

W3D4 - Pratica



W3D4 - Pratica PDF

Esercizio

L'esercizio di oggi mira a consolidare le conoscenze acquisite.

Vedremo due esercizi: I) la configurazione di una policy sul firewall windows; II) una packet capture con Wireshark.

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

Esercizio:

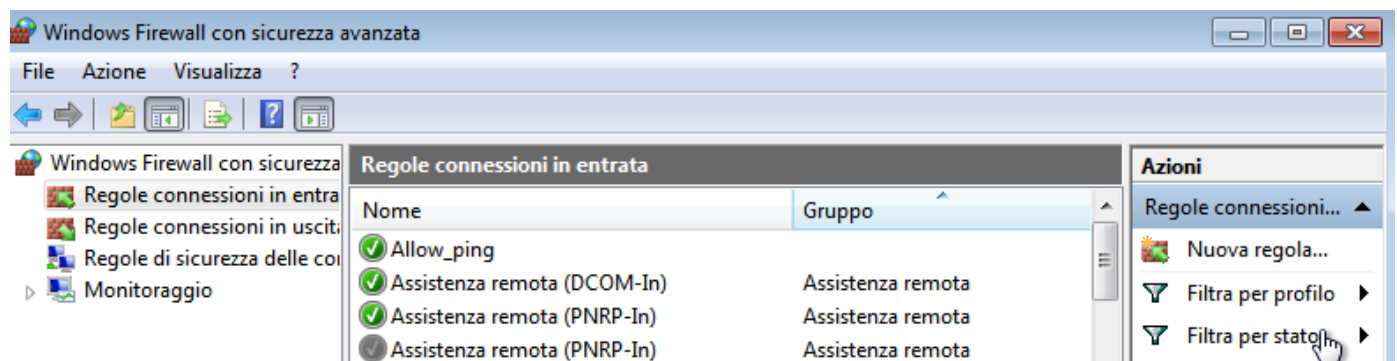
- ☐ Configurare policy per permettere il ping da macchine Linux a Macchina Windows 7 nel nostro laboratorio (Windows firewall)
- ☐ Utilizzo dell'utility InetSim per l'emulazione di servizi Internet
- ☐ Cattura di pacchetti con Wireshark

Per permettere il ping da Linux a Win7 devo iniziare a fare un settaggio del Firewall di Win7.

Dal menu iniziale cerco Windows Firewall

- Impostazioni avanzate
- Regole connessioni in entrata
- Azione
- Nuova regola (scelgo una regola che permetterà il ping con kali)
- Personalizzata
- Tutti i programmi
- Tipo di protocollo ICMPv4
- Qualsiasi indirizzo IP
- Consenti la connessione
- "Allow_ping"

A questo punto il **ping tra kali e win7 verrà consentito**



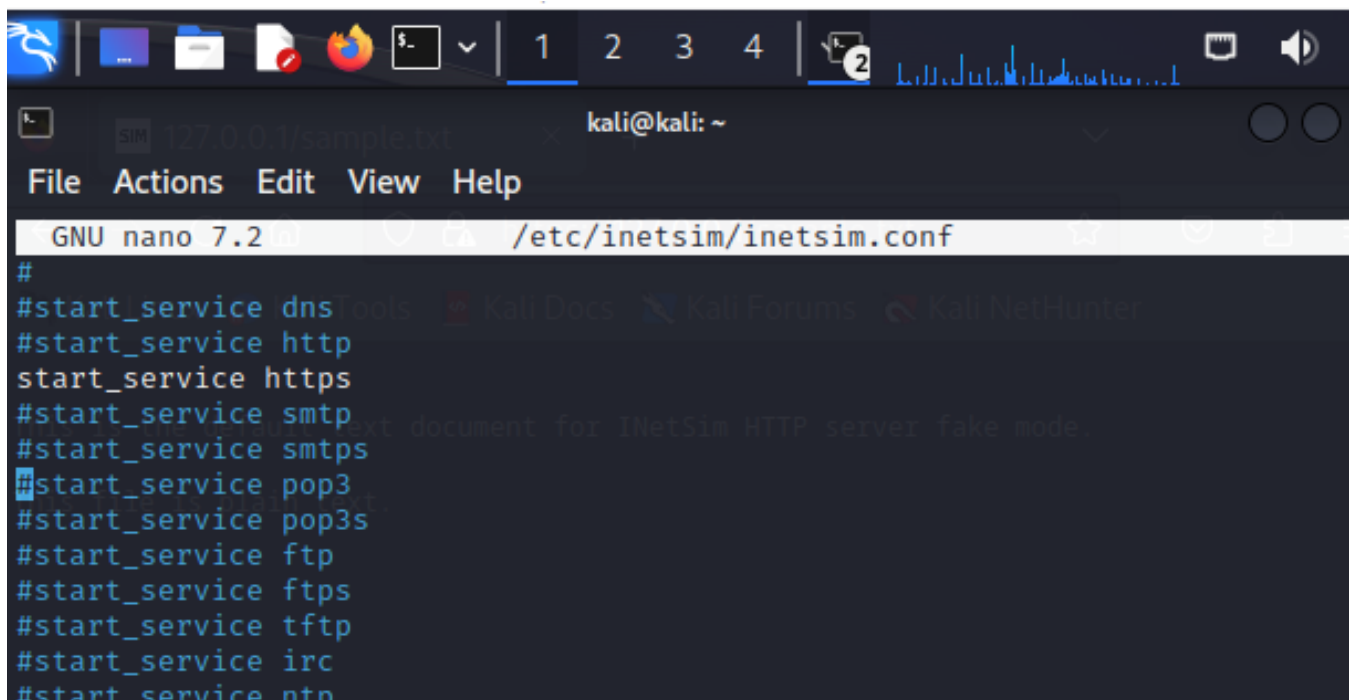
Ora utilizzerò un altro tool preinstallato su Kali, cioè **InetSim** che serve a simulare alcuni servizi internet per esempio HTTP e HTTPS. Verrà così emulata una connessione come fosse reale e la si vedrà nel dettaglio proprio per vedere la differenza tra HTTP e HTTPS.

Passo ora alla configurazione di InetSim tramite il file `inetsim.conf` al path

`/etc/inetsim`

Si possono scegliere quali servizi avviare e su quale porta avviarli.

Avvio Kali ed eseguo il comando `sudo nano /etc/inetsim/inetsim.conf`



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

Ora attivo solo il servizio HTTPS quindi aggiungo un `#` prima di ogni riga lasciando scoperto appunto HTTPS così da analizzare il traffico dell'HTTPS

Nella sezione successiva possiamo decidere l'IP dove avviare il servizio; quindi, il `service_bind_address` e di default viene proposto 127.0.0.1, ma lo modifichiamo con il numero IP 192.168.50.100 (kali) perché il servizio deve avvenire tra Kali e Win7

```
File Actions Edit View Help
GNU nano 7.2 /etc/inetsim/inetsim.conf
#start_service chargen_udp
#start_service dummy_tcp
#start_service dummy_udp

this is the default text document for INetSim HTTP server fake mode.
#####
# service_bind_address
#
# IP address to bind services to
#
# Syntax: service_bind_address <IP address>
#
# Default: 127.0.0.1
#
#service_bind_address 10.10.10.1

#####
# service_run_as_user
#
# User to run services
```

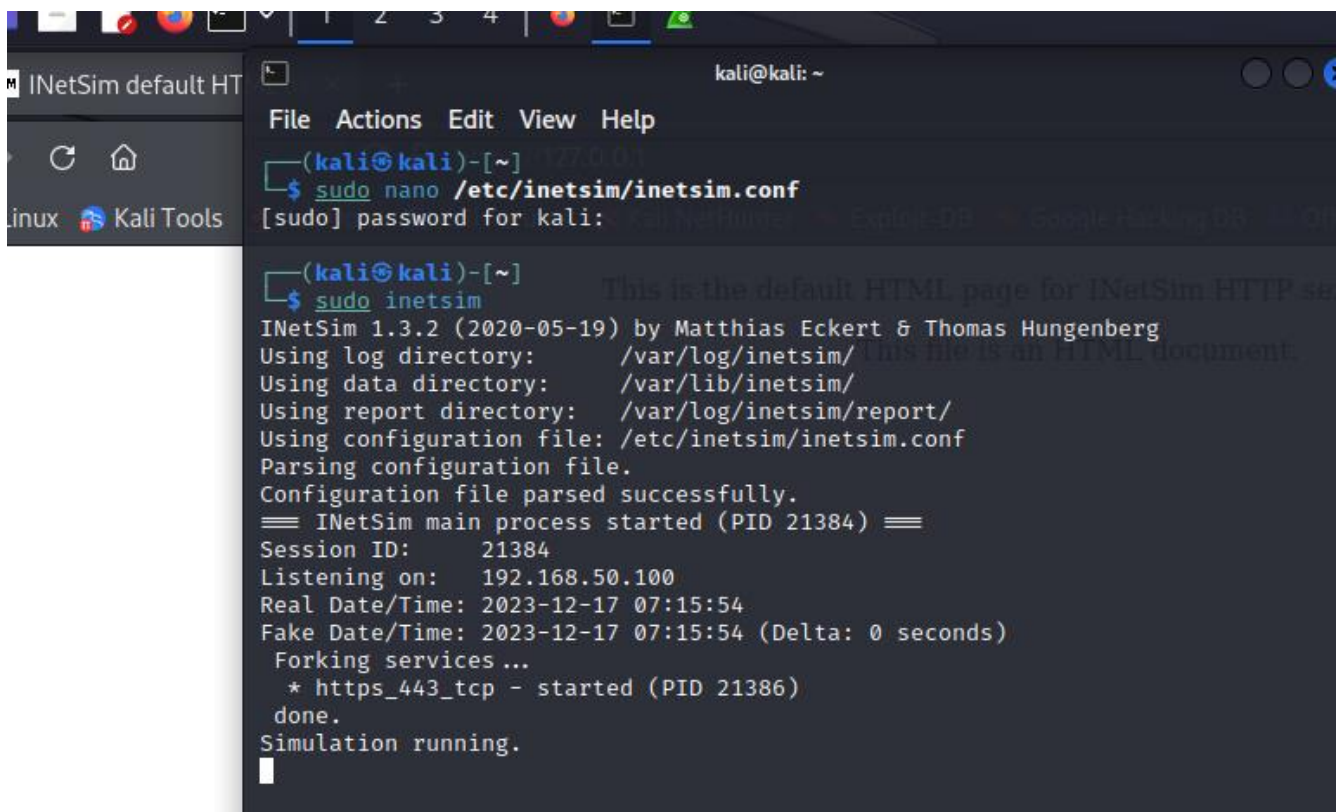
```
GNU nano 7.2 /etc/inetsim/inetsim.conf
#start_service dummy_udp

#####
# service_bind_address
#
# IP address to bind services to
#
# Syntax: service_bind_address <IP address>
#
# Default: 127.0.0.1
#
#service_bind_address 192.168.50.100

#####
# service_run_as_user
#
# User to run services
```

Ora, per simulare uno scenario reale, InetSim ci mette a disposizione dei **FAKE files** ovvero dei file vuoti con un'estensione che possiamo richiedere come se fossero delle risorse reali

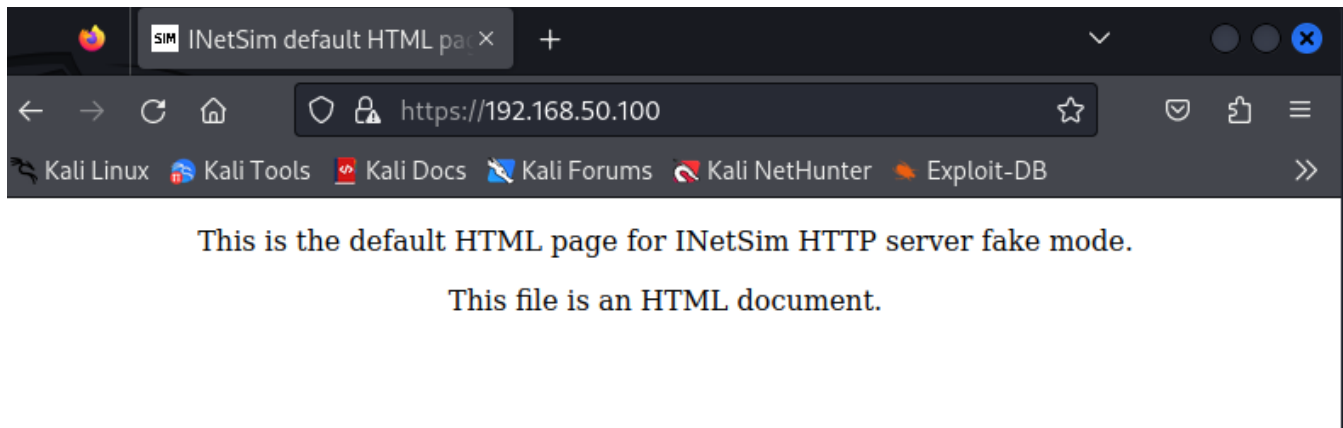
Scriviamo `sudo inetsim`



```
(kali@kali)-[~]
$ sudo nano /etc/inetsim/inetsim.conf
[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 21384) ==
Session ID: 21384
Listening on: 192.168.50.100
Real Date/Time: 2023-12-17 07:15:54
Fake Date/Time: 2023-12-17 07:15:54 (Delta: 0 seconds)
Forking services ...
* https_443_tcp - started (PID 21386)
done.
Simulation running.
```

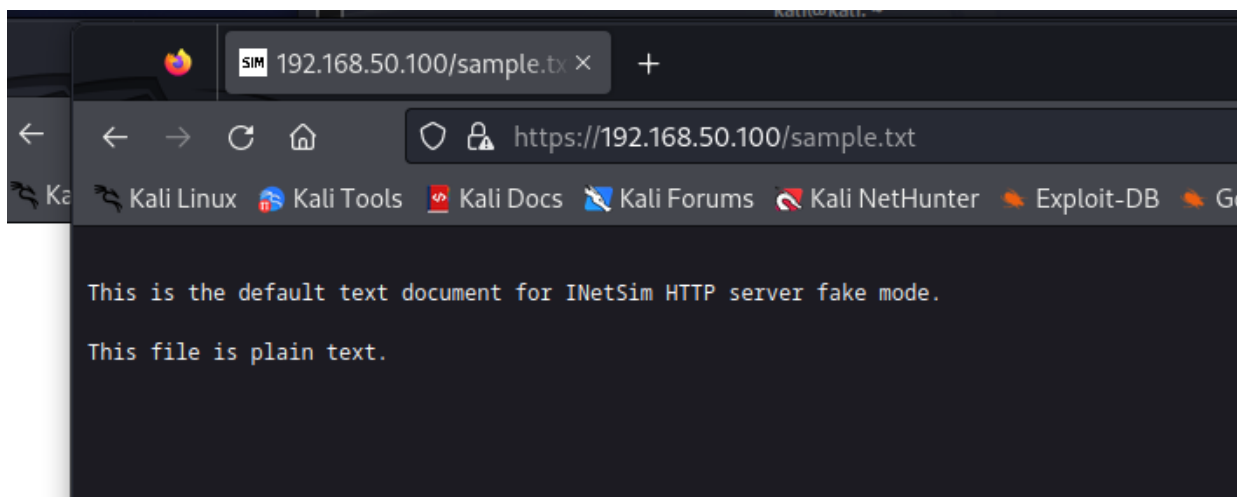
Il servizio HTTPS deve essere in ascolto sulla porta 443 del Local Host; quindi, proviamo a connetterci alla porta 443 del local host da un web browser su Kali

Andiamo su browser di kali e digitiamo [HTTP://192.168.50.100](http://192.168.50.100) e accettiamo il **WARNING di sicurezza**

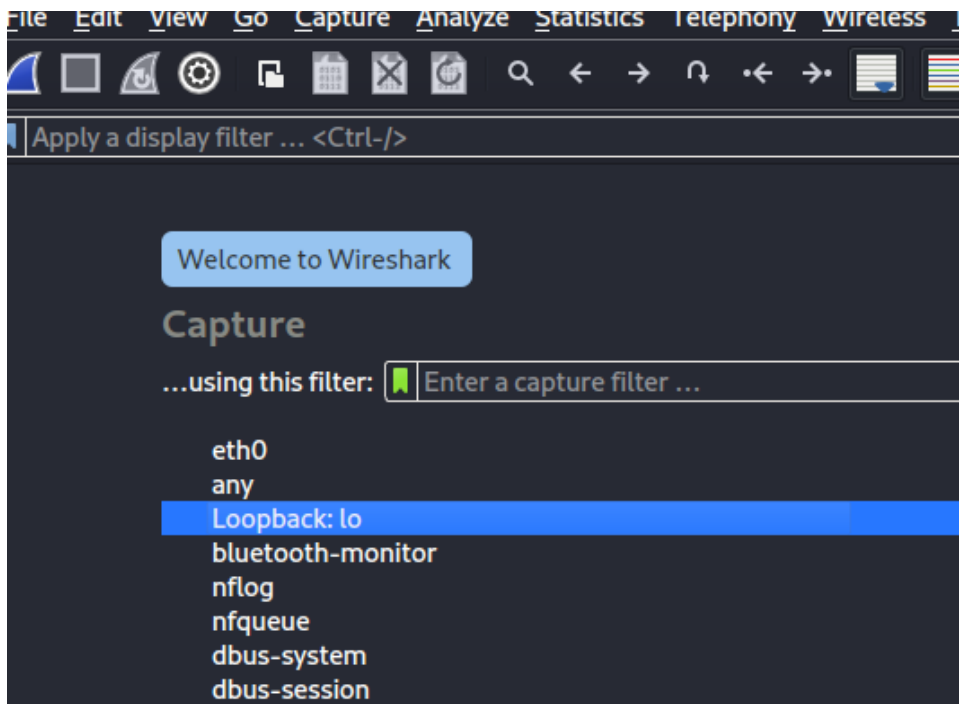


Dopodiché richiediamo

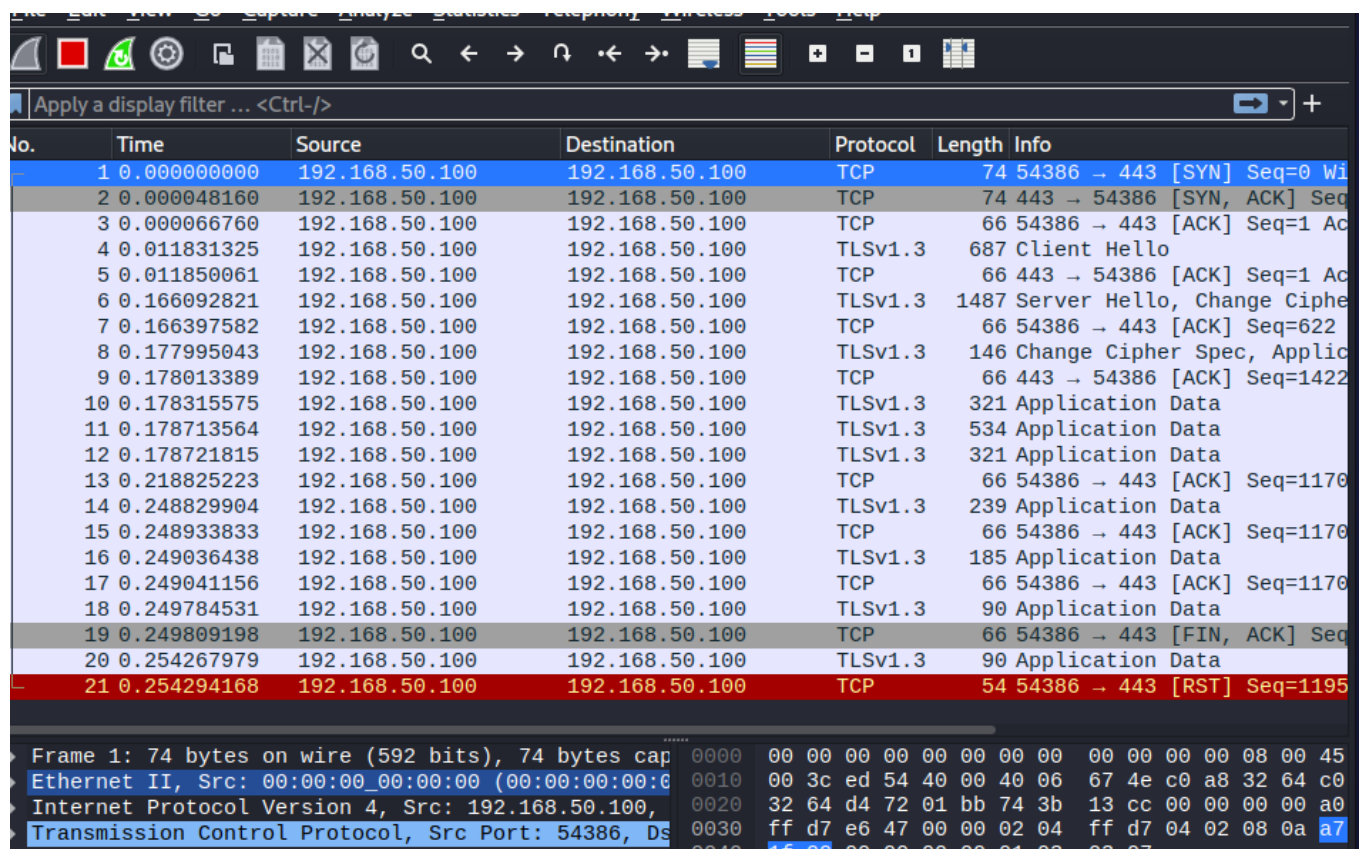
Uno dei file fittizi messi a disposizione da Inetsim andando sulla barra di ricerca del browser e digitando [HTTPS://192.168.50.100/sample.txt](https://192.168.50.100/sample.txt)



A questo punto avviamo **WIRESHARK** che si metterà in ascolto sull'interfaccia di **LOOPBACK** e ci connettiamo al local Host facendo una richiesta file.txt. Si analizzerà quindi la comunicazione tra le parti e noteremo come viene instaurata la sessione TCP tra le parti (la sequenza **SYN, SYN ACK, ACK**)



Clicco REFRESH sul browser e ottengo



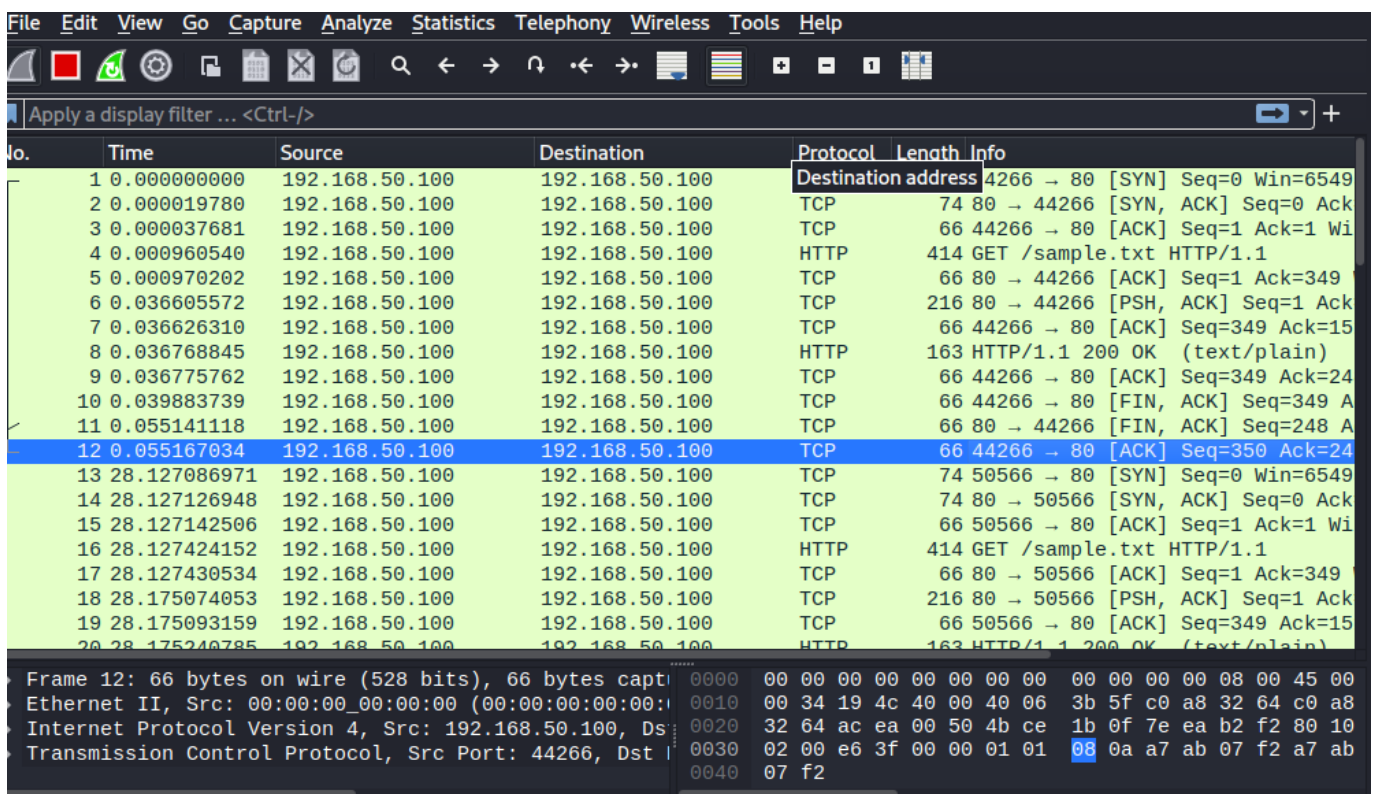
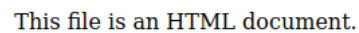
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.50.100	192.168.50.100	TCP	74	54386 → 443 [SYN] Seq=0 Win=0
2	0.000048160	192.168.50.100	192.168.50.100	TCP	74	443 → 54386 [SYN, ACK] Seq=1170
3	0.000066760	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [ACK] Seq=1170
4	0.011831325	192.168.50.100	192.168.50.100	TLSv1.3	687	Client Hello
5	0.011850061	192.168.50.100	192.168.50.100	TCP	66	443 → 54386 [ACK] Seq=1170
6	0.166092821	192.168.50.100	192.168.50.100	TLSv1.3	1487	Server Hello, Change Cipher Spec, Application Data
7	0.166397582	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [ACK] Seq=622
8	0.177995043	192.168.50.100	192.168.50.100	TLSv1.3	146	Change Cipher Spec, Application Data
9	0.178013389	192.168.50.100	192.168.50.100	TCP	66	443 → 54386 [ACK] Seq=1422
10	0.178315575	192.168.50.100	192.168.50.100	TLSv1.3	321	Application Data
11	0.178713564	192.168.50.100	192.168.50.100	TLSv1.3	534	Application Data
12	0.178721815	192.168.50.100	192.168.50.100	TLSv1.3	321	Application Data
13	0.218825223	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [ACK] Seq=1170
14	0.248829904	192.168.50.100	192.168.50.100	TLSv1.3	239	Application Data
15	0.248933833	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [ACK] Seq=1170
16	0.249036438	192.168.50.100	192.168.50.100	TLSv1.3	185	Application Data
17	0.249041156	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [ACK] Seq=1170
18	0.249784531	192.168.50.100	192.168.50.100	TLSv1.3	90	Application Data
19	0.249809198	192.168.50.100	192.168.50.100	TCP	66	54386 → 443 [FIN, ACK] Seq=1195
20	0.254267979	192.168.50.100	192.168.50.100	TLSv1.3	90	Application Data
21	0.254294168	192.168.50.100	192.168.50.100	TCP	54	54386 → 443 [RST] Seq=1195

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eth0
 Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
 Internet Protocol Version 4, Src: 192.168.50.100, Dst: 192.168.50.100
 Transmission Control Protocol, Src Port: 54386, Dst Port: 443

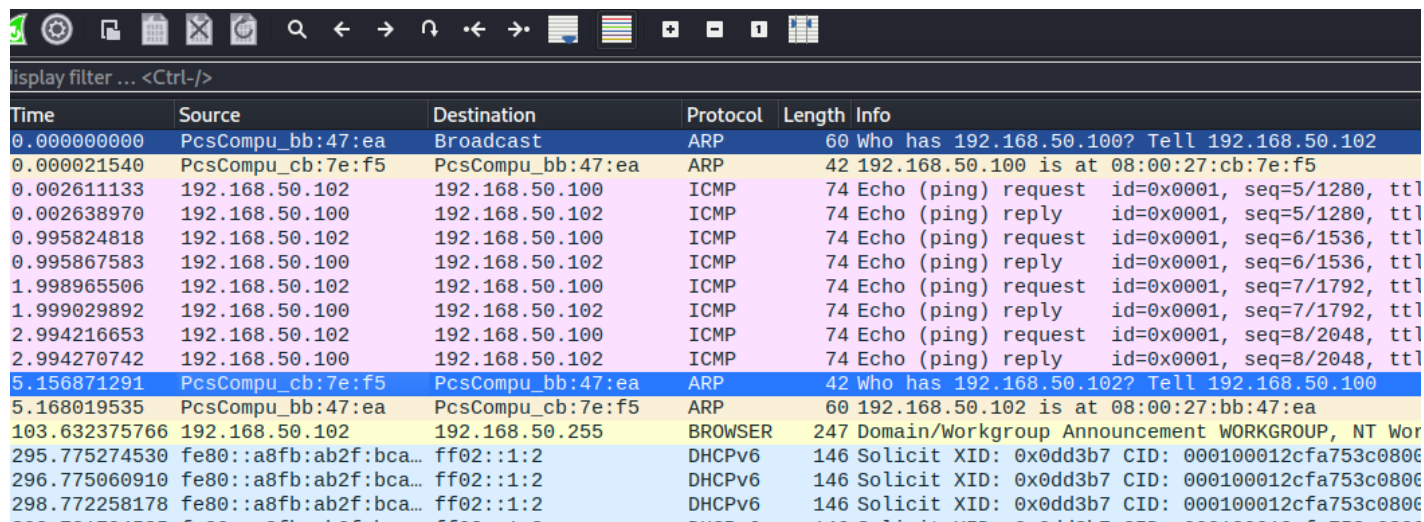
Le prime 3 sequenze [SYN], [SYN ACK], [ACK] sono quelle di creazione della comunicazione.

Ora effettuiamo lo stesso procedimento ma attraverso il servizio **HTTP** e la **porta 80**

```
File Actions Edit View Help
(kali@kali)-[~]
$ sudo nano /etc/inetsim/inetsim.conf
[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 22780) ==
Session ID: 22780
Listening on: 192.168.50.100
Real Date/Time: 2023-12-17 09:06:46
Fake Date/Time: 2023-12-17 09:06:46 (Delta: 0 seconds)
Forking services ...
* http_80_tcp - started (PID 22790)
done.
Simulation running.
```

Ora ho aperto il command su WIN7 e digitato ping 192.168.50.100 (kali) e aperto successivamente Wireshark su ETH0. È partita la richiesta ARP, le macchine pingano.



Wireshark packet capture interface showing traffic on the network. The display filter is set to 'eth 0'. The packet list shows the following entries:

Time	Source	Destination	Protocol	Length	Info
0.000000000	PcsCompu_bb:47:ea	Broadcast	ARP	60	Who has 192.168.50.100? Tell 192.168.50.102
0.000021540	PcsCompu_cb:7e:f5	PcsCompu_bb:47:ea	ARP	42	192.168.50.100 is at 08:00:27:cb:7e:f5
0.002611133	192.168.50.102	192.168.50.100	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=64
0.002638970	192.168.50.100	192.168.50.102	ICMP	74	Echo (ping) reply id=0x0001, seq=5/1280, ttl=64
0.995824818	192.168.50.102	192.168.50.100	ICMP	74	Echo (ping) request id=0x0001, seq=6/1536, ttl=64
0.995867583	192.168.50.100	192.168.50.102	ICMP	74	Echo (ping) reply id=0x0001, seq=6/1536, ttl=64
1.998965506	192.168.50.102	192.168.50.100	ICMP	74	Echo (ping) request id=0x0001, seq=7/1792, ttl=64
1.999029892	192.168.50.100	192.168.50.102	ICMP	74	Echo (ping) reply id=0x0001, seq=7/1792, ttl=64
2.994216653	192.168.50.102	192.168.50.100	ICMP	74	Echo (ping) request id=0x0001, seq=8/2048, ttl=64
2.994270742	192.168.50.100	192.168.50.102	ICMP	74	Echo (ping) reply id=0x0001, seq=8/2048, ttl=64
5.156871291	PcsCompu_cb:7e:f5	PcsCompu_bb:47:ea	ARP	42	Who has 192.168.50.102? Tell 192.168.50.100
5.168019535	PcsCompu_bb:47:ea	PcsCompu_cb:7e:f5	ARP	60	192.168.50.102 is at 08:00:27:bb:47:ea
103.632375766	192.168.50.102	192.168.50.255	BROWSER	247	Domain/Workgroup Announcement WORKGROUP, NT Workstation
295.775274530	fe80::a8fb:ab2f:bca...	ff02::1:2	DHCPv6	146	Solicit XID: 0x0dd3b7 CID: 000100012cfa753c0806
296.775060910	fe80::a8fb:ab2f:bca...	ff02::1:2	DHCPv6	146	Solicit XID: 0x0dd3b7 CID: 000100012cfa753c0806
298.772258178	fe80::a8fb:ab2f:bca...	ff02::1:2	DHCPv6	146	Solicit XID: 0x0dd3b7 CID: 000100012cfa753c0806
