

Esercitazione 8 – Wireshark e ping SO Linux a windows con policy firewall che consentono il ping agli indirizzi ip dei sistemi Linux Based – Alessio Russo

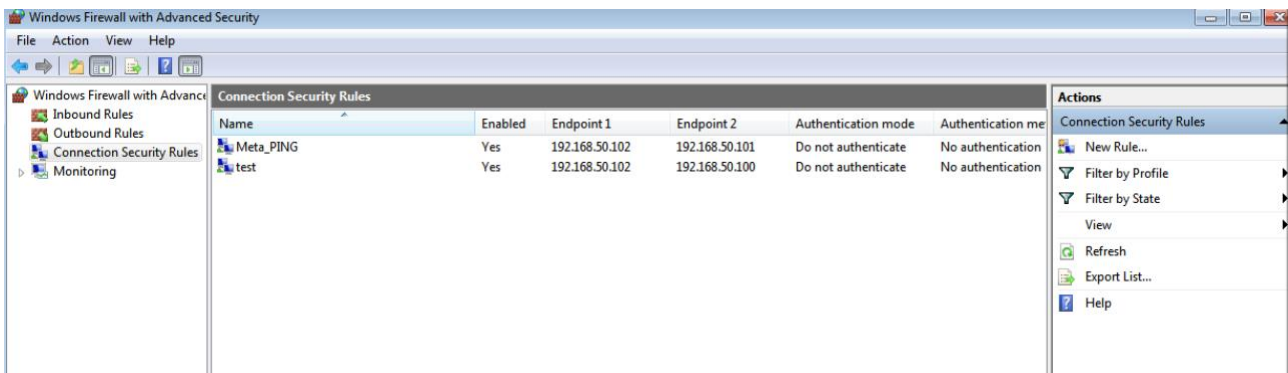
Nello specifico veniva richiesto:

- Configurazione di una policy per permettere il ping da macchine Linux a macchina Windows 7 nel nostro laboratorio;
- Utilizzo dell'utility InetSim per l'emulazione dei servizi Internet;
- Cattura dei pacchetti con Wireshark;

Configurazione policy per ping da macchine Linux a Windows 7

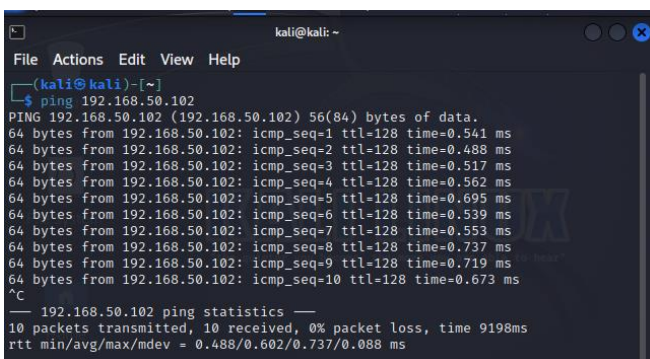
Per configurare una policy che consenta di far comunicare le macchine Linux based con Windows con Firewall attivo bisogna modificare i permessi di connessione all'interno delle impostazioni avanzate di windows, andando ad inserire gli ip di destinazione e quello di partenza delle macchine, consentendo successivamente, la connessione senza aver bisogno di autorizzazioni di alcun tipo. Rispettivamente le 3 macchine hanno i seguenti IP:

- Windows: 192.168.50.102;
- Kali Linux: 192.168.50.100;
- Metasploitable: 192.168.50.101;



Queste sopra sono le policy attive del firewall windows che permettono l'ingresso della connessione dalle macchine Linux. Come possiamo vedere di seguito i sistemi linux pingano senza problemi il sistema windows.

Kali -> Windows



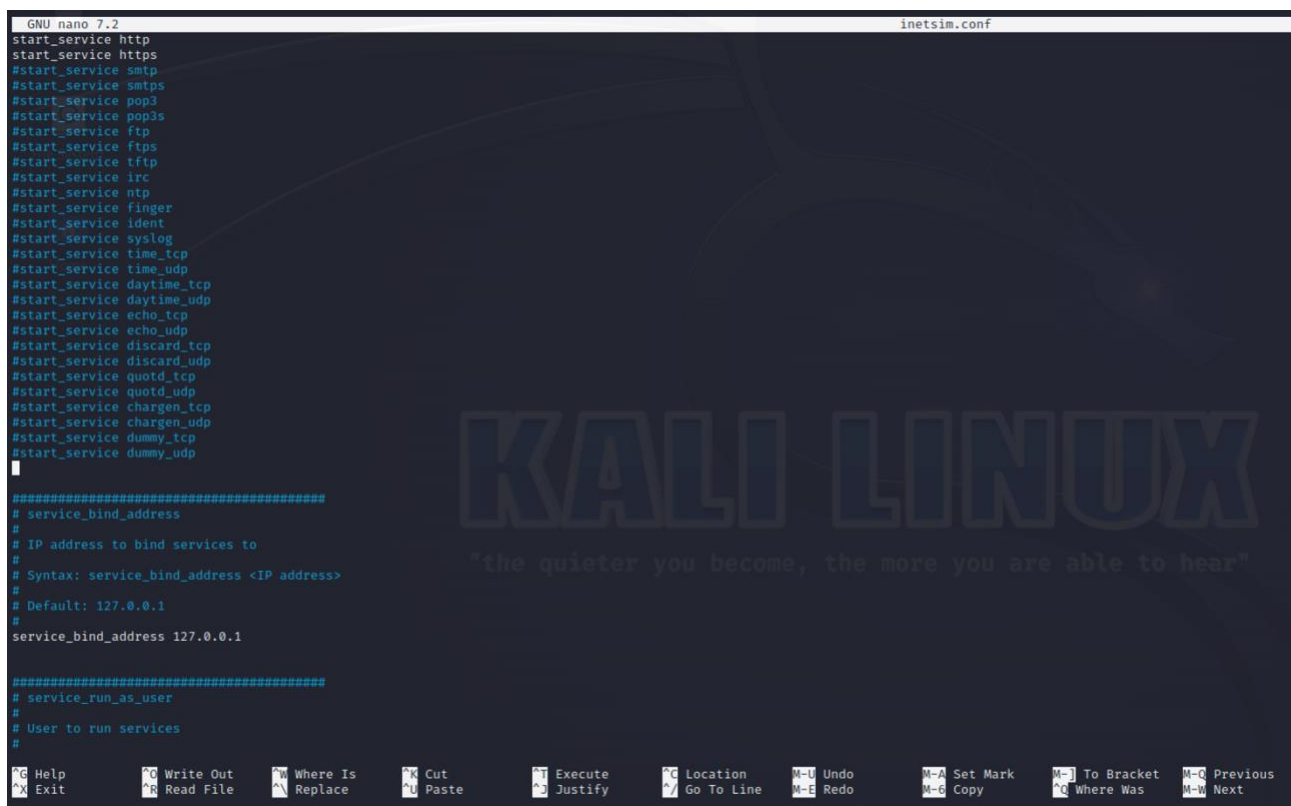
Metasploitable -> Windows

```
msfadmin@metasploitable:~$ 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=14.2 ms
64 bytes from 192.168.50.102: icmp_seq=2 ttl=128 time=0.739 ms
64 bytes from 192.168.50.102: icmp_seq=3 ttl=128 time=0.668 ms
64 bytes from 192.168.50.102: icmp_seq=4 ttl=128 time=0.625 ms
64 bytes from 192.168.50.102: icmp_seq=5 ttl=128 time=0.631 ms
^X64 bytes from 192.168.50.102: icmp_seq=6 ttl=128 time=0.636 ms
64 bytes from 192.168.50.102: icmp_seq=7 ttl=128 time=0.491 ms

--- 192.168.50.102 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6004ms
rtt min/avg/max/mdev = 0.491/2.578/14.256/4.768 ms
msfadmin@metasploitable:~$
```

Per l'utilizzo dell'utility InetSim dobbiamo innanzitutto effettuare la configurazione su macchina virtuale, in questo caso abbiamo scelto Kali linux.

Aprendo la macchina virtuale ci basterà digitare in prompt dei comandi il percorso
Nano /etc/inetSim/ -> ls -> sudo nano inetSim.conf



```
GNU nano 7.2                                inetSim.conf
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
#start_service daytime_tcp
#start_service daytime_udp
#start_service echo_tcp
#start_service echo_udp
#start_service discard_tcp
#start_service discard_udp
#start_service quotd_tcp
#start_service quotd_udp
#start_service chargen_tcp
#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 127.0.0.1

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

#####
# Help
# Exit
# Write Out
# Read File
# Where Is
# Replace
# Cut
# Paste
# Execute
# Justify
# Location
# Go To Line
# Undo
# Redo
# Set Mark
# Copy
# To Bracket
# Where Was
# Previous
# Next
```

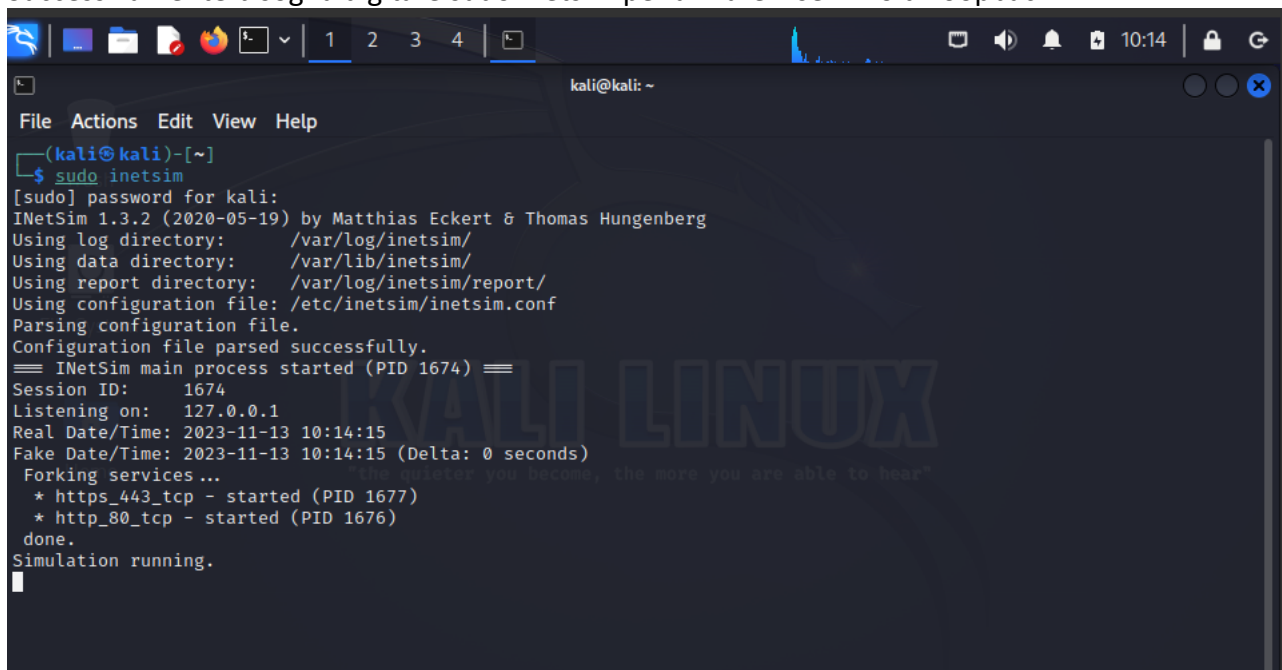
Andando a configurare i parametri come nella seguente immagine.

N.B. i parametri in bianco sono quelli attivi mentre quelli in celestino sono quelli che abbiamo disattivato andando a mettere un '#' avanti ad ogni riga di parametro che non ci interessa utilizzare, in questo caso i servizi attivi sono http e https.

Scendendo andremo a configurare ip del server interno che ci servira per la comunicazione del pacchetto in loopbak. Nello specifico bisognerà togliere il '#' dalla voce service_bind_address 127.0.0.1.

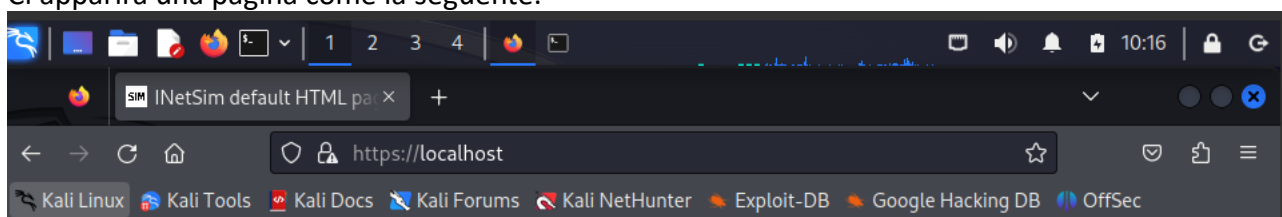
Infine, andremo a salvare la configurazione con ctrl + x -> Y + invio

Successivamente bisogna digitare sudo inetsim per avviare il servizio di loopback:

A terminal window on a Kali Linux system. The user runs 'sudo inetsim'. The terminal shows the password prompt, the version of inetsim (1.3.2), and the paths for log, data, and report directories. It then shows the configuration file being parsed successfully. The main process starts with PID 1674. It lists the session ID, listening port (127.0.0.1), and real/fake dates. It then forks services: https_443_tcp (PID 1677) and http_80_tcp (PID 1676). The simulation is running.

```
(kali@kali)-[~]
$ sudo inetsim
[sudo] password for kali:
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 1674) ==
Session ID:    1674
Listening on:  127.0.0.1
Real Date/Time: 2023-11-13 10:14:15
Fake Date/Time: 2023-11-13 10:14:15 (Delta: 0 seconds)
Forking services...
  * https_443_tcp - started (PID 1677)
  * http_80_tcp  - started (PID 1676)
done.
Simulation running.
```

Una volta avviata la simulazione bisogna innanzitutto aprire il browser e digitare: <https://localhost>
Ci apparirà una pagina come la seguente.



Infine basterà aprire wireshark, selezionare la porta di loopback e avviare la procedura di analisi della rete per far vedere come la richiesta arriva al server e questo la accetta dando un OK alla richiesta GET, che sarà una richiesta https sulla porta 80.

21	4.710626646	127.0.0.1	127.0.0.1	TCP	74	56864 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=33027339 TSecr=0 WS=128
22	4.7106653450	127.0.0.1	127.0.0.1	TCP	74	80 → 56864 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM TSval=33027339 TSecr=33027339 WS=128
23	4.710665115	127.0.0.1	127.0.0.1	TCP	66	56864 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=33027339 TSecr=33027339
24	5.822813813	127.0.0.1	127.0.0.1	HTTP	497	GET / HTTP/1.1
25	5.822859567	127.0.0.1	127.0.0.1	TCP	66	80 → 56864 [ACK] Seq=1 Ack=432 Win=65152 Len=0 TSval=33028452 TSecr=33028452
26	5.842207762	127.0.0.1	127.0.0.1	TCP	216	80 → 56864 [PSH, ACK] Seq=1 Ack=432 Win=65536 Len=150 TSval=33028471 TSecr=33028452 [TCP segment of a reassembled PDU]
27	5.842208334	127.0.0.1	127.0.0.1	TCP	66	56864 → 80 [ACK] Seq=432 Ack=151 Win=65488 Len=0 TSval=33028471 TSecr=33028471
28	5.842303843	127.0.0.1	127.0.0.1	HTTP	324	HTTP/1.1 200 OK (text/html)
29	5.842307276	127.0.0.1	127.0.0.1	TCP	66	56864 → 80 [ACK] Seq=432 Ack=409 Win=65152 Len=0 TSval=33028471 TSecr=33028471
30	5.842761446	127.0.0.1	127.0.0.1	TCP	66	56864 → 80 [FIN, ACK] Seq=432 Ack=409 Win=65536 Len=0 TSval=33028472 TSecr=33028471
31	5.850005887	127.0.0.1	127.0.0.1	TCP	66	80 → 56864 [FIN, ACK] Seq=409 Ack=433 Win=65536 Len=0 TSval=33028479 TSecr=33028472
32	5.850045089	127.0.0.1	127.0.0.1	TCP	66	56864 → 80 [ACK] Seq=433 Ack=410 Win=65536 Len=0 TSval=33028479 TSecr=33028479

Inoltre, poiché le 3 macchine comunicano tra di loro è possibile effettuare l'analisi dei ping scambiati su rete eth0.

Kali -> Metasploitable;

1	0.000000000	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) request id=0x4313, seq=5
2	0.000029807	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) reply id=0x4313, seq=5
3	0.449822644	PcsCompu_aa:75:24	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.101
4	0.842278706	PcsCompu_cb:7e:f5	Broadcast	ARP	42	Who has 192.168.50.102? Tell 192.168.50.101
5	0.995684064	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) request id=0x4313, seq=6
6	0.995749507	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) reply id=0x4313, seq=6
7	1.069990415	PcsCompu_cb:7e:f5	PcsCompu_aa:75:24	ARP	42	Who has 192.168.50.101? Tell 192.168.50.101
8	1.070602119	PcsCompu_aa:75:24	PcsCompu_cb:7e:f5	ARP	60	192.168.50.101 is at 08:00:27:aa:75:24
9	1.867665537	PcsCompu_cb:7e:f5	Broadcast	ARP	42	Who has 192.168.50.102? Tell 192.168.50.101
10	1.991794158	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) request id=0x4313, seq=7
11	1.991862515	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) reply id=0x4313, seq=7
12	2.891036094	PcsCompu_cb:7e:f5	Broadcast	ARP	42	Who has 192.168.50.102? Tell 192.168.50.101
13	2.987918617	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) request id=0x4313, seq=8
14	2.987952192	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) reply id=0x4313, seq=8
15	3.436591007	PcsCompu_aa:75:24	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.101
16	3.915068676	PcsCompu_cb:7e:f5	Broadcast	ARP	42	Who has 192.168.50.102? Tell 192.168.50.101
17	3.983485701	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) request id=0x4313, seq=9
18	3.983508290	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) reply id=0x4313, seq=9
19	4.433399737	PcsCompu_aa:75:24	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.101

Windows – Kali

1	0.000000000	PcsCompu_cb:7e:f5	Broadcast	ARP	42	Who has 192.168.50.102? Tell 192.168.50.101
2	0.000515776	PcsCompu_12:5c:b9	PcsCompu_cb:7e:f5	ARP	60	192.168.50.102 is at 08:00:27:12:5c:b9
3	0.000523339	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=1
4	0.001024389	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=1
5	1.001553050	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=2
6	1.002458290	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=2
7	2.007066159	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=3
8	2.007661179	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=3
9	3.017286803	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=4
10	3.017812315	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=4
11	4.042003783	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=5
12	4.042811803	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=5
13	4.594495730	PcsCompu_12:5c:b9	PcsCompu_cb:7e:f5	ARP	60	Who has 192.168.50.100? Tell 192.168.50.101
14	4.594532554	PcsCompu_cb:7e:f5	PcsCompu_12:5c:b9	ARP	42	192.168.50.100 is at 08:00:27:cb:7e:f5
15	5.043655325	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=6
16	5.044639304	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=6
17	6.046691259	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=7
18	6.047185112	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=7
19	7.049620308	192.168.50.100	192.168.50.102	ICMP	98	Echo (ping) request id=0x2dd3, seq=8
20	7.050183164	192.168.50.102	192.168.50.100	ICMP	98	Echo (ping) reply id=0x2dd3, seq=8

Concludendo si può dire che le richieste sono state effettuate correttamente e che il tutto funziona.