

Policy & Packet Capture - Esercizio di Pratica W3D4

RICHIESTA

L'esercizio di oggi si compone di due fasi: dobbiamo innanzitutto configurare una policy su Windows Firewall che consenta al PC Windows del nostro laboratorio di rispondere alla richiesta di Ping della macchina Kali Linux.

In seconda battuta dobbiamo effettuare una cattura di pacchetti tramite l'utilizzo del software Wireshark in esecuzione sulla macchina virtuale Kali Linux

~~~

## SOLUZIONE

Con le versioni più moderne di Windows - in particolare Windows 10 e Windows 11 - il firewall interno impedisce al PC di rispondere al Ping. È una misura di sicurezza, che ho aggirato creando una regola nel firewall chiamata "Consenti Ping".

La regola che ho creato consente di aggirare le impostazioni predefinite di Windows 11, ma solo per i computer che ho autorizzato. Eventuali computer non inclusi nelle definizioni delle regole, non riceveranno risposta alla richiesta di Ping.

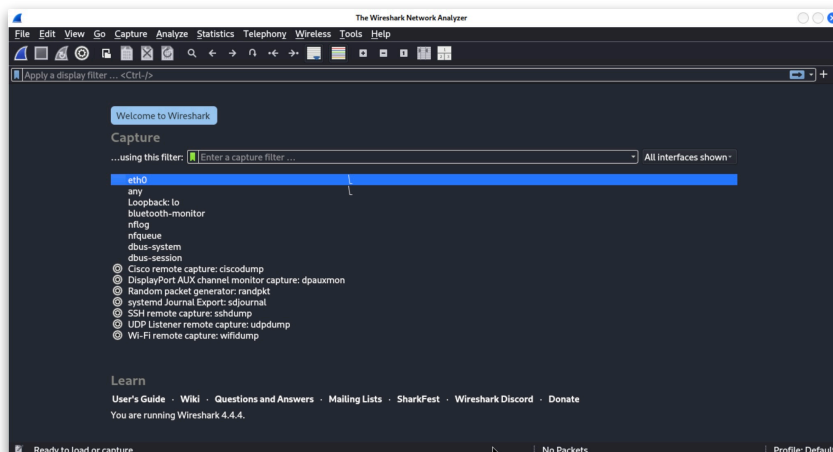
La seconda parte dell'esercizio verte su Wireshark, un software che ci consente di vedere nel dettaglio tutte le comunicazioni che avvengono fra i computer della rete. Per testarne il funzionamento, ho messo in comunicazione la macchina virtuale con Kali Linux e quella con Metasploitable 2, sempre attraverso una richiesta di Ping.

Wireshark ha tenuto nota e registrato tutta la sequenza di passaggi di pacchetti e, grazie a questa registrazione, è possibile ottenere molte preziose informazioni su questo processo. Le informazioni possono essere salvate e analizzate secondo necessità.

~~~

IMMAGINI

Nell'immagine di fianco, la schermata iniziale di Wireshark, in esecuzione sulla macchina virtuale Kali Linux del nostro laboratorio

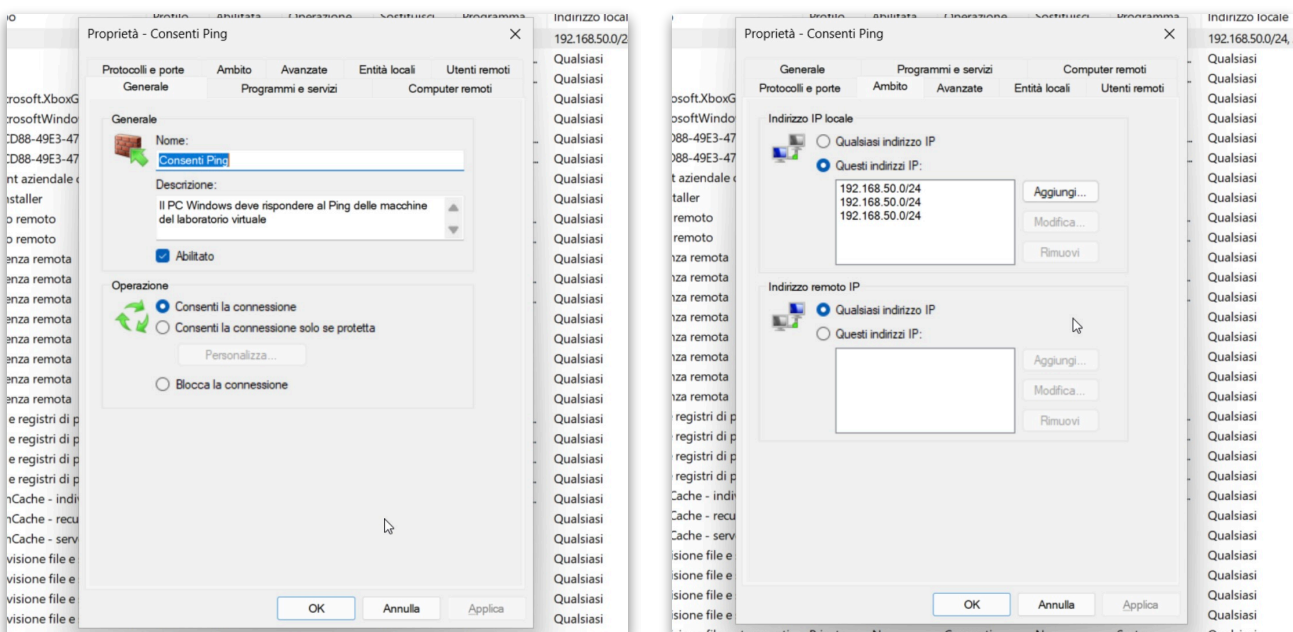


DETTAGLI TECNICI

Dalle impostazioni avanzate del Firewall di Windows 11, ho creato la policy "Consenti Ping", impostando il protocollo interessato (ICMPv4) e, nella sezione Ambito, gli indirizzi IP dei computer per cui vogliamo che Windows 11 risponda al Ping. Nello specifico, nella sezione Ambito ho inserito i seguenti IP:

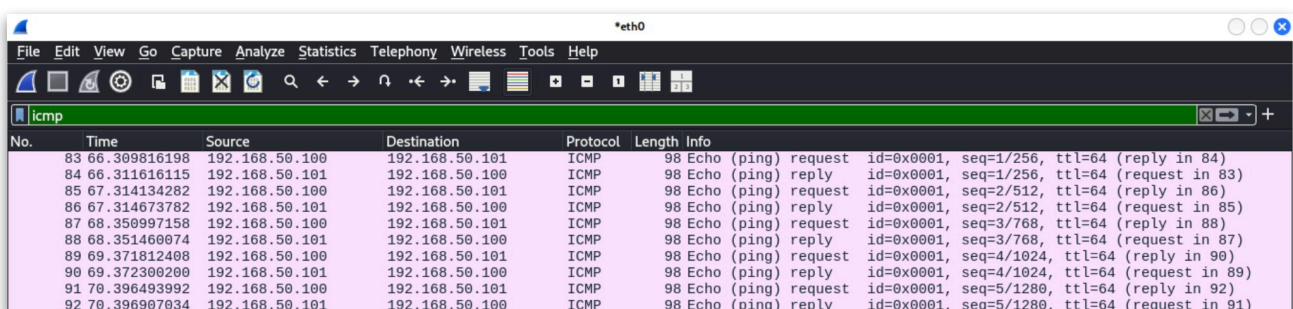
- **192.168.50.100/24** (l'indirizzo della VM Kali Linux)
- **192.168.50.101/24** (l'indirizzo della VM Metasploitable)
- **192.168.50.102/24** (l'indirizzo della VM Windows 7)

Essendo la prima regola in ordine gerarchico a governare il funzionamento del protocollo ICMPv4, Windows 11 risponderà al Ping da queste tre macchine nonostante la policy generale gli imponga di non farlo. Per via della policy generale, però, Windows 11 continuerà a non rispondere ad eventuali richieste di Ping di altre macchine, dato l'IP diverso da quelli autorizzati.



Per quel che riguarda invece la seconda parte dell'esercizio, utilizziamo Wireshark, un analizzatore di pacchetti di rete in transito sulle diverse interfacce di rete. Questo software consente di ispezionare dettagliatamente tutte le informazioni relative ai singoli pacchetti.

Per testarne il funzionamento, ho inviato una richiesta di Ping dalla nostra VM con Kali Linux alla macchina VM con Metasploitable 2. Nell'interfaccia di Wireshark ho poi isolato i pacchetti relativi a questa comunicazione utilizzando il filtro "ICMP", il protocollo che gestisce appunto il Ping.



No.	Time	Source	Destination	Protocol	Length	Info
83	66.309816198	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=1/256, ttl=64 (reply in 84)
84	66.311616115	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256, ttl=64 (request in 83)
85	67.314134282	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=2/512, ttl=64 (reply in 86)
86	67.314673782	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=2/512, ttl=64 (request in 85)
87	68.350997158	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=3/768, ttl=64 (reply in 88)
88	68.351460074	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=3/768, ttl=64 (request in 87)
89	69.371812408	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=4/1024, ttl=64 (reply in 90)
90	69.372300200	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=4/1024, ttl=64 (request in 89)
91	70.396493992	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=64 (reply in 92)
92	70.396907034	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=5/1280, ttl=64 (request in 91)

Dalle informazioni presentate da Wireshark, possiamo vedere ben schematizzato il processo di richiesta di Ping, con le cinque "Request" della VM con Kali Linux (IP 192.168.50.100/24) e le cinque "Reply" positive della VM con Metasploitable 2 (IP 192.168.50.101/24).

Selezionando uno di questi pacchetti, è possibile ottenere informazioni più dettagliate. Nello specifico, possiamo ottenere informazioni sul MAC Address del mittente, sul MAC Address del destinatario, sugli indirizzi IP di mittente e destinatario, tutte le fasi di comunicazione secondo il modello ISO/OSI e il contenuto - in esadecimale - del pacchetto.

The image shows a Wireshark capture of network traffic on the *eth0 interface. The packet list pane displays a series of ICMP Echo (ping) requests and replies. The selected packet (No. 89) is an ICMP Echo (ping) request from 192.168.50.100 to 192.168.50.101. The packet details pane shows the structure of the packet: Ethernet II, Internet Protocol Version 4, and Internet Control Message Protocol. The packet bytes pane displays the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
83	66.309816198	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=1/256, ttl=64 (reply in 84)
84	66.311616115	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=1/256, ttl=64 (request in 83)
85	67.314134282	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=2/512, ttl=64 (reply in 86)
86	67.314673782	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=2/512, ttl=64 (request in 85)
87	68.350997158	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=3/768, ttl=64 (reply in 88)
88	68.351460074	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=3/768, ttl=64 (request in 87)
89	69.371812408	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=4/1024, ttl=64 (reply in 90)
90	69.372300200	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=4/1024, ttl=64 (request in 89)
91	70.396493992	192.168.50.100	192.168.50.101	ICMP	98	Echo (ping) request id=0x0001, seq=5/1280, ttl=64 (reply in 92)
92	70.396907034	192.168.50.101	192.168.50.100	ICMP	98	Echo (ping) reply id=0x0001, seq=5/1280, ttl=64 (request in 91)

Frame 89: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
Ethernet II, Src: c6:52:ad:b4:67:4a (c6:52:ad:b4:67:4a), Dst: 82:d5:c4:b2:c0:72
Internet Protocol Version 4, Src: 192.168.50.100, Dst: 192.168.50.101
Internet Control Message Protocol

0000 82 d5 c4 b2 c0 72 c6 52 ad b4 67 4a 08 00 45 00R.R.gJ.E
0010 00 54 65 ad 40 00 40 01 ee e1 c0 a8 32 64 c0 a8 ..Te @ @ ...2d
0020 32 65 08 00 de dc 00 01 00 04 41 0a d4 67 00 00 2eA.g
0030 00 00 38 d9 0c 00 00 00 00 10 11 12 13 14 15 ..8
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25!"#\$%
0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 &'()*+,-./012345
0060 36 37 67