W3 D5 - 11 07 2025
- Giuseppe Gigliotti
- Relazione Policy & Packet Capture

Policy & Packet Capture

Vedremo due esercizi:

1) la configurazione di una policy sul firewall windows;

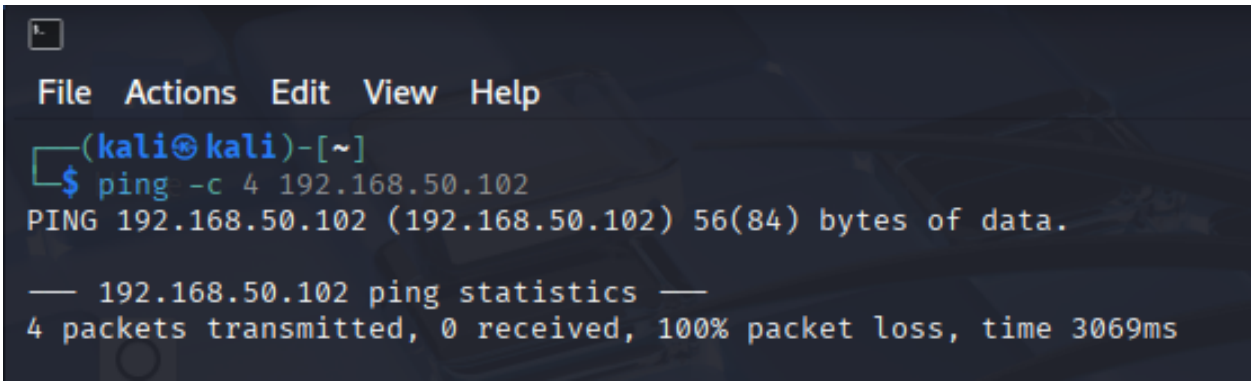
2) una packet capture con Wireshark.

Vedremo anche come simulare alcuni servizi di rete con un tool pre-installato su Kali Linux (InetSim)

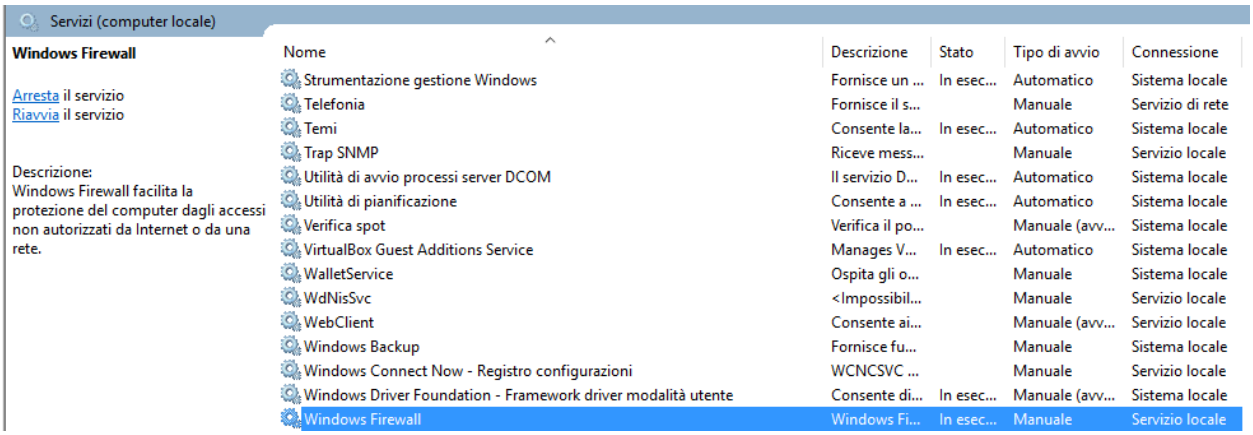
Esercizio:

● Configurare policy per permettere il ping da macchina Linux a macchina Windows nel nostro laboratorio (Windows firewall)

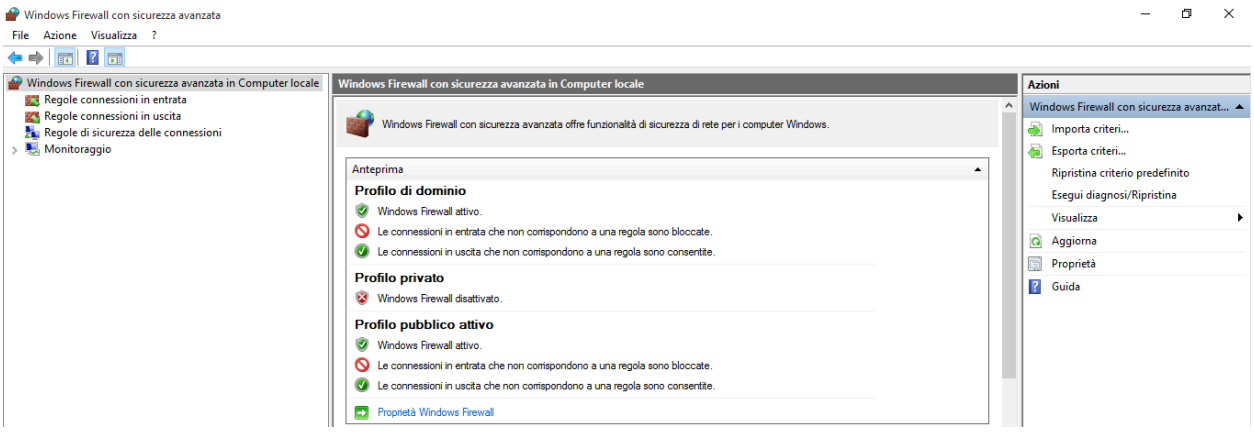
Ricordando dalle configurazioni delle macchine virtuali ricordiamo che non possiamo effettuare con la macchina kali il ping alla macchina windows con IP: 192.168.50.102. Per completezza verifichiamo facendo un ping



Per abilitare la comunicazione tra le due macchine ci spostiamo sulla macchina windows. Prima di tutto controlliamo se il firewall sia accesso o spento tramite la ricerca sui servizi di windows.

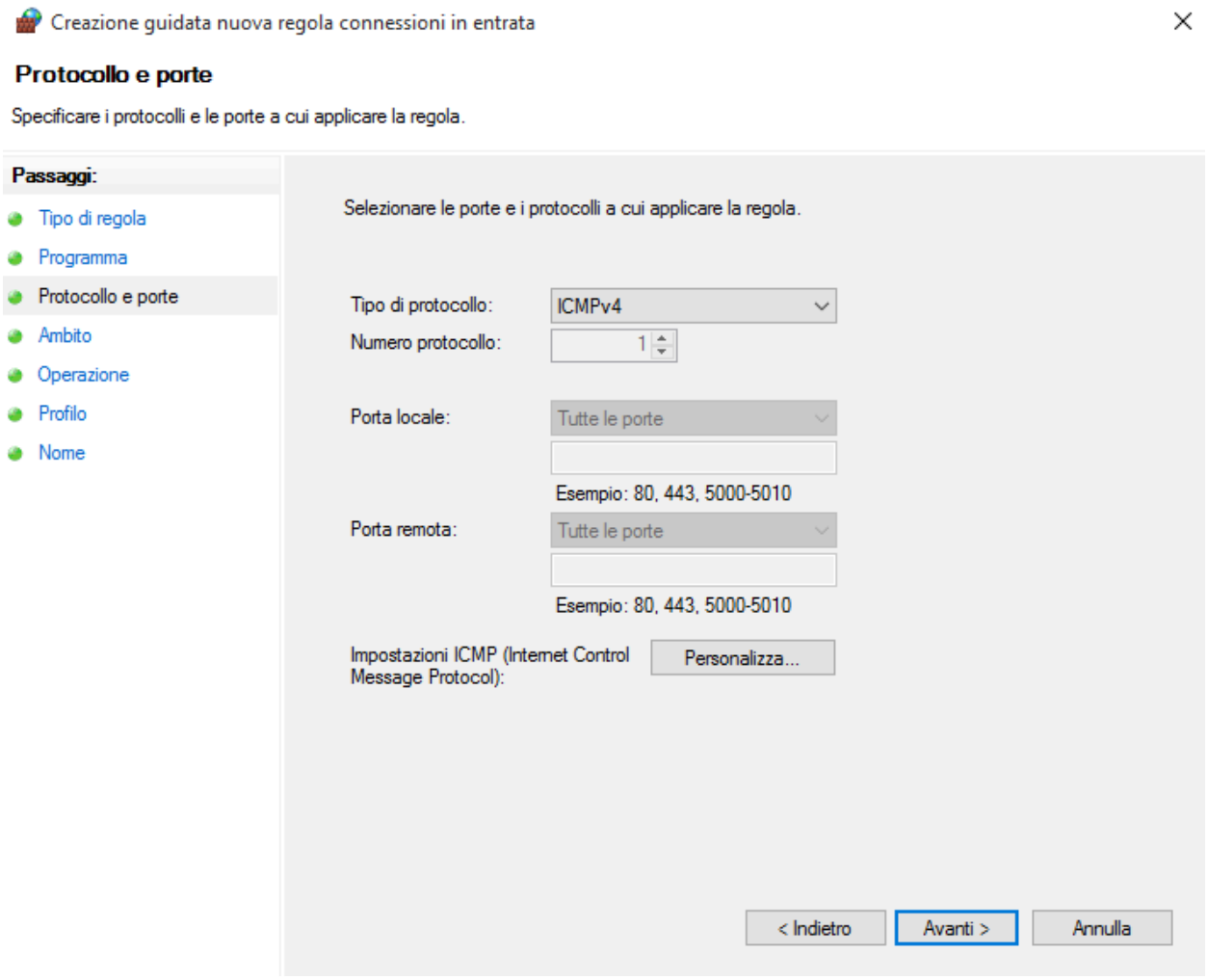


Dopodiché cerchiamo nel menù Windows firewall con sicurezza Avanzata

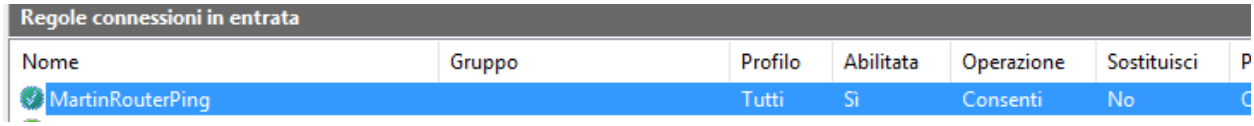


Nel nostro caso vogliamo effettuare una connessione in entrata tramite il comando ping, quindi andiamo su “regole connessione in entrata”. Per creare la nostra regola, andiamo sul lato destro e clicchiamo nuova regola

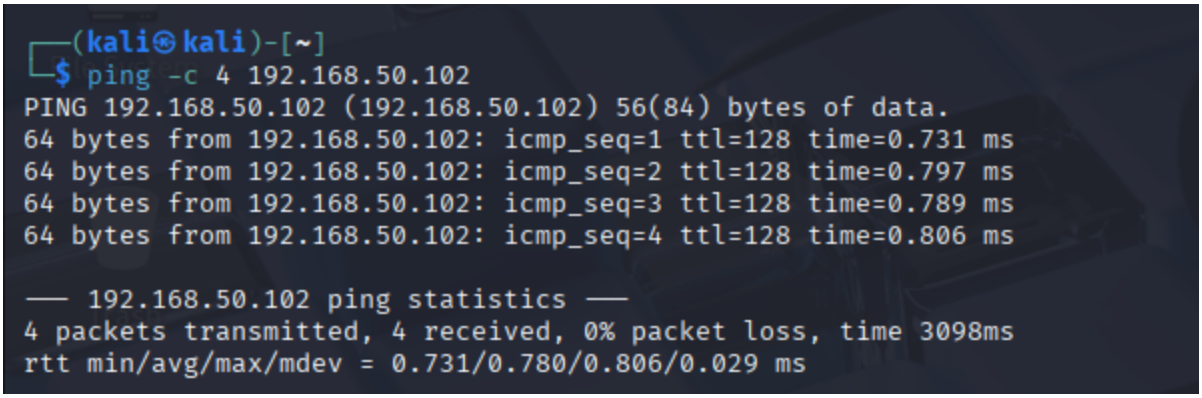
Non appena clicchiamo su nuova regola, selezioniamo il tipo di regola, nel nostro caso selezioniamo “personalizzato” , a quali programmi dobbiamo applicare la regola, e mettiamo la spunta su tutti i programmi. Arrivati a selezione porte e protocolli, per utilizzare il tool ping abbiamo bisogno del protocollo ICMP.



Alla sezione “selezionare gli indirizzi IP locali e/o remoti a cui applicare la regola, lasciamo la spunta per entrambi su qualsiasi indirizzo. Andando avanti arriviamo a selezionare l’azione desiderata, nel nostro caso “consenti connessione” . Infine arriviamo a scegliere il nome della nostra regola, per il nostro esercizio la regola per le connessioni in entrata sarà: MartinRouterPing



Non appena creata la nostra regola, riproviamo con il ping dalla macchina Kali



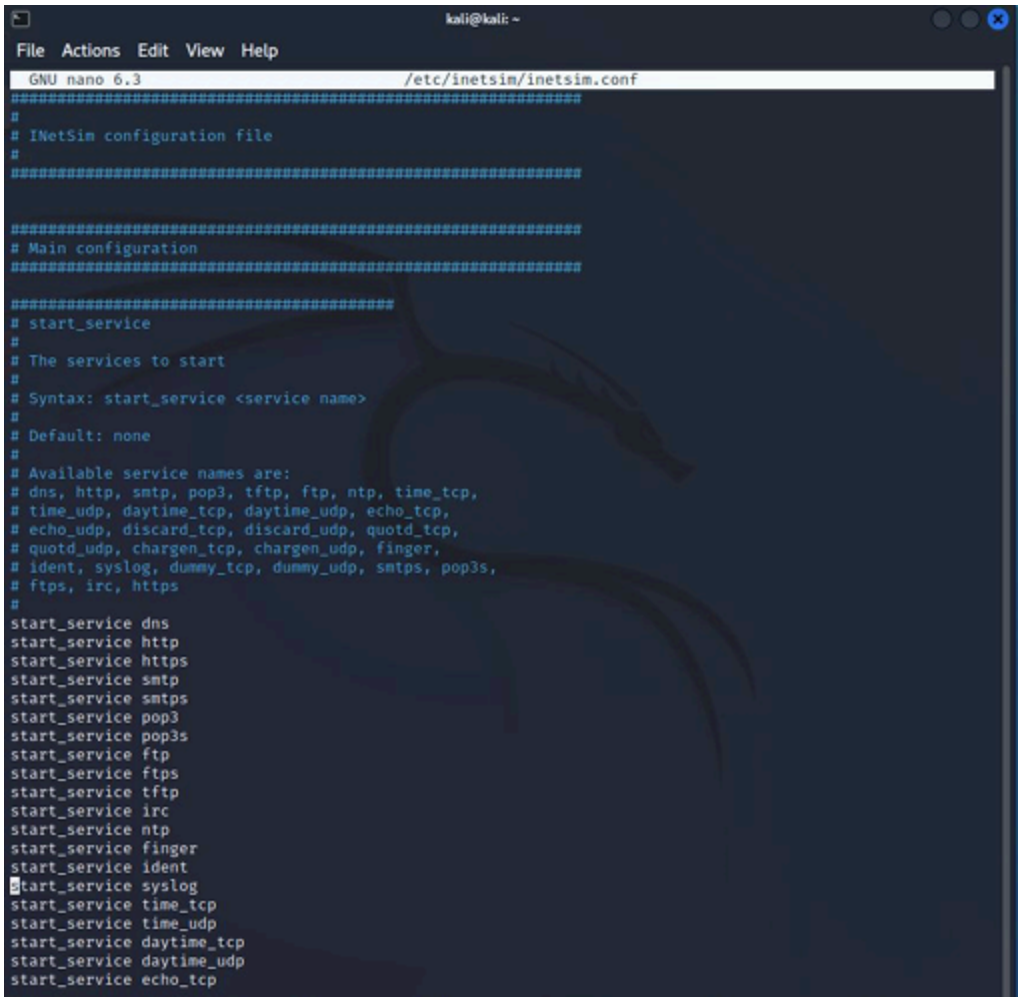
Notiamo che i pacchetti vengono inviati senza problemi dopo l'implementazione della regola di connessione in entrata.

- Utilizzo dell'utility InetSim per l'emulazione di servizi Internet

Utilizzeremo inetsim per l'esercizio 2. Questo è un simulatore di servizi internet, è pre-installato su Kali Linux e per avviarlo basta eseguire il comando "inetsim" dal terminale Kali con un account con privilegi di amministratore. Se eseguito senza customizzazioni, inetsim emula una quantità di servizi che va oltre le nostre necessità, quindi vedremo come configurare inetsim per avviare solo alcuni dei servizi necessari.

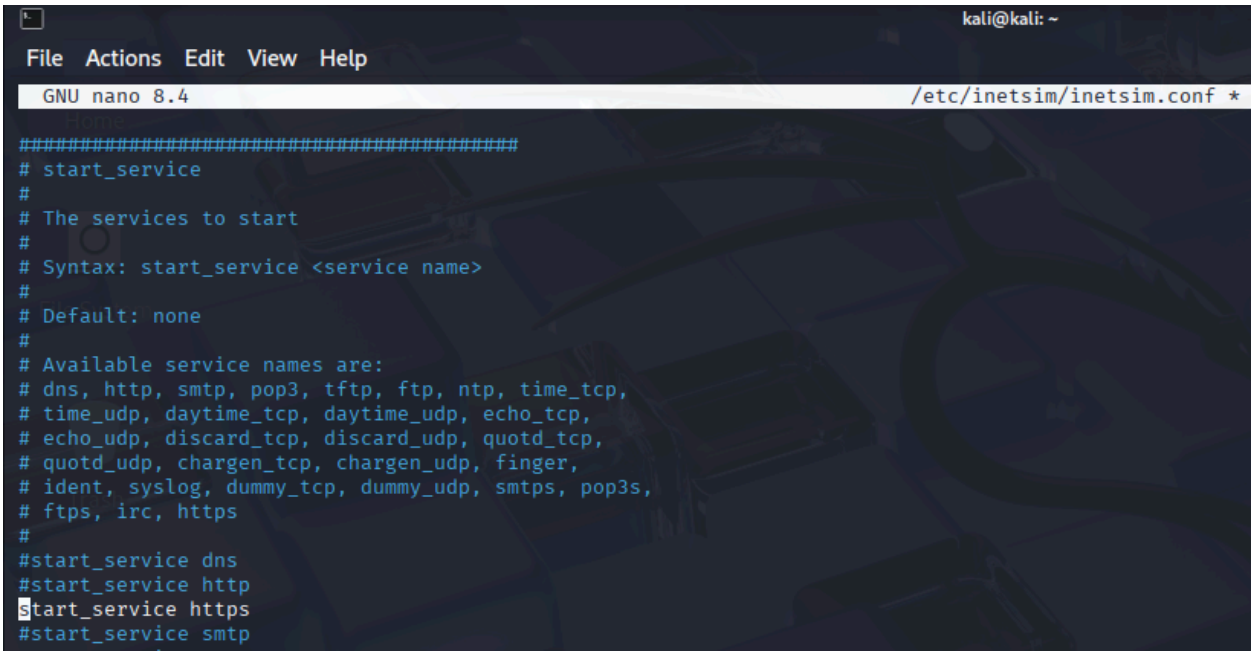
InetSim si può configurare modificando il contenuto del file inetsim.conf
al path /etc/inetsim, si possono scegliere quali servizi avviare e su che porta avviarli.

usando il comando: sudo nano /etc/inetsim/inetsim.conf , possiamo visualizzare il file di configurazione come nell'immagine sottostante

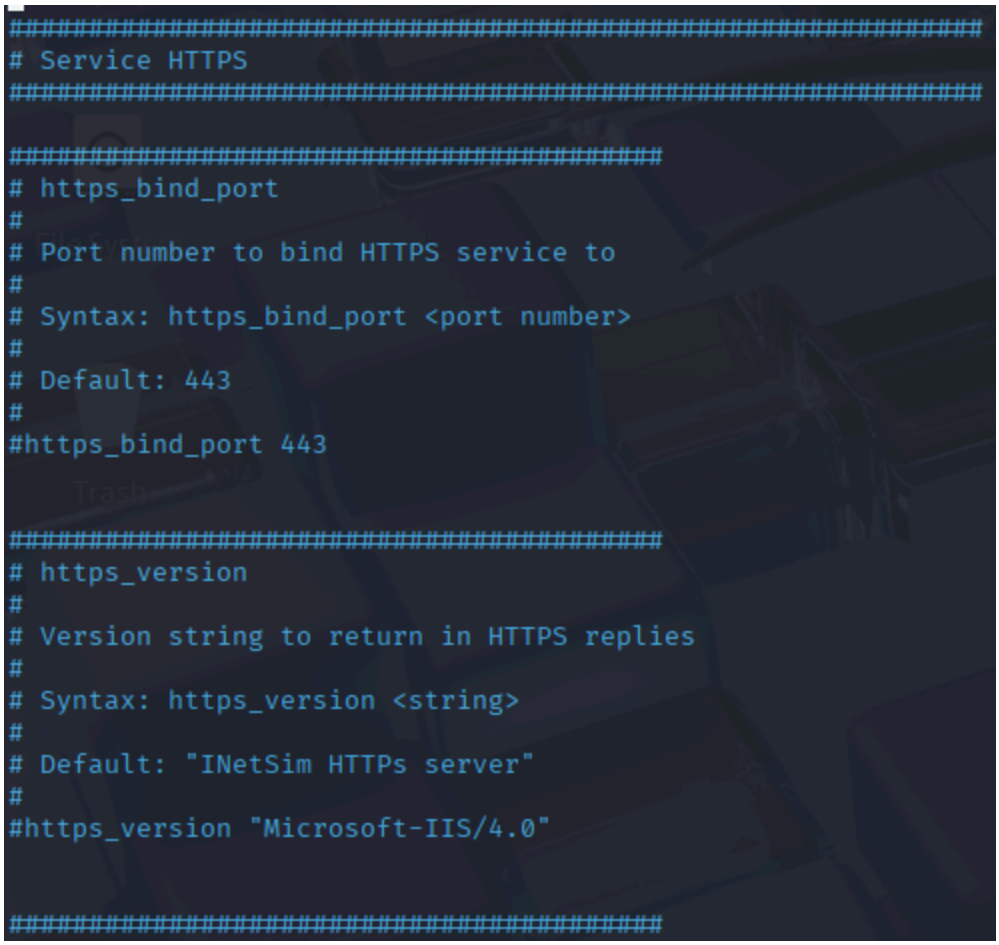


Per lo scopo di questa esercitazione, vogliamo attivare solamente il servizio HTTPS.

Quindi commentiamo tutto il resto, aggiungendo un carattere “#” prima di ogni riga. Le righe commentate, ovvero che iniziano con il carattere “#” non verranno eseguite e di conseguenza i relativi servizi non avviati.



Dopo aver commentato le righe dei servizi che non vogliamo usare e lasciato senza il commento il servizio https, possiamo scendere in basso nel file per visualizzare le impostazioni del servizio https



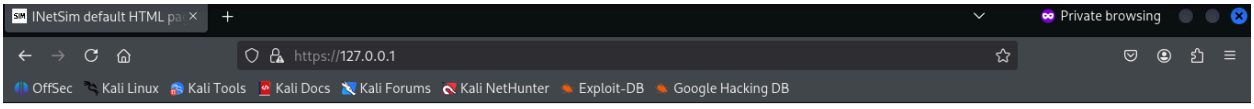
```
##################################################################
# Service HTTPS
##################################################################

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

########################################
# https_version
#
# Version string to return in HTTPS replies
#
# Syntax: https_version <string>
#
# Default: "INetSim HTTPs server"
#
#https_version "Microsoft-IIS/4.0"

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

Controllato il tutto, possiamo eseguire il comando: sudo inetsim da terminale

```
┌──(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 12744) ===
Session ID:     12744
Listening on:   127.0.0.1
Real Date/Time: 2025-07-23 01:08:40
Fake Date/Time: 2025-07-23 01:08:40 (Delta: 0 seconds)
 Forking services...
  * https_443_tcp - started (PID 12754)
 done.
Simulation running.
```

Notando che i servizi sono in ascolto su l'indirizzo di localhost (127.0.0.1), andiamo ad inserire lo stesso in un browser. Precisazione nella barra del browser inseriremo la url

https://127.0.0.1 che ci restituisce la pagina web sottostante, ciò vuol dire che il servizio è attivo ed è raggiungibile tramite l'indirizzo 127.0.0.1



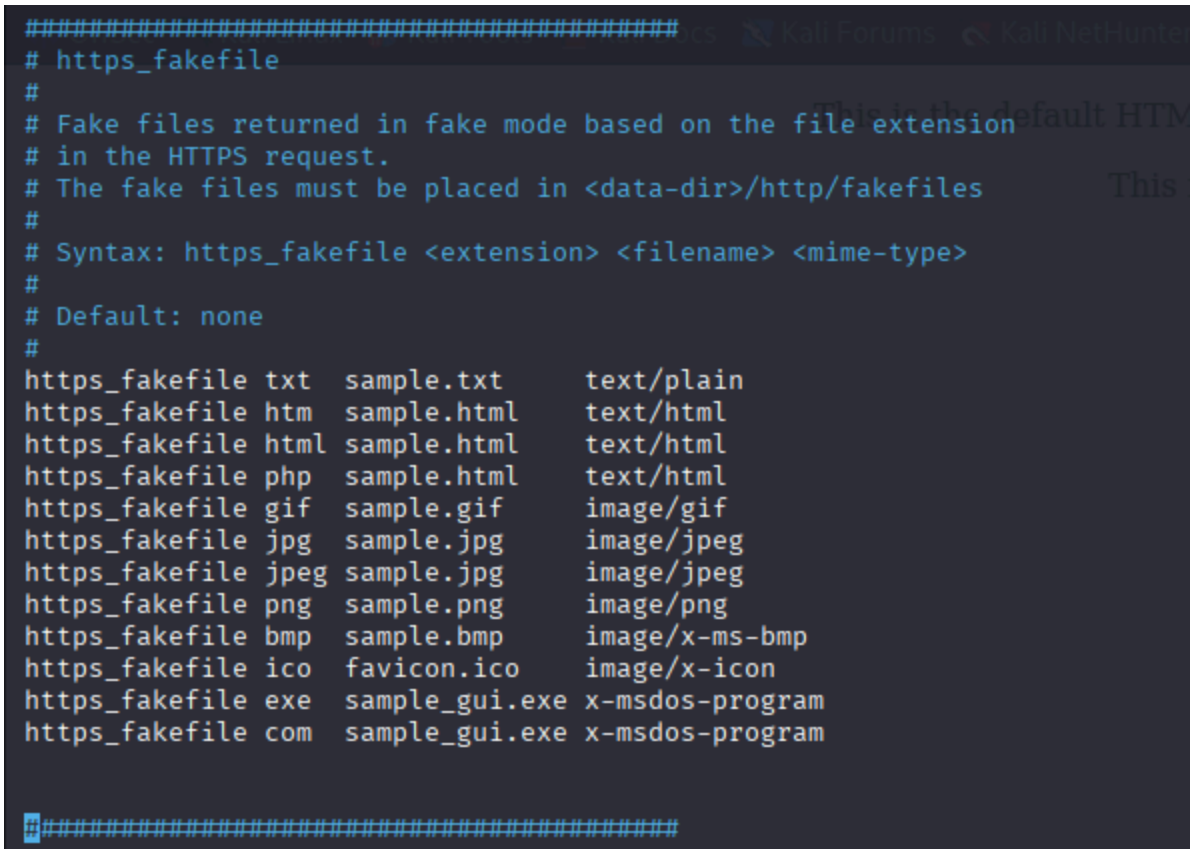
This is the default HTML page for INetSim HTTP server fake mode.

This file is an HTML document.

FACOLTATIVO:

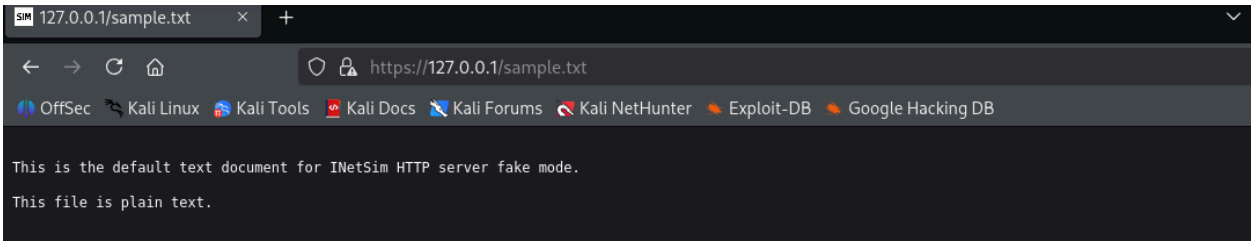
● Simulare altri servizi con InetSim

Come ulteriore test proviamo a richiedere uno dei file “fittizi” messi a disposizione da inetsim



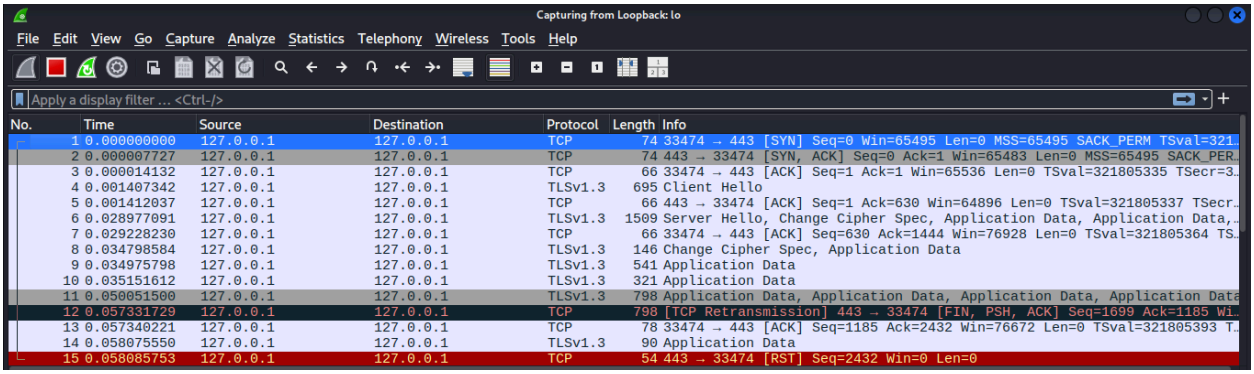
```
##########################################
# https_fakefile
#
# Fake files returned in fake mode based on the file extension
# in the HTTPS request.
# The fake files must be placed in <data-dir>/http/fakefiles
#
# Syntax: https_fakefile <extension> <filename> <mime-type>
#
# Default: none
#
https_fakefile txt  sample.txt     text/plain
https_fakefile htm  sample.html    text/html
https_fakefile html sample.html    text/html
https_fakefile php  sample.html    text/html
https_fakefile gif  sample.gif     image/gif
https_fakefile jpg  sample.jpg     image/jpeg
https_fakefile jpeg sample.jpg     image/jpeg
https_fakefile png  sample.png     image/png
https_fakefile bmp  sample.bmp     image/x-ms-bmp
https_fakefile ico  favicon.ico    image/x-icon
https_fakefile exe  sample_gui.exe x-msdos-program
https_fakefile com  sample_gui.exe x-msdos-program


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

Prendiamo per il nostro esempio il fake file sample.txt e tornando sul browser inseriamo la url https://127.0.0.1/sample.txt, dove ci restituirà

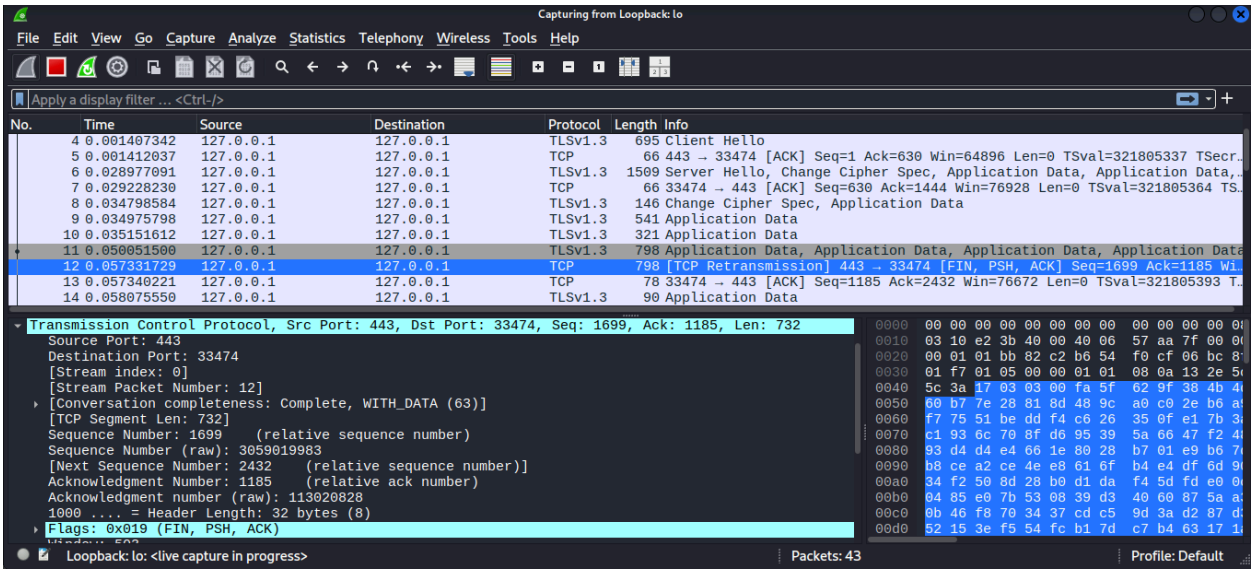


- Cattura di pacchetti con Wireshark
  - Procedere con lo sniffing delle comunicazioni

Per fare ciò utlizziamo wireshark. Per lo sniffing delle comunicazioni andremo in ascolto sull'indirizzo di lopback già utilizzato nei punti precedenti

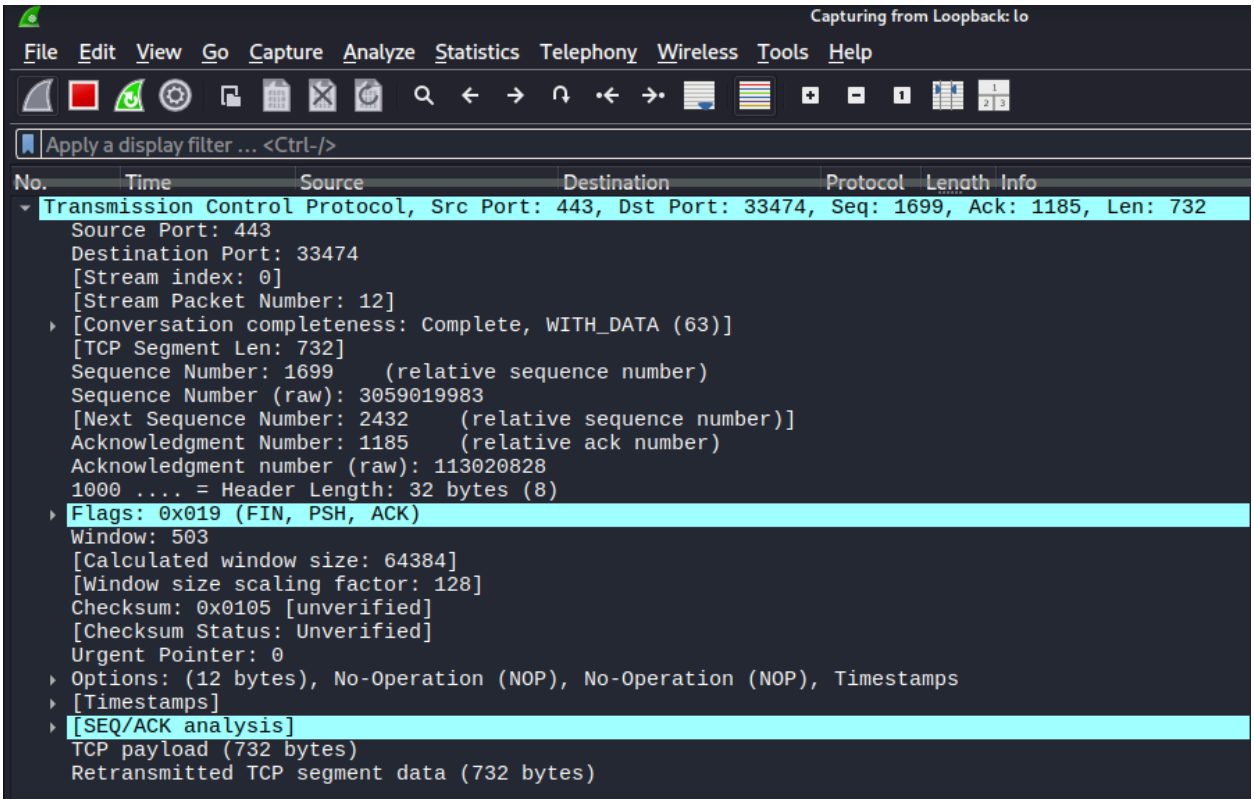


- Analizzare il contenuto dei pacchetti



Come pacchetto da analizzare prendiamo il pacchetto con il flag FIN, segno che la connessione TCP si è conclusa dopo un 3 way handshake

Vediamo meglio il pacchetto selezionato.


Capturing from Loopback: lo
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
Transmission Control Protocol, Src Port: 443, Dst Port: 33474, Seq: 1699, Ack: 1185, Len: 732
Source Port: 443
Destination Port: 33474
[Stream index: 0]
[Stream Packet Number: 12]
[Conversation completeness: Complete, WITH_DATA (63)]
[TCP Segment Len: 732]
Sequence Number: 1699 (relative sequence number)
Sequence Number (raw): 3059019983
[Next Sequence Number: 2432 (relative sequence number)]
Acknowledgment Number: 1185 (relative ack number)
Acknowledgment number (raw): 113020828
1000 .... = Header Length: 32 bytes (8)
Flags: 0x019 (FIN, PSH, ACK)
Window: 503
[Calculated window size: 64384]
[Window size scaling factor: 128]
Checksum: 0x0105 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
[Timestamps]
[SEQ/ACK analysis]
TCP payload (732 bytes)
Retransmitted TCP segment data (732 bytes)