

Progetto M1 Cybersecurity Analyst

LABORATORIO VIRTUALE
KALI LINUX/WINDOWS 7

Made by Sciatella Andrea

Requisiti e servizi:



Impostare IP Kali
linux:
192.168.32.100/24

Impostare IP
Windows 7:
192.168.32.101/24

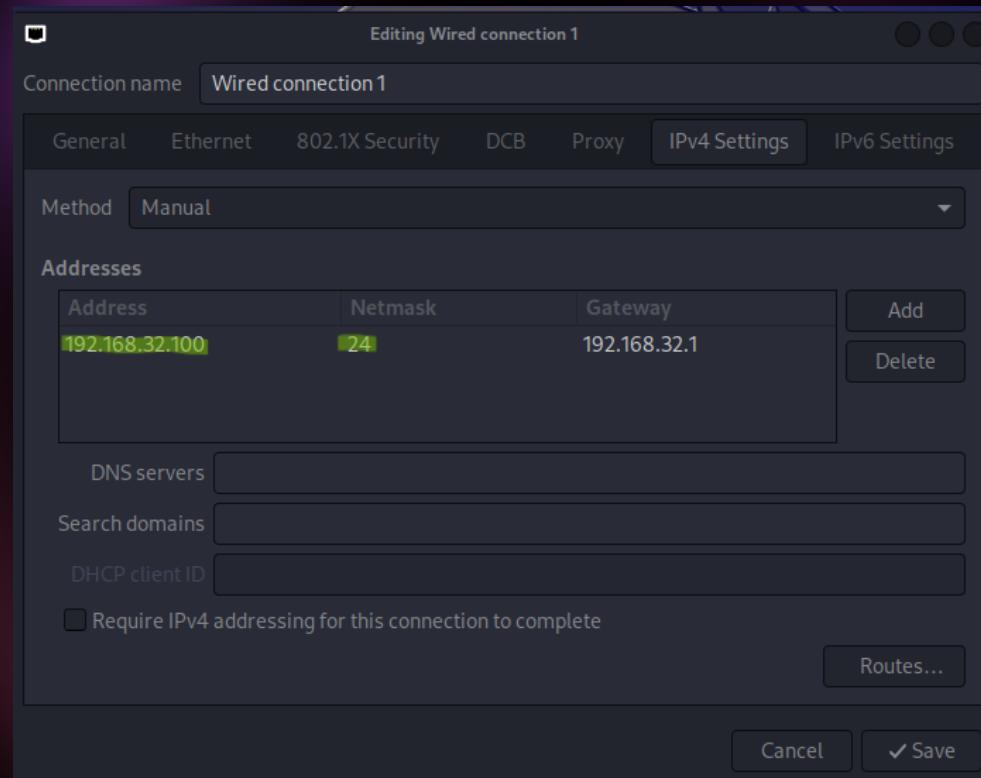
Servizio DNS per
risoluzione nomi di
dominio: attivo

HTTPS server: attivo

Settare IP statico a Kali Linux

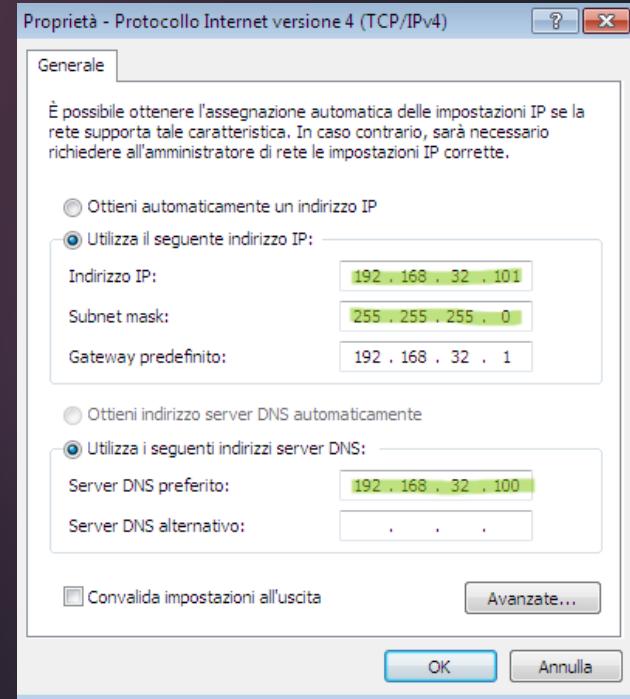
PRIMA DI TUTTO, DOBBIAMO ANDARE A SISTEMARE ALCUNE COSE PER FAR FUNZIONARE IL LABORATORIO:

MODIFICHEREMO L'INDIRIZZO IP DELLA VM DI KALI IN UN **INDIRIZZO STATICO.**



Settare IP statico a Windows 7:

SUCCESSIVAMENTE, RIPETEREMO LA MEDESIMA OPERAZIONE CON WINDOWS 7, ANCHE SE IN QUESTO CASO AVRÀ BISOGNO DI UN'ACCORTEZZA IN PIÙ: NELL'
SERVER DNS PREFERITO
ANDREMO AD INSERIRE **L'IP DELLA VM KALI LINUX.**

A screenshot of a Windows 7 Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The user runs 'ipconfig' and the output shows the following network configuration:

```
C:\> ipconfig
Windows IP Configuration

Scheda Ethernet Connessione alla rete locale (LAN):
  Suffixo DNS specifico per connessione: 
  Indirizzo IPv6 locale rispetto al collegamento . : fe80::8c4a:34eb:310a:5d9bz
  Indirizzo IPv4. . . . . : 192.168.32.101
  Subnet mask . . . . . : 255.255.255.0
  Gateway predefinito . . . . . : 192.168.32.1

Scheda Tunnel isatap.{2BFB1146-5C81-4312-AB9C-44B56BE7580F}:
  Stato supporto. . . . . : Supporto disconnesso
  Suffixo DNS specifico per connessione: 

C:\>
```

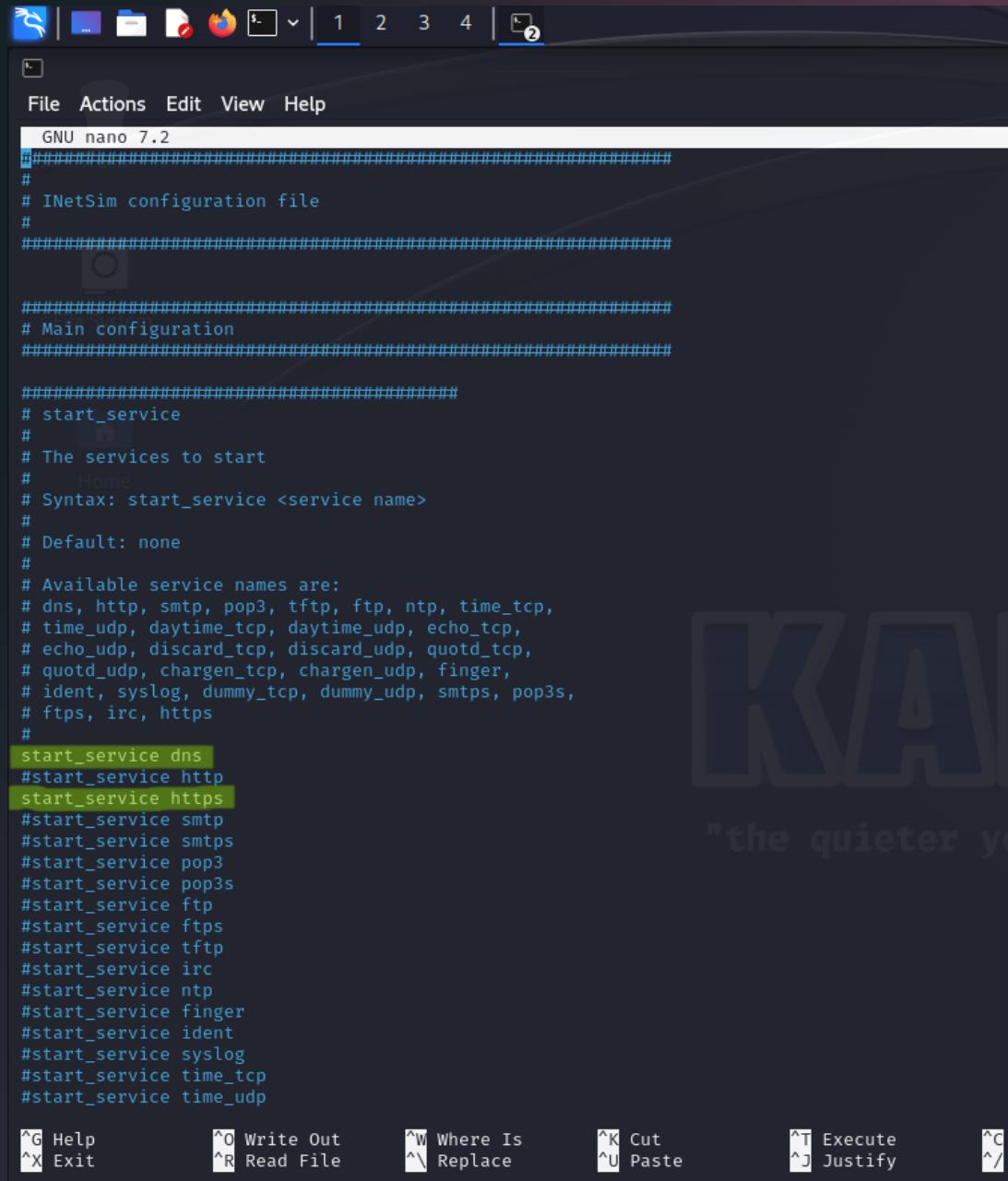
The IP address, subnet mask, and gateway are highlighted in green.

Configurazione del tool Inetsim:

Per la preparazione dell'emulatore di un web server, utilizzeremo **Inetsim** (tool open source per la simulazione di servizi Internet standard, preinstallato in Kali Linux).

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

Con il comando «**sudo nano /etc/inetsim/inetsim.conf**» andremo prima ad utilizzare i privilegi di amministratore immettendo la password, per poi utilizzare **Nano**, un editor di testo per modificare le impostazioni del tool.



The screenshot shows a terminal window titled "GNU nano 7.2" displaying the InetSim configuration file. The file contains several comments and service definitions. The services listed 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. The services "http", "https", and "start_service https" are highlighted in green, indicating they are currently active or being modified.

```
File Actions Edit View Help
GNU nano 7.2
#
# INetSim configuration file
#
#####
#
## Main configuration
#####

#####
# start_service
#
# The services to start
# Home
# Syntax: start_service <service name>
#
# 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
#start_service ntp
#start_service finger
#start_service ident
#start_service syslog
#start_service time_tcp
#start_service time_udp

^G Help      ^O Write Out    ^W Where Is      ^K Cut        ^T Execute     ^C
^X Exit      ^R Read File    ^\ Replace      ^U Paste      ^J Justify    ^/

```

Configurazione del tool Inetsim:

Una volta entrati nel file .conf, andremo a cambiare le impostazioni del **HTTPS**, **HTTP** e **DNS** utilizzando i dati delle macchine prima inseriti. Nel file andremo a **rimuovere** e/o ad **aggiungere** il simbolo «#» davanti ai servizi a noi necessari. In questo caso abbiamo lasciato disponibili solo i servizi **DNS** e **HTTPS** necessari alla prima parte del laboratorio.

Configurazione del tool Inetsim:

I prossimi settings da cambiare sono quelli riguardanti il **DNS (Domain Name System)**: quindi attiveremo la porta in cui il **DNS** starà in ascolto (53 di default), e infine assegneremo il nome del domain che utilizzeremo (*epicode.internal*) e l'indirizzo IP di Kali Linux nella sezione **dns static**.

Sniffing con Wireshark:



Prima di attivare il tool per l'emulazione, andremo ad attivare e tenere a portata di mano **Wireshark**.

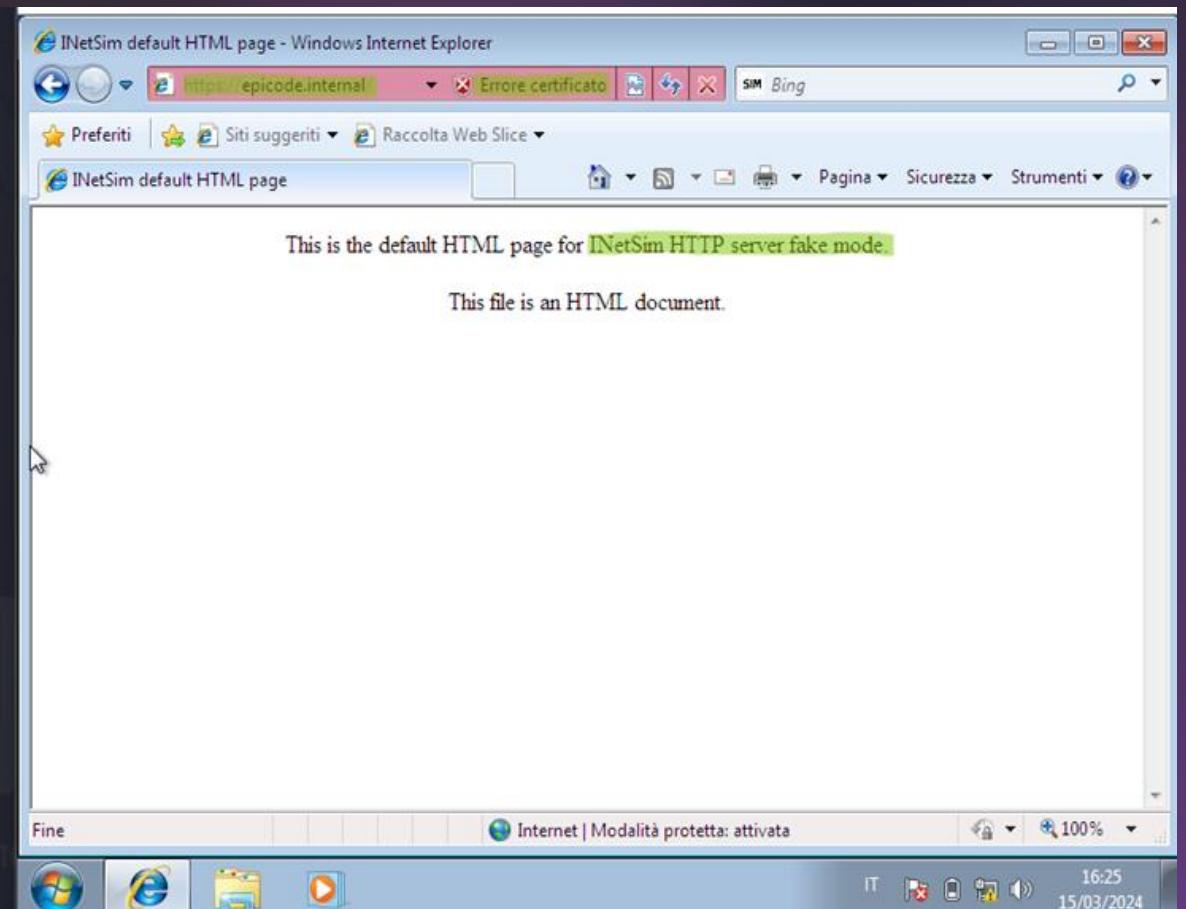
Wireshark è un software creato **per analisi di protocollo** e concepito come **packet sniffer** (letteralmente *annusa pacchetti*). Con questo tool andremo ad **analizzare nei dettagli il traffico dati** che verrà generato dalle nostre VM.

Il software è preinstallato in Kali Linux e usa una **GUI** (*Graphic User Interface* o *interfaccia grafica*) per semplificare la lettura del processo di *sniffing*.

Eseguire Inetsim:

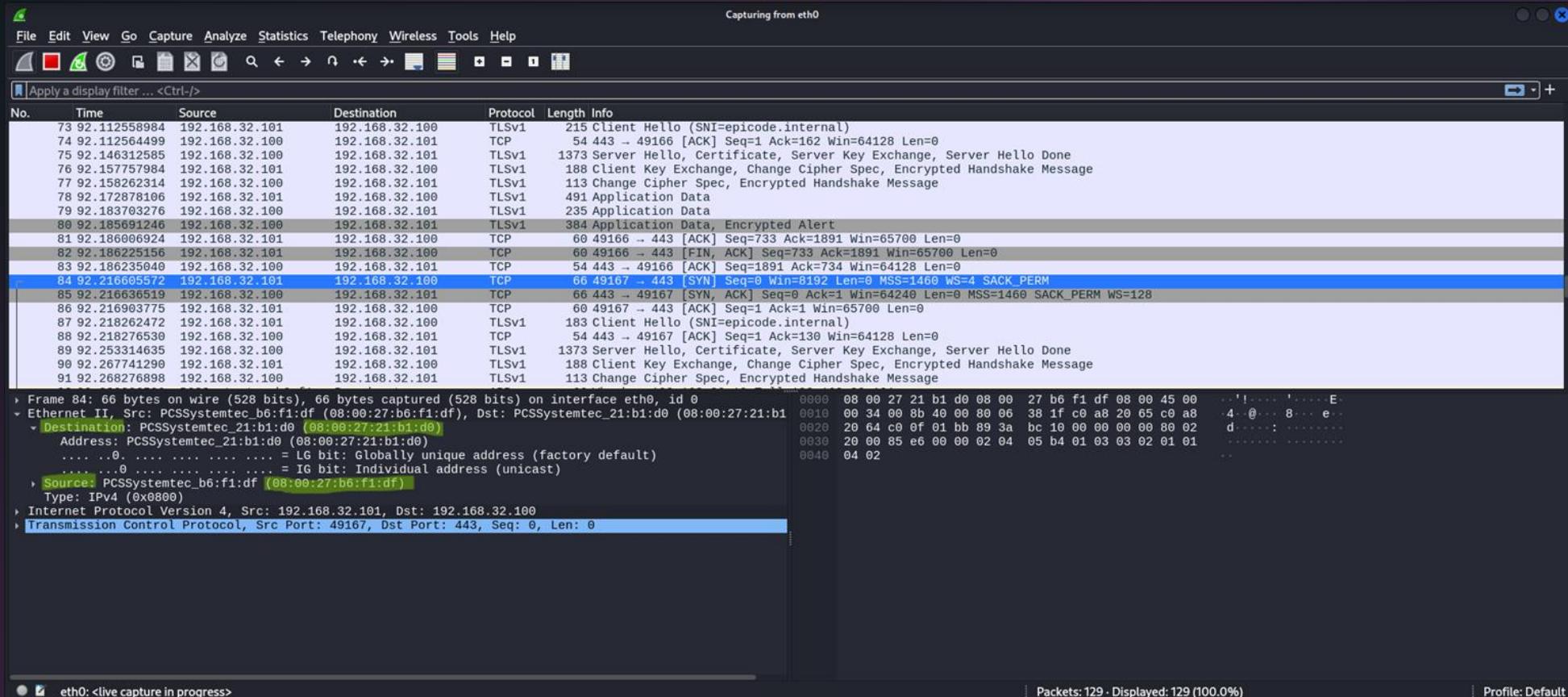
Dopo aver configurato i nostri strumenti, possiamo finalmente eseguire inetsim con «**sudo inetsim**» e mandare «*online*» il nostro sito fittizio. Inserendo «***https://epicode.internal***» in Internet Explorer di Windows 7 verrà caricata la pagina con il protocollo **HTTPS** selezionato prima, ma priva di certificazione **SSL o TLS valida**.

```
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 24371) ==  
Session ID: 24371  
Listening on: 192.168.32.100  
Real Date/Time: 2024-03-15 11:23:02  
Fake Date/Time: 2024-03-15 11:23:02 (Delta: 0 seconds)  
Forking services ...  
* dns_53_tcp_udp - started (PID 24373)  
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 24374)  
done.  
Simulation running.
```



Pacchetti HTTPS:

Una volta effettuato l'accesso alla pagina, **Wireshark rileva tutti i pacchetti possibili**. Come possiamo vedere nella foto, abbiamo catturato pacchetti **TCP** compresa la **Three-Way Handshake** (per creare una connessione tra client e server) e **TLS(v1)**.



Pacchetti HTTPS:

CERCANDO NEI DATI CRITTOGRAFATI POSSIAMO NOTARE CHE NEL PAYLOAD, COMPARA IL **DOMINIO INSERITO PRECEDENTEMENTE** NELLA CONFIGURAZIONE DEL DNS, DATI CHE COMPAGNO IN CHIARO NEL PROTOCOLLO **HTTP**.

The screenshot shows a Wireshark capture of an HTTPS session. The packet list pane displays 110 captured and 110 displayed packets. The selected packet is a TLSv1 Change Cipher Spec message (No. 71) from 192.168.32.101 to 192.168.32.100. The details pane shows the TLS handshake process, including the Client Hello (SNI=epicode.internal), Server Hello, and Change Cipher Spec messages. The bytes pane shows the raw hex and ASCII data of the selected packet, where the server name 'epicode.internal' is clearly visible in the ASCII dump.

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Capturing from eth0

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
61	33.567170428	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
62	33.581588658	192.168.32.101	192.168.32.100	TLSv1	491	Application Data
63	33.592529578	192.168.32.100	192.168.32.101	TLSv1	235	Application Data
64	33.594540026	192.168.32.100	192.168.32.101	TLSv1	384	Application Data, Encrypted Alert
65	33.594815568	192.168.32.101	192.168.32.100	TCP	60	49166 → 443 [ACK] Seq=733 Ack=1891 Win=65700 Len=0
66	33.595119897	192.168.32.101	192.168.32.100	TCP	60	49166 → 443 [FIN, ACK] Seq=733 Ack=1891 Win=65700 Len=0
67	33.595313874	192.168.32.100	192.168.32.101	TCP	54	443 → 49166 [ACK] Seq=1891 Ack=734 Win=64128 Len=0
68	33.636693227	192.168.32.101	192.168.32.100	TCP	66	49167 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
69	33.636734596	192.168.32.100	192.168.32.101	TCP	66	443 → 49167 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
70	33.637138281	192.168.32.101	192.168.32.100	TCP	60	49167 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
71	33.638513558	192.168.32.101	192.168.32.100	TLSv1	183	Client Hello (SNI=epicode.internal)
72	33.638524417	192.168.32.100	192.168.32.101	TCP	54	443 → 49167 [ACK] Seq=1 Ack=130 Win=64128 Len=0
73	33.669055577	192.168.32.100	192.168.32.101	TLSv1	1373	Server Hello, Certificate, Server Key Exchange, Server Hello Done
74	33.679710093	192.168.32.101	192.168.32.100	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
75	33.680209904	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
76	33.702810063	PCSSystemtec_b6:f1:... Broadcast		ARP	60	Who has 192.168.32.1? Tell 192.168.32.101
77	33.884129189	192.168.32.100	192.168.32.101	TCP	113	[TCP Retransmission] 443 → 49167 [PSH, ACK] Seq=1320 Ack=264 Win=64128 Len=59
78	33.885193288	192.168.32.101	192.168.32.100	TCP	66	49167 → 443 [ACK] Seq=264 Ack=1379 Win=64320 Len=0 SLE=1320 SRE=1379
79	34.607300591	PCSSystemtec_b6:f1:... Broadcast		ARP	60	Who has 192.168.32.1? Tell 192.168.32.101

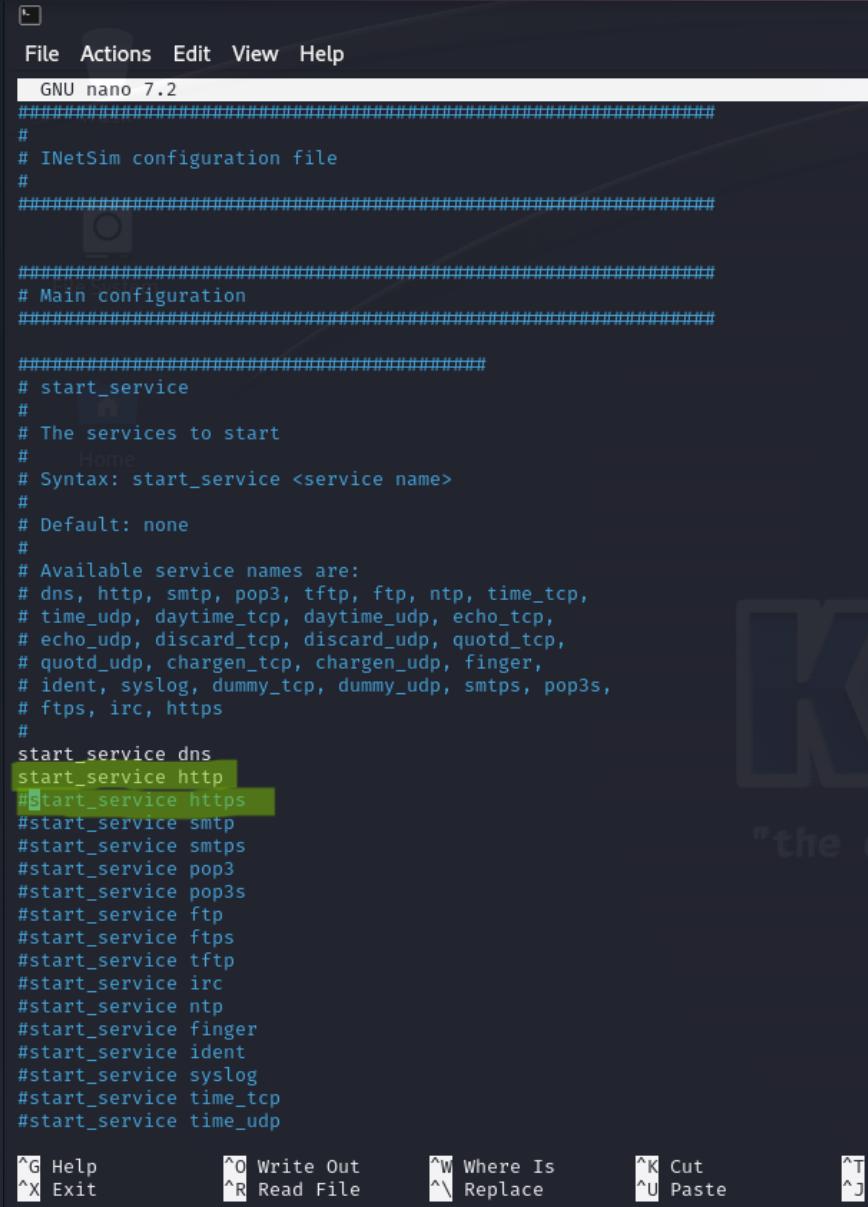
Version: TLS 1.0 (0x0301)
Random: 65f602d3c982bec3d7d575bfd786586fe08f2096223c0b71b7c82e9cae4dbbc
Session ID Length: 0
Cipher Suites Length: 24
Cipher Suites (12 suites)
Compression Methods Length: 1
Compression Methods (1 method)
Extensions Length: 55
Extension: renegotiation_info (len=1)
Extension: server_name (len=21) name=epicode.internal
Type: server_name (0)
Length: 21
Server Name Indication extension
Server Name list length: 19
Server Name Type: host_name (0)
Server Name length: 16
Server Name: epicode.internal
Extension: status_request (len=5)
Extension: supported_groups (len=6)
Type: supported_groups (10)

0000 08 00 27 21 b1 d0 08 00 27 b6 f1 df 08 00 45 00 .!.' E.
0010 00 a9 00 7a 40 00 80 06 37 bb c0 a8 20 65 c0 a8 ..z@.7.e..
0020 20 64 c0 0f 01 bb 21 4a c0 20 82 7f 2b b7 50 18 d..!J.+P.
0030 40 29 64 bf 00 00 16 03 01 00 7c 01 00 00 78 03 @)d..|..x.
0040 01 65 f6 02 d3 3c 98 2b ec 3d 7d 57 5b fd 78 65 .e..<+.=]W[.xe
0050 86 fe 08 f2 09 62 23 c0 b7 1b 7c 82 e9 ca e4 db ..b#.||....
0060 bc 00 00 18 02 f0 00 35 00 05 00 0a c0 13 c0 14/.5.....
0070 c0 09 c0 0a 00 32 00 38 00 13 00 04 01 00 00 37 ..2.8.....7
0080 ff 01 00 01 00 00 00 00 15 00 13 00 00 10 65 70ep
0090 69 63 6f 64 65 2e 69 6e 74 65 72 6e 61 6c 00 05 icode.in tErnal..
00a0 00 05 01 00 00 00 00 00 a0 00 06 00 04 00 17 00
00b0 18 00 0b 00 02 01 00

Server Name (tls.handshake.extensions_server_name), 16 bytes

Packets: 110 · Displayed: 110 (100.0%)

Profile: Default



```
File Actions Edit View Help
GNU nano 7.2
#####
# INetSim configuration file
#
#####

#####
# Main configuration
#####

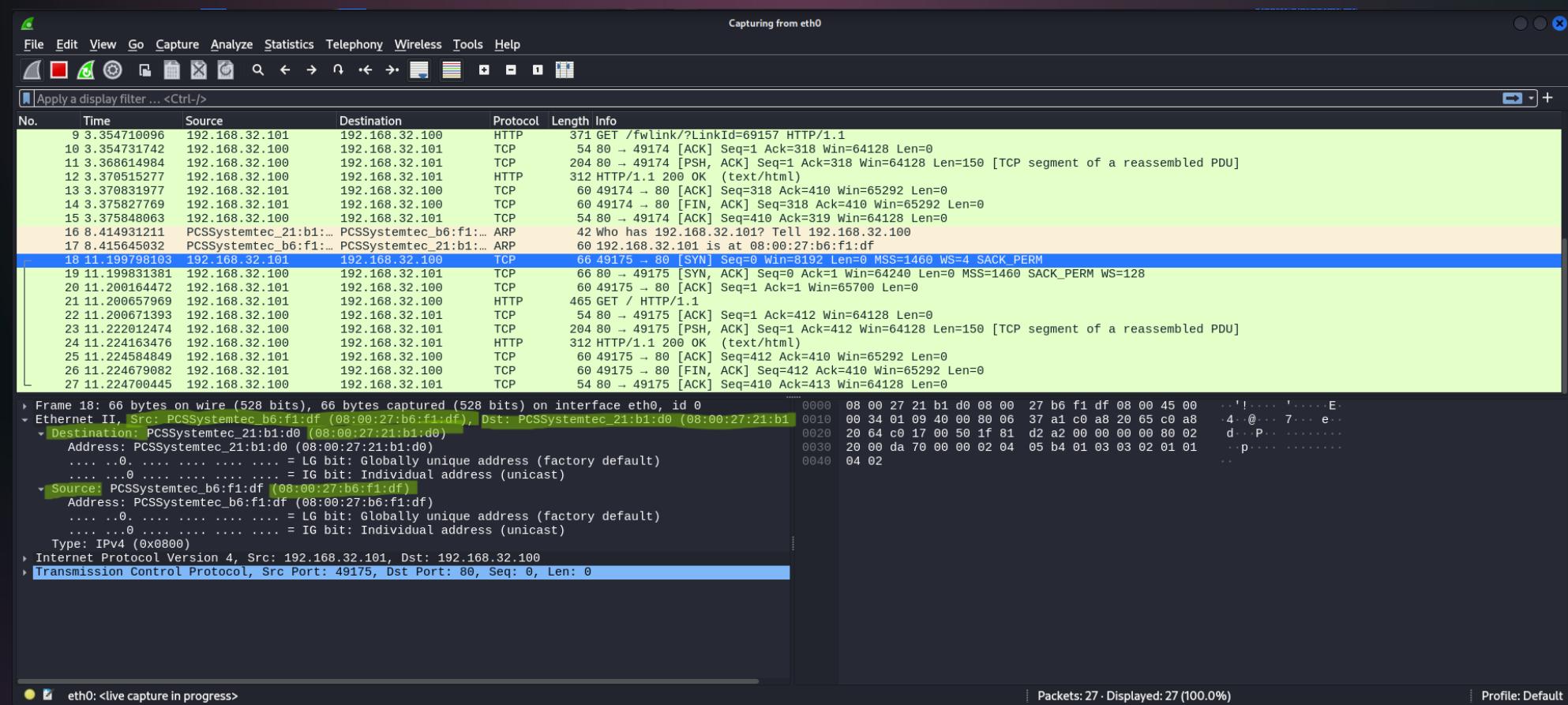
#####
# start_service
#
# The services to start
#   Home
# Syntax: start_service <service name>
#
# Default: none
#
# Available service names are:
# dns, http, smtp, pop3, tftp, ftp, ntp, time_tcp,
# time_udp, daytime_tcp, daytime_udp, echo_tcp,
# echo_udp, discard_tcp, discard_udp, quodt_tcp,
# quodt_udp, 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 ft�
#start_service tftp
#start_service irc
#start_service ntp
#start_service finger
#start_service ident
#start_service syslog
#start_service time_tcp
#start_service time_udp

^G Help      ^O Write Out    ^W Where Is      ^K Cut          ^T
^X Exit      ^R Read File     ^\ Replace      ^U Paste        ^J
```

Seconda configurazione Inetsim:

Il prossimo passo è cambiare configurazione al nostro Inetsim, impostando l'emulazione di un **protocollo HTTP**.

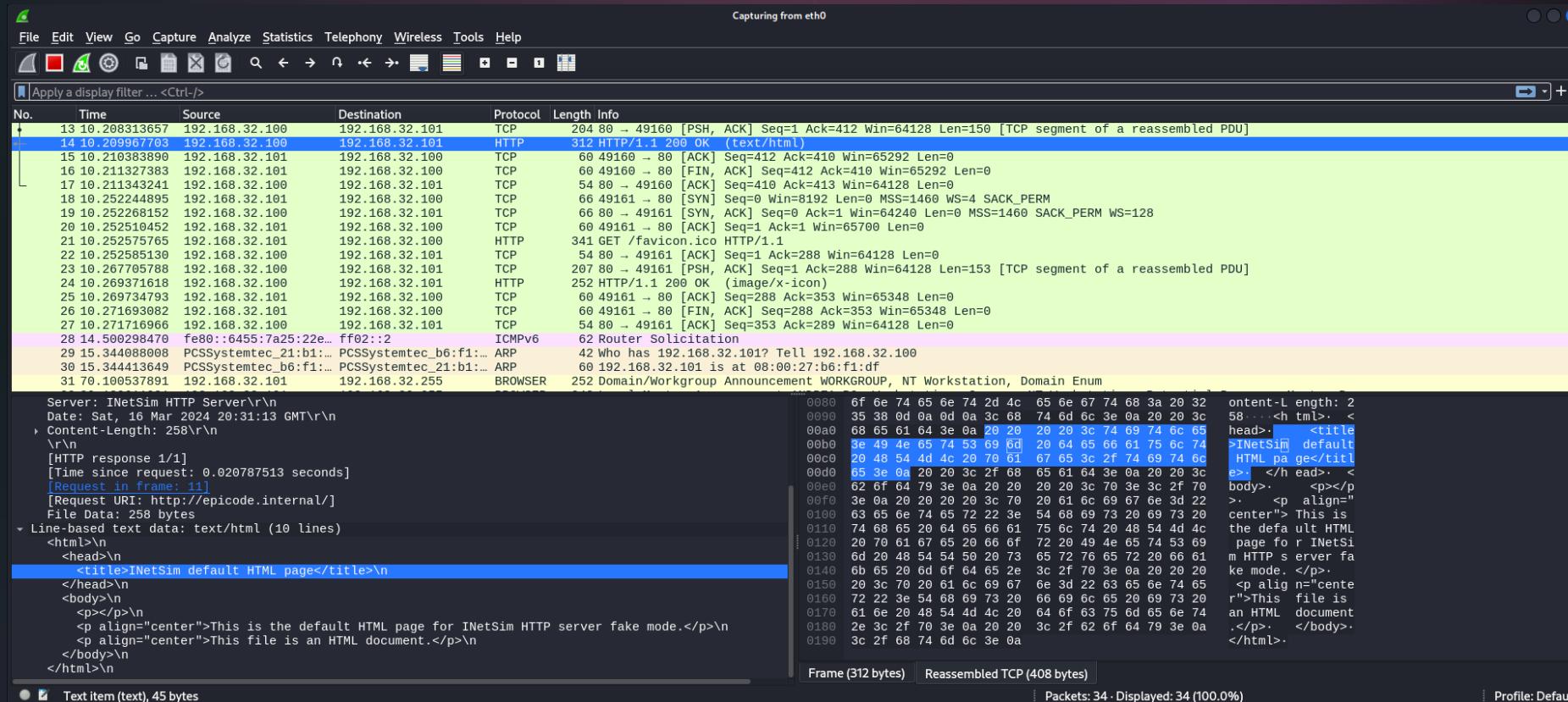
Passando sempre dal file *.conf* andremo a sostituire il protocollo **HTTPS** con **HTTP**, salveremo il file e lanceremo di nuovo **Inetsim** con **Wireshark**.



Pacchetti HTTP:

Come abbiamo già visto, **Wireshark** ci mostrerà i pacchetti catturati come nel caso precedente, ma questa volta mostrerà **pacchetti e protocolli diversi**.

Pacchetti HTTP:



Come conferma di **apertura e funzionamento** del sito fittizio possiamo vedere tra i protocolli, il nostro HTTP evidenziato (contenente il corpo del sito) che con il codice di stato «**200 OK**» conferma che la trasmissione è stata **ricevuta, compresa e processata correttamente** dal server in questione!

Considerazioni Finali:

Arrivati alla fine del laboratorio possiamo notare delle differenze nel ***modus operandi dei due protocolli***:

- La prima cosa che risalta all'occhio è la **mancanza di crittografie nel protocollo HTTP**, che trasmette tutti i **dati in chiaro**. Ciò significa che qualsiasi informazione inviata tra il tuo computer e il server di destinazione (come nel nostro caso) **può essere letta da chiunque intercetti la comunicazione**, quindi con il classico *MITMA* o *Man In The Middle Attack* e di conseguenza un **ARP Poisoning** (o Spoofing) si potrebbe arrivare al **furto di dati**.
- Nel **protocollo HTTPS** invece è presente il protocollo **TLSv1 (Transport Layer Security)**. Il **TLS** è un **protocollo di sicurezza** che lavora principalmente in **4 livelli OSI (Layer 7-6-5-4)**, **garantisce l'integrità dei dati** e **fornisce sicurezza e autenticazione tramite una crittografia a chiave asimmetrica**. Guardando la schermata di wireshark durante la scansione del sito **HTTPS**, possiamo notare infatti che quasi tutti i pacchetti del protocollo **TLSv1** sono **crittografati e quindi quasi impossibili da decifrare**.

In conclusione, si può accettare che utilizzare siti che impiegano il protocollo HTTPS sono MOLTO più sicuri dei siti in HTTP.

FINE.

