

# Esercitazione 8

## Wireshark e Windows Firewall

Ettore Farris - 11/11/2023

### 1) Descrizione sintetica dell'esercitazione

L'esercitazione è finalizzata a consolidare le conoscenze su Wireshark e sul funzionamento del Firewall.

#### **Consegna dell'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*

#### **Caratteristiche del Lab:**

Il nostro LAB è composto da 3 macchine virtuali appartenenti alla stessa **rete interna** di virtual box:

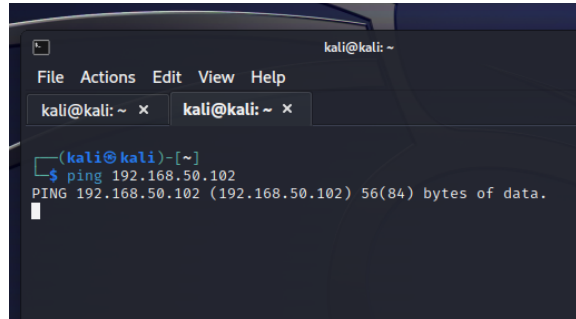
- Kali Linux (IP: 192.168.50.100)
- Metasploitable (IP: 192.168.50.101)
- Windows 7 (IP: 192.168.50.102)

## 2) Configurazione Policy Firewall

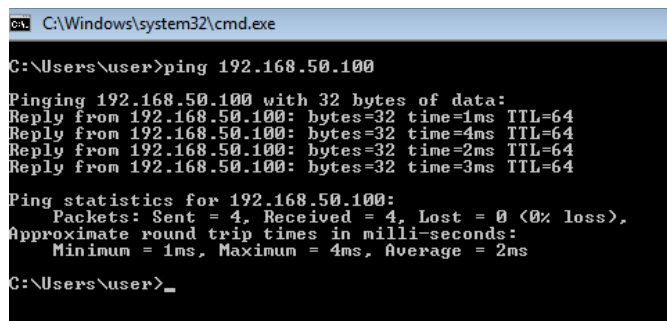
### - **Windows Firewall blocca i ping**

Il firewall di Windows 7 di default non consente traffico in arrivo da altri dispositivi. Per consentire il ping da una macchina appartenente alla stessa rete locale bisogna pertanto impostare una regola inbound nelle impostazioni avanzate di Windows Firewall.

Se infatti proviamo a pingare la macchina Windows 7 da quella Kali Linux, questa invierà i pacchetti senza successo. La macchina Kali Linux non riesce a comunicare con quella Windows 7 per via dell'impostazione del Firewall. Windows tuttavia può pingare Kali Linux con successo.



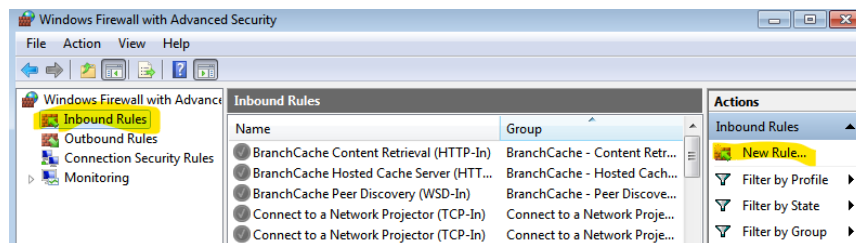
```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ x kali@kali: ~ x  
(kali@kali)-[~]  
$ ping 192.168.50.102  
PING 192.168.50.102 (192.168.50.102) 56(84) bytes of data:  
_
```



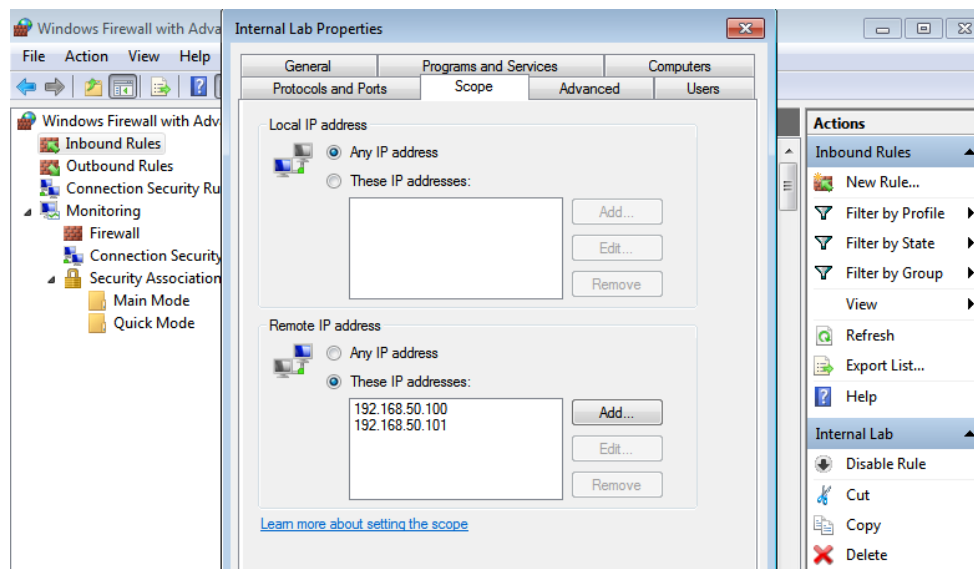
```
C:\Windows\system32\cmd.exe  
C:\Users\user>ping 192.168.50.100  
Pinging 192.168.50.100 with 32 bytes of data:  
Reply from 192.168.50.100: bytes=32 time=1ms TTL=64  
Reply from 192.168.50.100: bytes=32 time=4ms TTL=64  
Reply from 192.168.50.100: bytes=32 time=2ms TTL=64  
Reply from 192.168.50.100: bytes=32 time=3ms TTL=64  
Ping statistics for 192.168.50.100:  
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
Minimum = 1ms, Maximum = 4ms, Average = 2ms  
C:\Users\user>
```

### - **Configurazione Policy Windows Firewall dalle impostazioni avanzate**

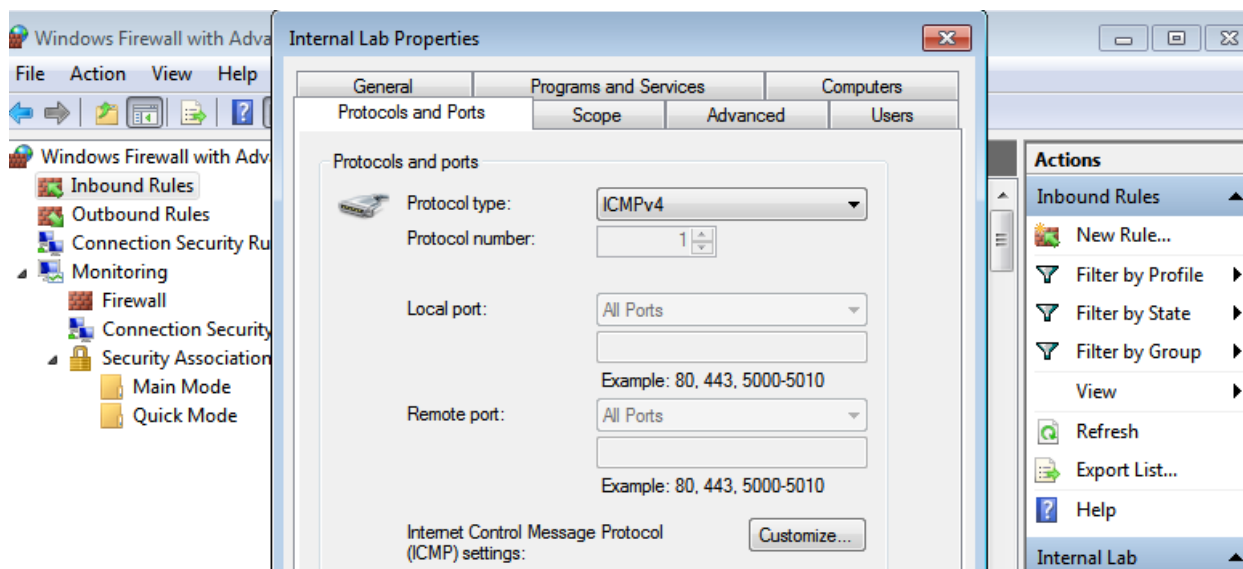
Sulla macchina Windows 7, dalle impostazioni del Firewall impostiamo una nuova regola inbound specificando gli IP a cui è concesso inviare traffico e limitando il traffico solo per i pacchetti ICMPv4 (ping).



Su "Scope" impostiamo "Local IP Address" (ovvero l'IP della macchina ricevente, quindi Windows 7 stessa) su "Any IP Address". Su "Remote IP Address" (cioè le macchine mittenti) specifichiamo gli indirizzi IP a cui è consentito inviare traffico. In questo caso, le macchine Metasploitable (192.168.50.101) e Kali Linux (192.168.50.100).

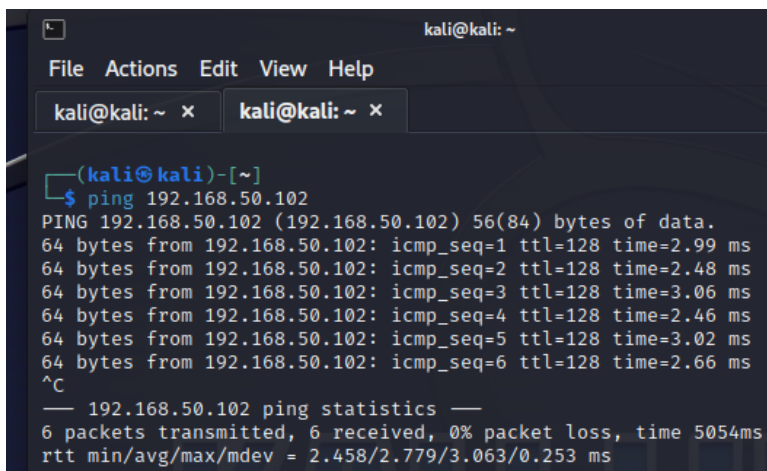


Nella tab "Protocols and Ports" impostiamo "Protocol Type" su "ICMPv4" per permettere solo il traffico ping dalle macchine della nostra rete.



- **Windows permette il traffico in entrata dalla macchina Kali Linux**

La regola appena creata consente a Kali Linux di inviare ping a Windows 7 con successo.



```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ x kali@kali: ~ x  
(kali@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=2.99 ms  
64 bytes from 192.168.50.102: icmp_seq=2 ttl=128 time=2.48 ms  
64 bytes from 192.168.50.102: icmp_seq=3 ttl=128 time=3.06 ms  
64 bytes from 192.168.50.102: icmp_seq=4 ttl=128 time=2.46 ms  
64 bytes from 192.168.50.102: icmp_seq=5 ttl=128 time=3.02 ms  
64 bytes from 192.168.50.102: icmp_seq=6 ttl=128 time=2.66 ms  
^C  
— 192.168.50.102 ping statistics —  
6 packets transmitted, 6 received, 0% packet loss, time 5054ms  
rtt min/avg/max/mdev = 2.458/2.779/3.063/0.253 ms
```

### 3) Utilizzo dell'utility InetSim per l'emulazione di servizi Internet su Kali Linux

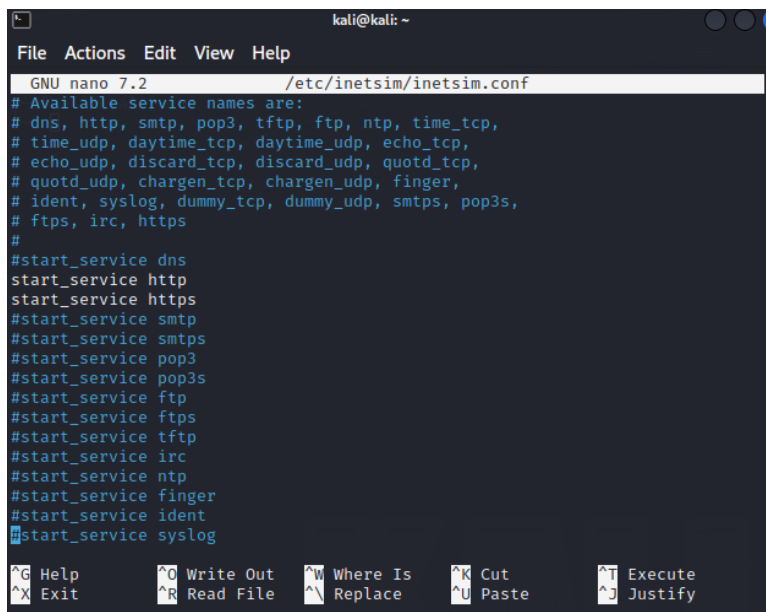
Per utilizzare Wireshark e intercettare traffico, utilizzeremo INetSim, un software che serve per simulare servizi internet in un lab come il nostro. Nel nostro caso, INetSim ci consentirà di accedere a un sito internet fittizio su localhost simulando una connessione reale. Il localhost corrisponde all'indirizzo IP 127.0.0.1 comunemente conosciuto come indirizzo di *loopback*.

- **Configurazione di INetSim**

Prima di usare il software, modifichiamo alcune impostazioni nel file di configurazione col seguente comando:

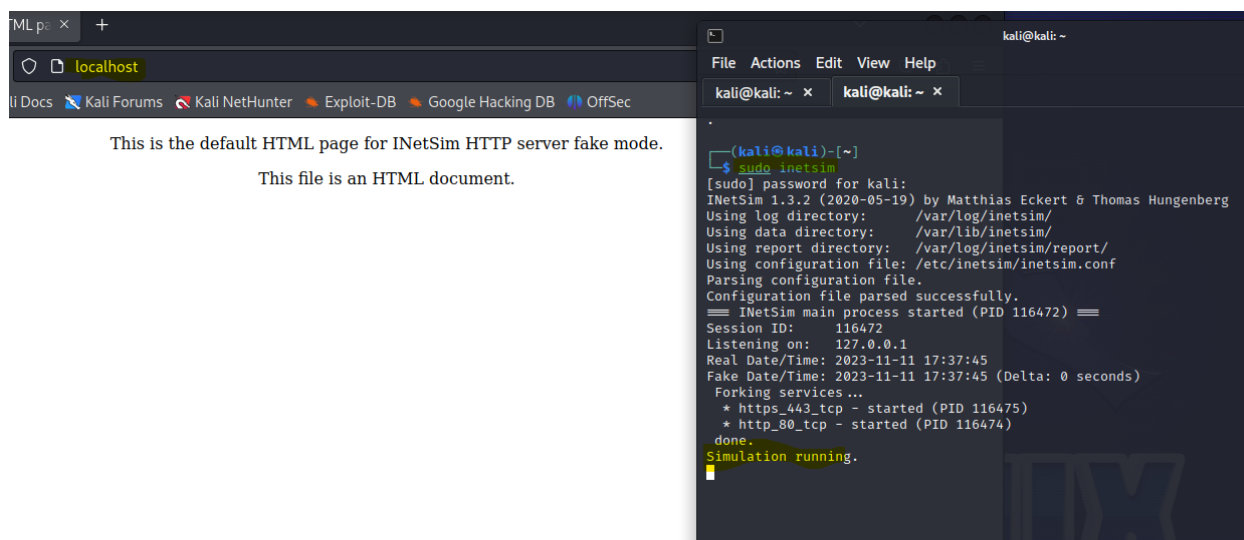
*"sudo nano /etc/inetsim/inetsim.conf"*

Per prima cosa commentiamo tutti i servizi tranne quelli http e https che ci consentiranno di accedere al sito internet e salviamo il file.



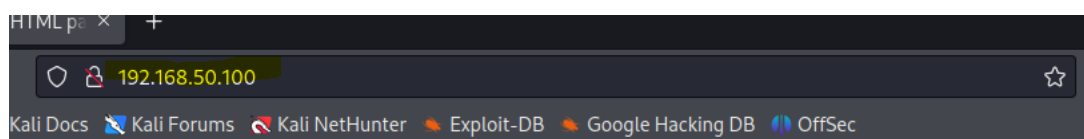
```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/inetsim/inetsim.conf  
# Available service names are:  
# dns, http, smtp, pop3, tftp, ftp, ntp, time_tcp,  
# time_udp, daytime_tcp, daytime_udp, echo_tcp,  
# echo_udp, discard_tcp, discard_udp, quotd_tcp,  
# quotd_udp, chargen_tcp, chargen_udp, finger,  
# ident, syslog, dummy_tcp, dummy_udp, smtps, pop3s,  
# ftps, irc, https  
#  
#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  
#start_service finger  
#start_service ident  
#start_service syslog  
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute  
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify
```

Lanciando il programma con “*sudo inetsim*” inizierà la simulazione e un server virtuale verrà attivato. Aprendo il browser e visitando l’indirizzo “*http://localhost*” verrà visualizzato un sito internet.



Qualora si volesse accedere al sito internet con un altro indirizzo IP e non con localhost (impostazione di default del programma) bisogna togliere il commento a “*service\_bind\_address*” e scrivere l’IP desiderato.

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



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

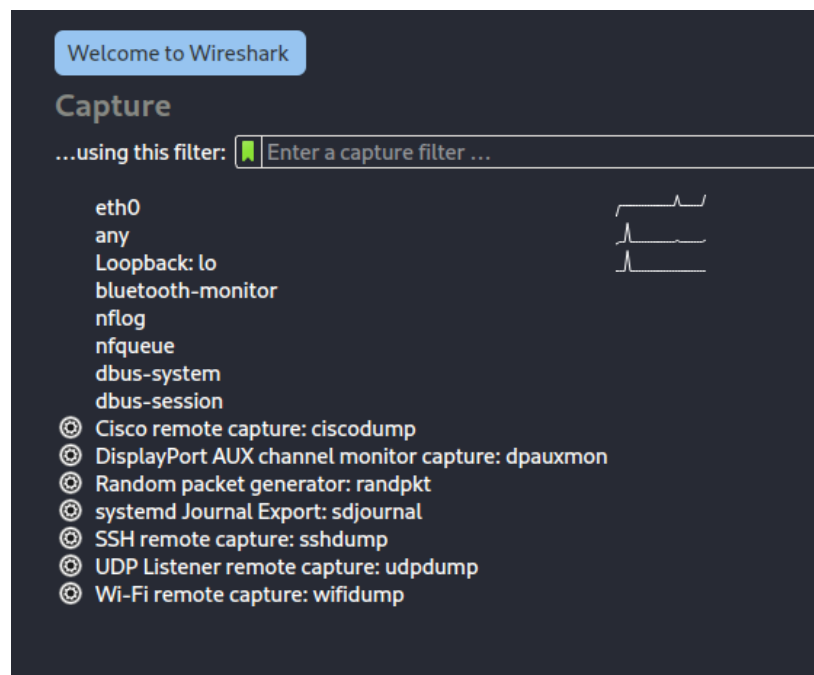
This file is an HTML document.

Se ad esempio impostiamo l’ip della macchina virtuale stesso, verrà visualizzato sempre lo stesso sito internet. Sia in questo caso che con lavorando con l’IP localhost, il PC è come se contattasse se stesso per simulare una connessione a un server esterno. L’unica differenza è che con localhost il traffico passa per l’interfaccia di loopback, mentre con l’ip della macchina virtuale stessa passa dalla scheda di rete in uso, in questo caso eth0. Vedremo nel dettaglio questo aspetto nella sezione su Wireshark.

## 4) Cattura di pacchetti con Wireshark

Wireshark è uno strumento di analisi del traffico di rete che consente agli utenti di catturare e ispezionare i pacchetti di dati che transitano attraverso una rete. Selezionando la scheda di rete, si può analizzare il traffico che passa per questa. In questa esercitazione vedremo due casi:

- il traffico che passa per l'interfaccia di loopback, ovvero quella relativa all'indirizzo localhost (127.0.0.1).
- il traffico che passa per l'interfaccia eth0;



### - **Traffico dell'interfaccia di loopback**

Apriamo una simulazione con INetSim con l'indirizzo di default, ovvero localhost. Visitando il sito sul browser, è come se stessimo inviando delle richieste GET a un server reale. Il traffico, come detto prima, passa per l'interfaccia di loopback e pertanto, selezionandola, saremo in grado di intercettarlo su Wireshark.

Visitando l'indirizzo *"http://localhost"* dal browser, si intercettano i pacchetti in transito su Wireshark. Nella foto sotto, si possono vedere la *three-way-handshake*, la richiesta GET sulla porta 80 e la risposta 200 del server.

The screenshot displays the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with icons for various functions like opening files, saving, and zooming.

The main pane shows a list of captured packets. A filter "Apply a display filter ... <Ctrl-/" is applied at the top right. The packet list has columns for No., Time, Source, Destination, Protocol, Length, and Info. Packets 1 through 12 are visible, all originating from 127.0.0.1 and destined for 127.0.0.1. Packet 1 is a TCP SYN packet from port 57590 to port 80. Subsequent packets show the progression of the connection attempt, including ACKs and FINs.

The bottom pane shows the details of the selected packet (No. 1). It identifies the frame as 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). The next layer is Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1. The final layer is Transmission Control Protocol, Src Port: 57590, Dst Port: 80. The hex dump below shows the raw bytes of the packet, starting with ff d7 fe 30 00 00 c1 d6 1c 4c 00 00 00 a0 00 00 ff d1 00 00 00 01 03 03 07.

Essendo un sito in HTTP, quindi non criptato, possiamo intercettare e leggere il contenuto html della pagina.

Tools Help

Protocol	Length	Info
TCP	74	57590 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PER
TCP	74	80 → 57590 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=654
TCP	66	57590 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=120012
HTTP	497	GET / HTTP/1.1
TCP	66	80 → 57590 [ACK] Seq=1 Ack=432 Win=65152 Len=0 TSval=1200
TCP	216	80 → 57590 [PSH, ACK] Seq=1 Ack=432 Win=65536 Len=150 TSv
TCP	66	57590 → 80 [ACK] Seq=432 Ack=151 Win=65408 Len=0 TSval=12
HTTP	324	HTTP/1.1 200 OK (text/html)
TCP	66	57590 → 80 [ACK] Seq=432 Ack=409 Win=65152 Len=0 TSval=12
TCP	66	57590 → 80 [FIN, ACK] Seq=432 Ack=409 Win=65536 Len=0 TSv
TCP	66	80 → 57590 [FIN, ACK] Seq=409 Ack=433 Win=65536 Len=0 TSv
TCP	66	57590 → 80 [ACK] Seq=433 Ack=410 Win=65536 Len=0 TSval=12

3c 68 74 6d 6c 3e 0a 20 20 20 3c 68 65 61 64	p.<html> . <head
20 20 20 20 3c 74 69 74 6c 65 3e 49 4e 65	> . <title>Ine
69 6d 20 64 65 66 61 75 6c 74 20 48 54 4d	tSim def ault HTM
7f 61 67 65 3c 2f 74 69 74 6c 65 3e 0a 20	L page< /title>.
2f 68 65 61 64 3e 0a 20 20 20 3c 62 6f 64 79	</head> . <body
20 20 20 20 3c 70 3e 3c 2f 70 3e 0a 20 20	> . <p ></p>.
3c 70 20 61 6c 69 67 6e 3d 22 63 65 6e 74	<p all gn="cent
22 3e 54 68 69 73 20 69 73 20 74 68 65 20	er">This is the
66 75 6c 74 20 48 54 4d 4c 20 70 61 67	default HTML pag
66 6f 72 20 49 4e 65 74 53 69 6d 20 48 54	for IN etSim HT
20 73 65 72 76 65 72 20 66 61 6b 65 20 6d	TP serve r fake m
65 2e 3c 2f 70 3e 0a 20 20 20 20 3c 70 20	ode.</p> . <p



## - **Traffico dell'interfaccia eth0**

Per testare l'interfaccia eth0 possiamo pingare la macchina Windows 7 in modo da generare traffico su quella scheda di rete.

```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ x kali@kali: ~ x  
(kali@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=2.99 ms  
64 bytes from 192.168.50.102: icmp_seq=2 ttl=128 time=2.48 ms  
64 bytes from 192.168.50.102: icmp_seq=3 ttl=128 time=3.06 ms  
64 bytes from 192.168.50.102: icmp_seq=4 ttl=128 time=2.46 ms  
64 bytes from 192.168.50.102: icmp_seq=5 ttl=128 time=3.02 ms  
64 bytes from 192.168.50.102: icmp_seq=6 ttl=128 time=2.66 ms
```

Su Wireshark nel frattempo, selezioniamo l'interfaccia e vediamo immediatamente il traffico dei pacchetti ICMP tra le due macchine.

