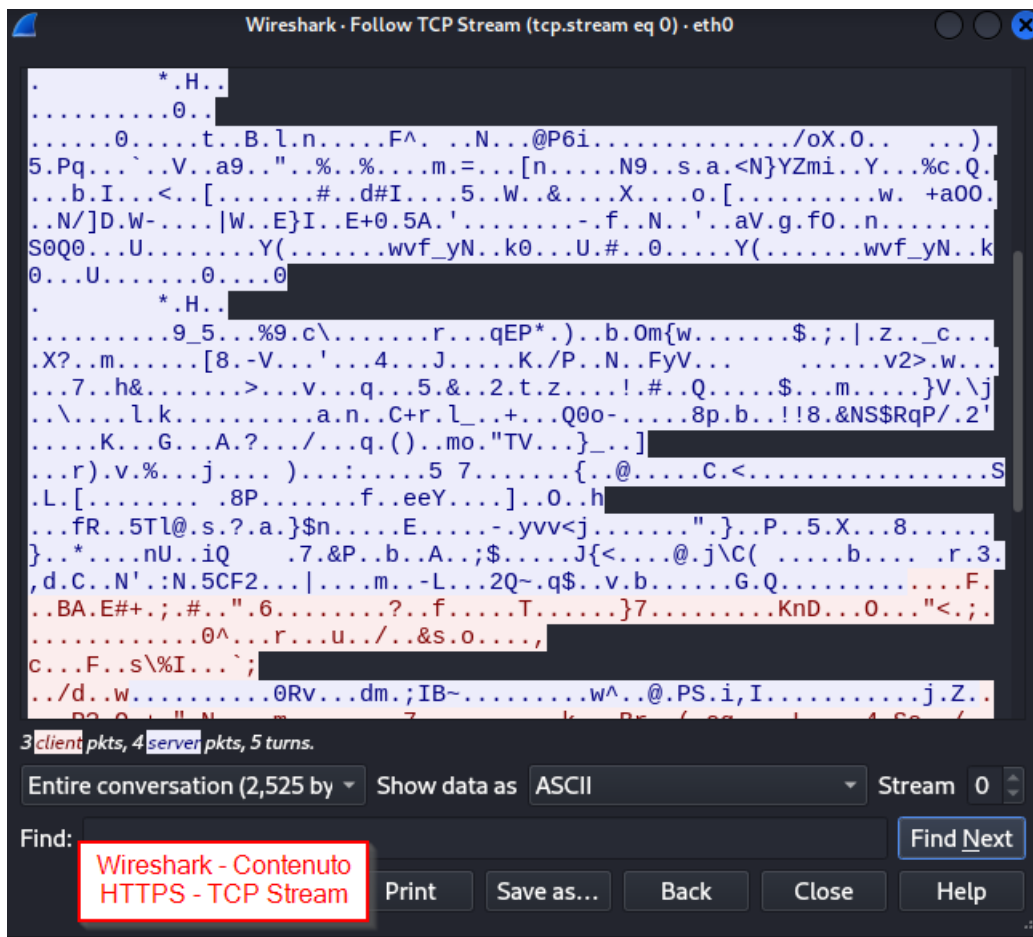
Traccia_2: Analisi pacchetti HTTPS con Wireshark

Wireshark è un software per analizzare protocolli, usato di solito per la soluzione di problemi di rete. Ci arriva nativamente con Kali Linux. Che sia dal terminal o dalla GUI, lo facciamo partire per incominciare il packet sniffing. Una volta dentro, scegliamo il traffico che passa per la scheda di rete eth0, quella che possiede la configurazione di Kali che abbiamo esposto prima.



Da questa schermata possiamo osservare meglio quello che accade dietro le quinte. Il MAC address usato durante gli scambi lungo il canale TCP corrispondono a quello del Windows, a sinistra, e Kali, a destra.

Osserviamo come i due computer inizializzano e portano a termine il Three Way Handshake, seguito da Client Hello, Server Hello scambi di certificati, chiavi etc. Sopra c'è persino la richiesta e la risposta DNS, che associa il nome digitato sul browser all'IP specificato. Per quanto riguarda il contenuto della richiesta HTTPS, ovviamente ci arriva tutto criptato come nella figura sottostante.



Traccia_3: analisi pacchetti HTTP Wireshark e confronto

La Traccia_3, al contrario, vuole che si mostra uno scambio di pacchetti per mezzo del servizio HTTP. Il procedimento resta molto simile.

Ricarichiamo la pagina con protocollo HTTP e prepariamo Wireshark. A differenza di HTTPS, qui ci è concesso vedere e analizzare sia la richiesta con il metodo GET/ HTTP che la risposta HTTP / 200 OK, inoltre, manca totalmente la suite di crittografia. Tutte le informazioni ci arrivano in chiaro, ad esempio Accept-Language IT, Connection Keep Alive, il corpo HTML del testo in pagina etc. Lascio prima l'analisi dei pacchetti e subito dopo il contenuto TCP stream.

Wireshark interface showing a packet capture on eth0. The packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.101	192.168.32.100	TCP	66	49468 → 80 [SYN] Seq=0 Win=8192
2	0.000030344	192.168.32.100	192.168.32.101	TCP	66	80 → 49468 [SYN, ACK] Seq=0 Ack=
3	0.000387181	192.168.32.101	192.168.32.100	TCP	60	49468 → 80 [ACK] Seq=1 Ack=1 Win=
4	0.000656260	192.168.32.101	192.168.32.100	HTTP	361	GET / HTTP/1.1
5	0.000667219	192.168.32.100	192.168.32.101	TCP	54	80 → 49468 [ACK] Seq=1 Ack=308 Win=
6	0.031937367	192.168.32.100	192.168.32.101	TCP	204	80 → 49468 [PSH, ACK] Seq=1 Ack=
7	0.036690069	192.168.32.100	192.168.32.101	HTTP	312	HTTP/1.1 200 OK (text/html)
8	0.037113658	192.168.32.101	192.168.32.100	TCP	60	49468 → 80 [ACK] Seq=308 Ack=410 Win=
9	0.037302682	192.168.32.101	192.168.32.100	TCP	60	49468 → 80 [FIN, ACK] Seq=308 Ack=
10	0.037318448	192.168.32.100	192.168.32.101	TCP	54	80 → 49468 [ACK] Seq=410 Ack=308

Packet details for Frame 1:

- Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface eth0, id 0
- Ethernet II, Src: PcsCompu_c3:b4:08 (08:00:27:c3:b4:08), Dst: PcsCompu_53:0c:ba (08:00:27:53:0c:ba)
- Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
- Transmission Control Protocol, Src Port: 49468, Dst Port: 80, Seq: 0, Len: 0

Wireshark-HTTP-MAC address

08-00-27-C3-B4-08

ether 08:00:27:53:0c:ba

Wireshark - Follow TCP Stream (tcp.stream eq 0) - eth0

```

GET / HTTP/1.1
Accept: */*
Accept-Language: it-IT
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0)
Accept-Encoding: gzip, deflate
Host: epicode.internal
Connection: Keep-Alive

HTTP/1.1 200 OK
Content-Type: text/html
Connection: Close
Date: Fri, 17 Nov 2023 23:06:58 GMT
Server: INetSim HTTP Server
Content-Length: 258

<html>
<head>
<title>INetSim default HTML page</title>
</head>
<body>
<p></p>
<p align="center">This is the default HTML page for INetSim HTTP server fake mode.</p>

```

Packet 4. 1 client pkt, 2 server pkts, 1 turn. Click to select.

Entire conversation (715 bytes) Show data as ASCII Stream 0

Wireshark - Contenuto HTTP-TCP Stream