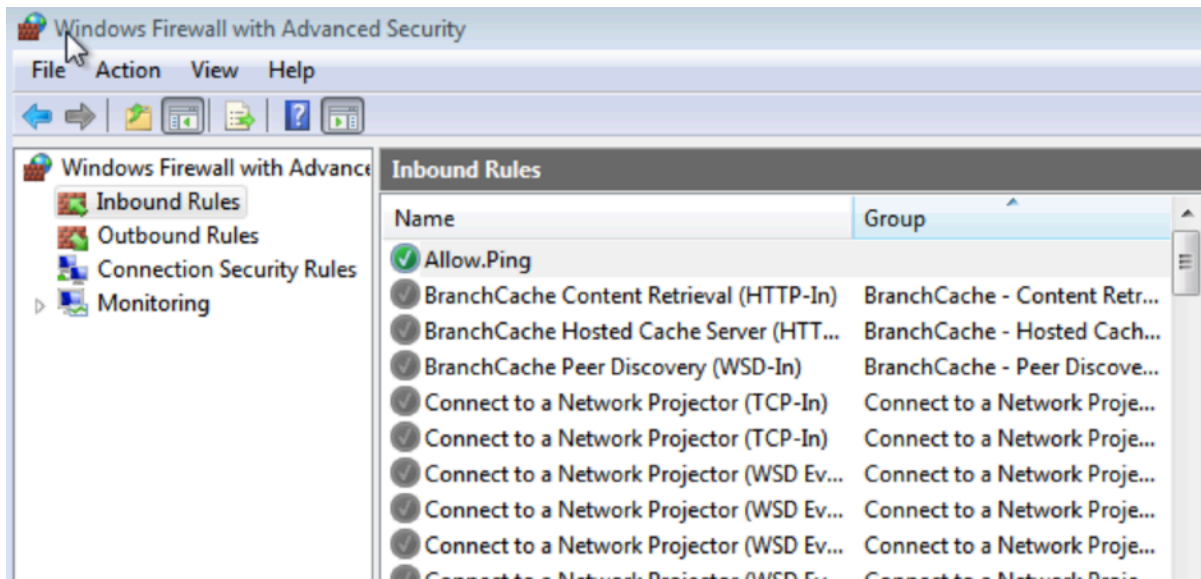


Configurare policy (Windows Firewall) per permettere il ping da Linux a Windows 7.



Configurata la policy “Allow.Ping” controllo se Windows7 (192.168.50.102) riceverà correttamente i pacchetti inviati da Kali (192.168.50.100)

```
salvatoremorale@kali: ~  
File Azioni Modifica Visualizza Aiuto  
zsh: corrupt history file /home/salvatoremorale/.zsh_history  
  
(salvatoremorale@kali)-[~]  
$ ifconfig eth0  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.50.100 netmask 255.255.255.0 broadcast 192.168.50.255  
    inet6 fe80::6c04:b7ff:fee8:7f17 prefixlen 64 scopeid 0x20<link>  
    ether 6e:04:b7:e8:7f:17 txqueuelen 1000 (Ethernet)  
    RX packets 68 bytes 3580 (3.4 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 17 bytes 2528 (2.4 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
(salvatoremorale@kali)-[~]  
$ ping 192.168.50.102  
PING 192.168.50.102 (192.168.50.102) 56(84) bytes of data.  
64 bytes from 192.168.50.102: icmp_seq=1 ttl=128 time=6.54 ms  
64 bytes from 192.168.50.102: icmp_seq=2 ttl=128 time=3.62 ms  
64 bytes from 192.168.50.102: icmp_seq=3 ttl=128 time=8.37 ms  
^C  
— 192.168.50.102 ping statistics —  
3 packets transmitted, 3 received, 0% packet loss, time 2013ms  
rtt min/avg/max/mdev = 3.621/6.173/8.365/1.953 ms  
  
(salvatoremorale@kali)-[~]  
$
```

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

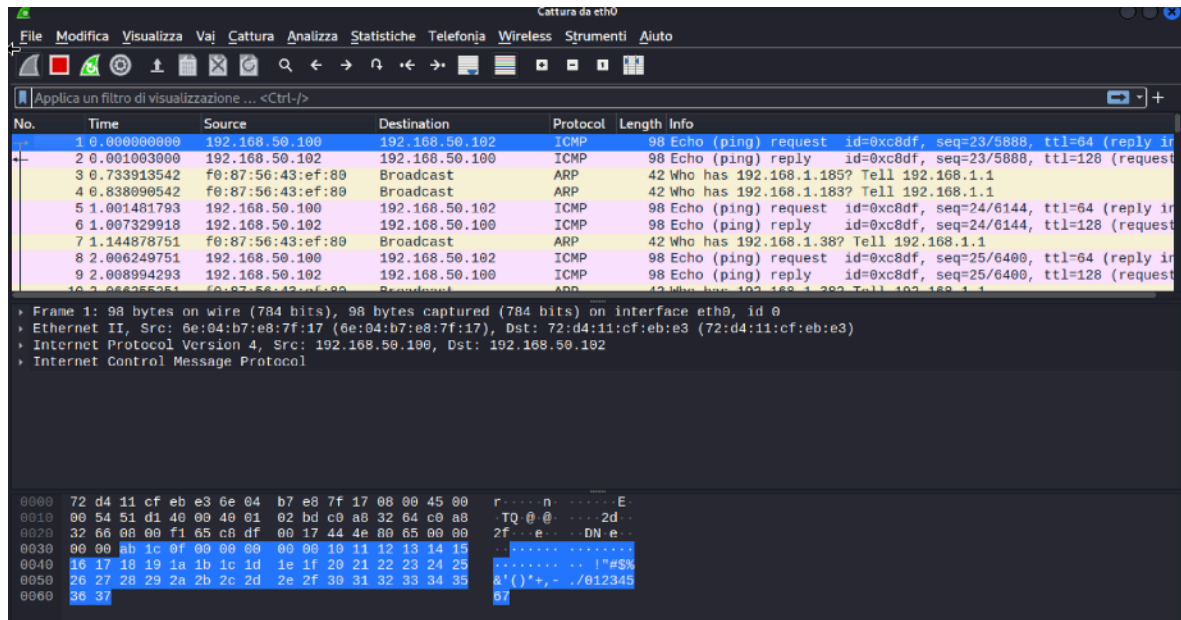
Configurazione:

```
salvatoremorale@kali: ~  
File Azioni Modifica Visualizza Aiuto  
GNU nano 6.2 /etc/inetsim/inetsim.conf  
# start_service chargen_udp  
# start_service dummy_tcp  
# 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  
#  
# Syntax: service_run_as_user <username>  
  
^G Help      ^O Salva      ^W Cerca      ^K Cut        ^T Execute  
^X Esci      ^R Inserisci  ^_ Sostituisci  ^U Paste     ^J Giustifica
```

```
salvatoremorale@kali: ~  
File Azioni Modifica Visualizza Aiuto  
  
(salvatoremorale@kali)-[~]  
$ sudo nano /etc/inetsim/inetsim.conf  
[sudo] password di salvatoremorale:  
  
(salvatoremorale@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 41522) ==  
Session ID: 41522  
Listening on: 192.168.50.100  
Real Date/Time: 2023-12-18 15:15:29  
Fake Date/Time: 2023-12-18 15:15:29 (Delta: 0 seconds)  
Forking services...  
* https_443_tcp - started (PID 41524)  
done.  
Simulation running.  
█
```

Cattura di pacchetti con Wireshark.

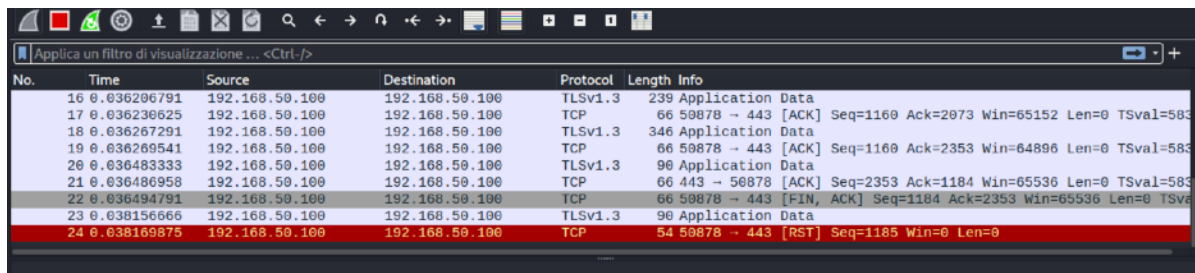
Cattura dei pacchetti sull'interfaccia eth0 durante il ping tra Kali e Windows (servizio https):



The screenshot shows a Wireshark capture of ICMP Echo (ping) packets on interface eth0. The capture filter is 'eth0'. The packet list shows several ping requests and replies between 192.168.50.100 and 192.168.50.102. The packet details pane shows the structure of an ICMP Echo request, including the type, code, identifier, and sequence number. The packet bytes pane shows the raw data of the captured packet.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0xc8df, seq=23/5888, ttl=64 (reply in 0.000000000)
2	0.001003000	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0xc8df, seq=23/5888, ttl=128 (request in 0.001003000)
3	0.733913542	f0:87:56:43:ef:80	Broadcast	ARP	42	Who has 192.168.1.185? Tell 192.168.1.1
4	0.838090542	f0:87:56:43:ef:80	Broadcast	ARP	42	Who has 192.168.1.183? Tell 192.168.1.1
5	1.001481793	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0xc8df, seq=24/6144, ttl=64 (reply in 1.001481793)
6	1.007329918	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0xc8df, seq=24/6144, ttl=128 (request in 1.007329918)
7	1.144878751	f0:87:56:43:ef:80	Broadcast	ARP	42	Who has 192.168.1.38? Tell 192.168.1.1
8	2.006249751	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0xc8df, seq=25/6400, ttl=64 (reply in 2.006249751)
9	2.008994293	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0xc8df, seq=25/6400, ttl=128 (request in 2.008994293)

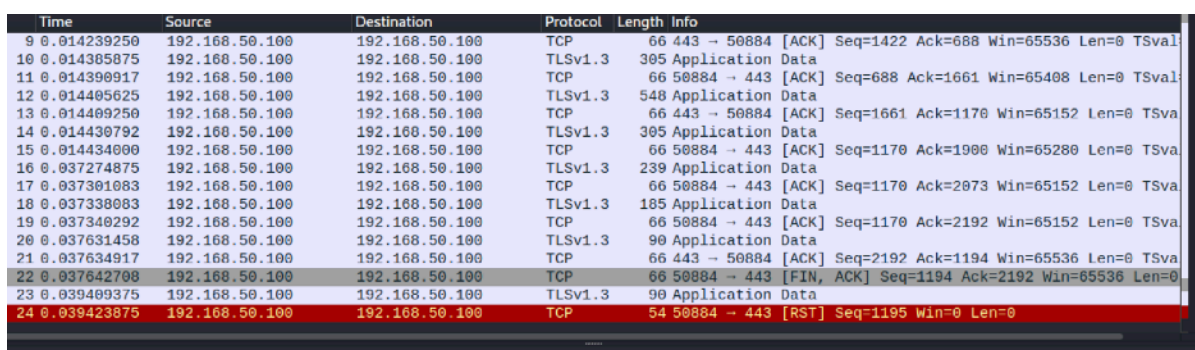
Cattura dei pacchetti sull'interfaccia loopback durante la ricerca nel browser dell'Indirizzo <https://192.168.50.100> (servizio https):



The screenshot shows a Wireshark capture of TLS traffic on the loopback interface. The capture filter is 'lo'. The packet list shows several TLSv1.3 packets, including Application Data, ACK, and FIN. The packet details pane shows the structure of a TLSv1.3 packet, including the version, type, length, and application data. The packet bytes pane shows the raw data of the captured packet.

No.	Time	Source	Destination	Protocol	Length	Info
16	0.036296791	192.168.50.100	192.168.50.100	TLSv1.3	239	Application Data
17	0.036230625	192.168.50.100	192.168.50.100	TCP	66	50878 → 443 [ACK] Seq=1160 Ack=2073 Win=65152 Len=0 TSval=583
18	0.036267291	192.168.50.100	192.168.50.100	TLSv1.3	346	Application Data
19	0.036269541	192.168.50.100	192.168.50.100	TCP	66	50878 → 443 [ACK] Seq=1160 Ack=2353 Win=64896 Len=0 TSval=583
20	0.036483333	192.168.50.100	192.168.50.100	TLSv1.3	90	Application Data
21	0.036486958	192.168.50.100	192.168.50.100	TCP	66	443 → 50878 [ACK] Seq=2353 Ack=1184 Win=65536 Len=0 TSval=583
22	0.036494791	192.168.50.100	192.168.50.100	TCP	66	50878 → 443 [FIN, ACK] Seq=1184 Ack=2353 Win=65536 Len=0 TSval=583
23	0.038156666	192.168.50.100	192.168.50.100	TLSv1.3	90	Application Data
24	0.038169875	192.168.50.100	192.168.50.100	TCP	54	50878 → 443 [RST] Seq=1185 Win=0 Len=0

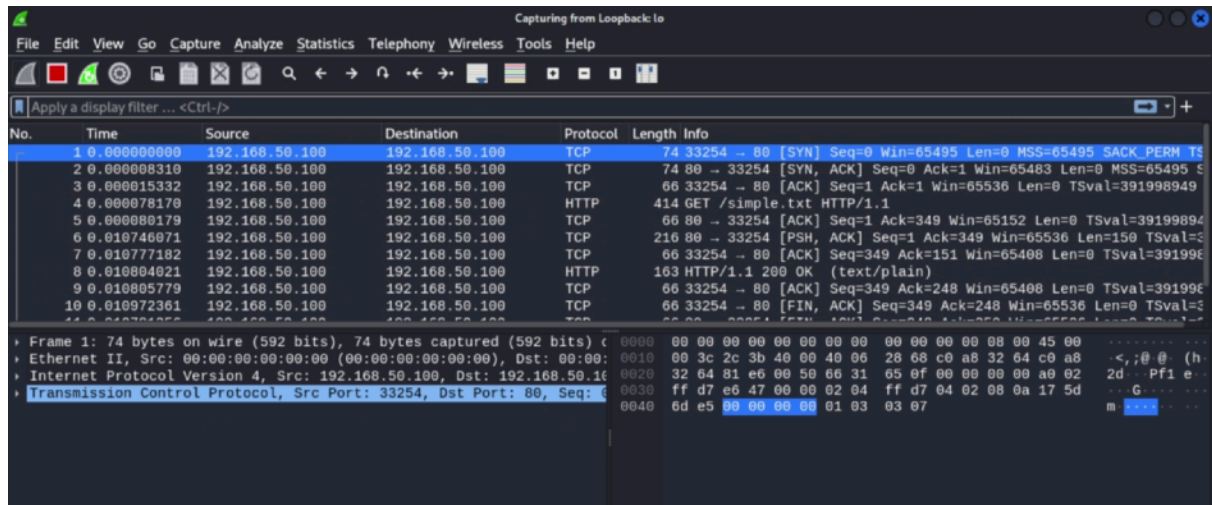
Cattura dei pacchetti sull'interfaccia loopback durante la ricerca nel browser dell'indirizzo <https://192.168.50.100/sample.txt> (servizio https):



The screenshot shows a Wireshark capture of TLS traffic on the loopback interface. The capture filter is 'lo'. The packet list shows several TLSv1.3 packets, including Application Data, ACK, and FIN. The packet details pane shows the structure of a TLSv1.3 packet, including the version, type, length, and application data. The packet bytes pane shows the raw data of the captured packet.

Time	Source	Destination	Protocol	Length	Info
9	0.014239250	192.168.50.100	TCP	66	443 → 50884 [ACK] Seq=1422 Ack=688 Win=65536 Len=0 TSval=583
10	0.014385975	192.168.50.100	TLSv1.3	305	Application Data
11	0.014399917	192.168.50.100	TCP	66	50884 → 443 [ACK] Seq=688 Ack=1661 Win=65408 Len=0 TSval=583
12	0.014405625	192.168.50.100	TLSv1.3	548	Application Data
13	0.014409250	192.168.50.100	TCP	66	443 → 50884 [ACK] Seq=1661 Ack=1170 Win=65152 Len=0 TSval=583
14	0.014430792	192.168.50.100	TLSv1.3	305	Application Data
15	0.014434000	192.168.50.100	TCP	66	50884 → 443 [ACK] Seq=1170 Ack=1900 Win=65280 Len=0 TSval=583
16	0.037274875	192.168.50.100	TLSv1.3	239	Application Data
17	0.037301083	192.168.50.100	TCP	66	50884 → 443 [ACK] Seq=1170 Ack=2073 Win=65152 Len=0 TSval=583
18	0.037338083	192.168.50.100	TLSv1.3	185	Application Data
19	0.037340292	192.168.50.100	TCP	66	50884 → 443 [ACK] Seq=1170 Ack=2192 Win=65152 Len=0 TSval=583
20	0.037631458	192.168.50.100	TLSv1.3	90	Application Data
21	0.037634917	192.168.50.100	TCP	66	443 → 50884 [ACK] Seq=2192 Ack=1194 Win=65536 Len=0 TSval=583
22	0.037642708	192.168.50.100	TCP	66	50884 → 443 [FIN, ACK] Seq=1194 Ack=2192 Win=65536 Len=0 TSval=583
23	0.039409375	192.168.50.100	TLSv1.3	90	Application Data
24	0.039423875	192.168.50.100	TCP	54	50884 → 443 [RST] Seq=1195 Win=0 Len=0

Catturando i pacchetti impostando InetSim sul servizio < http > notiamo che il protocollo TLS (Transport Layer Security) non verrà utilizzato.



Wireshark packet capture showing an HTTP GET request and response over TCP. The interface includes a menu bar, toolbar, packet list, packet details, and packet bytes panes.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.50.100	192.168.50.100	TCP	74	33254 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM T
2	0.000008310	192.168.50.100	192.168.50.100	TCP	74	80 → 33254 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 S
3	0.000015332	192.168.50.100	192.168.50.100	TCP	66	33254 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=391998949
4	0.000078170	192.168.50.100	192.168.50.100	HTTP	414	GET /simple.txt HTTP/1.1
5	0.000080179	192.168.50.100	192.168.50.100	TCP	66	80 → 33254 [ACK] Seq=1 Ack=349 Win=65152 Len=0 TSval=39199894
6	0.010746071	192.168.50.100	192.168.50.100	TCP	216	80 → 33254 [PSH, ACK] Seq=1 Ack=349 Win=65536 Len=150 TSval=3
7	0.010777182	192.168.50.100	192.168.50.100	TCP	66	33254 → 80 [ACK] Seq=349 Ack=151 Win=65408 Len=0 TSval=391998
8	0.010804021	192.168.50.100	192.168.50.100	HTTP	163	HTTP/1.1 200 OK (text/plain)
9	0.010805779	192.168.50.100	192.168.50.100	TCP	66	33254 → 80 [ACK] Seq=349 Ack=248 Win=65408 Len=0 TSval=391998
10	0.010972361	192.168.50.100	192.168.50.100	TCP	66	33254 → 80 [FIN, ACK] Seq=349 Ack=248 Win=65536 Len=0 TSval=3

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: 00:00:00:00:00:00 (00:00:00:00:00:00), Dst: 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: 33254, Dst Port: 80, Seq: 0