


## ESERCIZIO FACOLTATIVO M1W3D

### Introduzione ed obiettivo

L'esercitazione ha previsto la simulazione di un servizio DNS su Kali Linux tramite INetSim e la successiva cattura del traffico generato da un client Windows.

L'obiettivo è stato allestire un ambiente di prova semplice e riproducibile, attivare il servizio DNS emulato, inviare interrogazioni dal client e registrare le transazioni con Wireshark per esaminarne i dettagli ; in questo modo si è potuto verificare concretamente che le query vengono generate dal client, raggiungono il server simulato e che le relative trame DNS sono effettivamente presenti nel file di cattura fornendo così evidenze chiare e utilizzabili per la valutazione tecnica dell'esercitazione.

### PASSAGGIO 1



```
# INetSim configuration file
#
#####

#####

# Main configuration
#####

#####

start_service dns
```

Nel primo passaggio è stata configurata la macchina Kali Linux per l'attivazione del servizio DNS all'interno di INetSim.

Aperto il file di configurazione principale, situato nel percorso

“/etc/inetsim/inetsim.conf”, è stata rimossa la riga di commento e aggiunta la direttiva start service dns. In questo modo, al momento dell'avvio di INetSim, il servizio DNS viene automaticamente attivato e la macchina Kali diventa in grado di rispondere alle richieste di risoluzione dei nomi provenienti dalla rete di laboratorio. Questa operazione rappresenta la fase di preparazione necessaria per permettere l'emulazione del comportamento di un server DNS reale e costituisce la base per le prove successive di sniffing e analisi del traffico.

## PASSAGGIO 2

```
(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 7913) ==  
Session ID: 7913  
Listening on: 192.168.50.100  
Real Date/Time: 2025-10-16 08:38:25  
Fake Date/Time: 2025-10-16 08:38:25 (Delta: 0 seconds)  
Forking services ...  
* dns_53_tcp_udp - started (PID 7915)  
Can't locate object method "main_loop" via package "Net::DNS::Nameserver" at  
/usr/share/perl5/INetSim/DNS.pm line 69.  
* http_80_tcp - started (PID 7916)  
* https_443_tcp - started (PID 7917)  
done.  
Simulation running.  
█
```

INetSim è stato avviato sulla macchina Kali eseguendo il comando di avvio ed, al termine della procedura, la schermata ha restituito la conferma testuale “Simulation running” indicando in modo univoco che l’ambiente di simulazione è operativo ed i servizi fittizi sono attivi e pronti a rispondere alle richieste provenienti dalla rete locale.

Questo passaggio segna il punto in cui la macchina Kali assume il ruolo di server simulato, rendendosi disponibile per ricevere e registrare le connessioni provenienti dal client Windows durante le successive prove di comunicazione.

## PASSAGGIO3

cmd Prompt dei comandi

```
Microsoft Windows [Versione 10.0.10240]
(c) 2015 Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\user>nslookup www.google.com 192.168.50.100
DNS request timed out.
    timeout was 2 seconds.
Server: UnKnown
Address: 192.168.50.100

DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
DNS request timed out.
    timeout was 2 seconds.
*** Tempo scaduto per la richiesta a UnKnown

C:\Users\user>
```

Wireshark interface showing network traffic on eth0.

Filter: Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length
103	113.378960105	192.168.50.100	192.168.50.102	TCP	60
104	113.379674744	192.168.50.102	192.168.50.100	TCP	60
105	113.379674847	192.168.50.102	192.168.50.100	TCP	60
106	113.401468341	192.168.50.102	192.168.50.100	HTTP	504
107	113.401502317	192.168.50.100	192.168.50.102	TCP	54
108	113.427944200	192.168.50.100	192.168.50.102	TCP	204
109	113.433953544	192.168.50.100	192.168.50.102	HTTP	312
110	113.437028536	192.168.50.102	192.168.50.100	TCP	60
111	113.471509123	192.168.50.102	192.168.50.100	TCP	60
112	113.471540407	192.168.50.100	192.168.50.102	TCP	54
113	113.865282734	PCSSystemtec 4e:73:...	Broadcast	ARP	60

Frame 109: 312 bytes on wire (2496 bits)

- Ethernet II, Src: PCSSystemtec\_42:b7:9c
- Internet Protocol Version 4, Src: 192.168.50.100, Dst: 192.168.50.102
- Transmission Control Protocol, Src Port: 54444, Dst Port: 80
- [2 Reassembled TCP Segments (408 bytes)]
- Hypertext Transfer Protocol
- Line-based text data: text/html (10 lines)

Hex dump (Frame 109):

Offset	Hex
0000	08 00 27 4e 73 95 08 00 27 42 b7
0010	01 2a b4 1b 40 00 40 06 9f 97 c0
0020	32 66 00 50 c1 2d ac c7 b8 fe 3f
0030	01 f5 e7 37 00 00 3c 68 74 6d 6c
0040	68 65 61 64 3e 0a 20 20 20 20 3c
0050	3e 49 4e 65 74 53 69 6d 20 64 65
0060	20 48 54 4d 4c 20 70 61 67 65 3c
0070	65 3e 0a 20 20 3c 2f 68 65 61 64
0080	62 6f 64 79 3e 0a 20 20 20 20 3c
0090	3e 0a 20 20 20 20 3c 70 20 61 6c

Frame (312 bytes)    Reassembled TCP (408 bytes)

Dopo aver avviato l'ambiente di simulazione, dal terminale Windows è stato eseguito il comando `nslookup` per verificare la risoluzione di un dominio attraverso il server DNS simulato da INetSim.

In parallelo, tramite Wireshark da Kali, è stato possibile osservare in tempo reale il traffico generato dal client: le richieste DNS partite dall'indirizzo IP di Windows sono state catturate e analizzate confermando che la comunicazione è effettivamente avvenuta e che i pacchetti hanno raggiunto il server simulato.

Questo risultato evidenzia il corretto funzionamento dello sniffing di rete e dimostra come, anche in assenza di una risposta reale, sia possibile visualizzare e studiare il contenuto dei pacchetti trasmessi.

## ESERCIZIO FACOLTATIVO 2

L'obiettivo di questa parte del laboratorio è stato quello di verificare il corretto funzionamento della comunicazione tra la macchina Windows e la macchina Kali Linux con INetSim attivo, simulando uno scenario realistico di traffico di rete.

L'attività ha previsto due azioni principali: da un lato, la generazione di richieste da parte del client Windows verso il server Kali per testare la connettività e i servizi simulati; dall'altro, la cattura e l'analisi del traffico con Wireshark, al fine di osservare e comprendere come i pacchetti vengano effettivamente scambiati tra i due sistemi.

In questo modo l'esercizio ha permesso di collegare la teoria sulla comunicazione di rete con un'osservazione pratica, fornendo una visione diretta del flusso dei pacchetti e del comportamento del server di simulazione.

## PASSAGGIO 1

```

C:\> Prompt dei comandi

Microsoft Windows [Versione 10.0.10240]
(c) 2015 Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\user>ping -n 10 192.168.50.100

Esecuzione di Ping 192.168.50.100 con 32 byte di dati:
Risposta da 192.168.50.100: byte=32 durata=1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata=2ms TTL=64
Risposta da 192.168.50.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata=2ms TTL=64
Risposta da 192.168.50.100: byte=32 durata=1ms TTL=64
Risposta da 192.168.50.100: byte=32 durata=3ms TTL=64

Statistiche Ping per 192.168.50.100:
    Pacchetti: Trasmessi = 10, Ricevuti = 10,
    Persi = 0 (0% persi),
Tempo approssimativo percorsi andata/ritorno in millisecondi:
    Minimo = 0ms, Massimo = 3ms, Medio = 0ms

C:\Users\user>
```

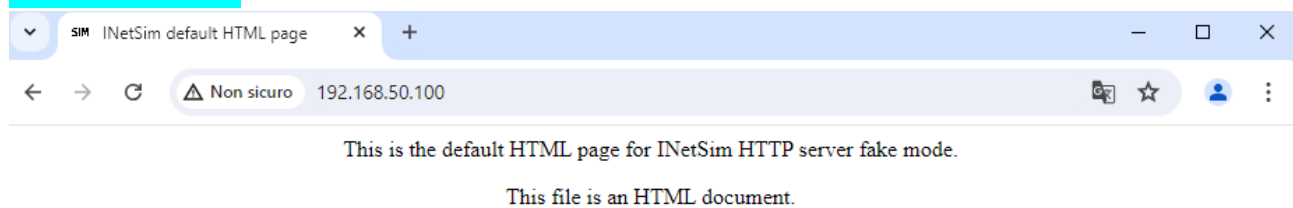
Nel primo passaggio è stata verificata la connettività di base tra il client Windows e la macchina Kali eseguendo un ping dall'host Windows verso l'indirizzo della Kali.

L'esecuzione del comando ha prodotto dieci richieste e dieci risposte, senza perdita di pacchetti.

Tali risultati confermano che la regola sul firewall è stata applicata correttamente e che gli endpoint sono raggiungibili sulla rete di laboratorio.

Questa verifica costituisce la premessa operativa necessaria prima di procedere con la simulazione dei servizi e con la cattura del traffico.

## PASSAGGIO 2



In questa fase è stata testata la risposta del server simulato avviato su Kali. Dal browser della macchina Windows è stato raggiunto l'indirizzo della Kali, ottenendo come risultato la pagina predefinita generata da INetSim. La comparsa di questa schermata ha confermato che la simulazione dei servizi web è attiva e che la comunicazione tra i due sistemi avviene correttamente, senza interferenze da parte del firewall o problemi di rete. Questo passaggio ha quindi dimostrato in modo pratico che l'ambiente di laboratorio è operativo e che i servizi fittizi rispondono come previsto.



## PASSAGGIO 3

Wireshark interface showing a capture on interface \*eth0. The packet list displays 11 packets. The first 10 are ICMP Echo (ping) requests from 192.168.50.102 to 192.168.50.100. The 11th packet is an ARP request from PCSSystemtec\_4e:73:95:00:00:00 to PCSSystemtec\_42:b7:9c:08:00:27.

No.	Time	Source	Destination	Protocol	Length
1	0.000000000	192.168.50.102	192.168.50.100	ICMP	74
2	0.000035101	192.168.50.100	192.168.50.102	ICMP	74
3	1.019470233	192.168.50.102	192.168.50.100	ICMP	74
4	1.019503992	192.168.50.100	192.168.50.102	ICMP	74
5	2.053228163	192.168.50.102	192.168.50.100	ICMP	74
6	2.053462347	192.168.50.100	192.168.50.102	ICMP	74
7	3.081431710	192.168.50.102	192.168.50.100	ICMP	74
8	3.081456639	192.168.50.100	192.168.50.102	ICMP	74
9	4.104117185	192.168.50.102	192.168.50.100	ICMP	74
10	4.104144478	192.168.50.100	192.168.50.102	ICMP	74
11	4.837575275	PCSSystemtec_4e:73:...	PCSSystemtec_42:b7:...	ARP	60

Frame 1: 74 bytes on wire (592 bits), 74 captured (584) bytes (4672 bits) on interface eth0  
Ethernet II, Src: PCSSystemtec\_4e:73:95:00:00:00, Dst: 08:00:27:42:b7:9c  
Internet Protocol Version 4, Src: 192.168.50.102, Dst: 192.168.50.100  
Internet Control Message Protocol

Wireshark interface showing a capture on interface \*eth0. The packet list displays 13 packets. The first 12 are HTTP traffic (GET requests and responses) between 192.168.50.102 and 192.168.50.100. The 13th packet is an ARP broadcast from PCSSystemtec\_4e:73:95:00:00:00.

No.	Time	Source	Destination	Protocol	Length
103	113.378960105	192.168.50.100	192.168.50.102	TCP	60
104	113.379674744	192.168.50.102	192.168.50.100	TCP	60
105	113.379674847	192.168.50.102	192.168.50.100	TCP	60
106	113.401468341	192.168.50.102	192.168.50.100	HTTP	504
107	113.401502317	192.168.50.100	192.168.50.102	TCP	54
108	113.427944200	192.168.50.100	192.168.50.102	TCP	204
109	113.433953544	192.168.50.100	192.168.50.102	HTTP	312
110	113.437028536	192.168.50.102	192.168.50.100	TCP	60
111	113.471509123	192.168.50.102	192.168.50.100	TCP	60
112	113.471540407	192.168.50.100	192.168.50.102	TCP	54
113	113.865282734	PCSSystemtec_4e:73:...	Broadcast	ARP	60

Frame 109: 312 bytes on wire (2496 bits), 312 captured (2500) bytes (20000 bits) on interface eth0  
Ethernet II, Src: PCSSystemtec\_42:b7:9c:08:00:27, Dst: 08:00:27:42:b7:9c  
Internet Protocol Version 4, Src: 192.168.50.100, Dst: 192.168.50.102  
Transmission Control Protocol, Src Port: 54444, Dst Port: 80  
[2 Reassembled TCP Segments (408 bytes)]  
Hypertext Transfer Protocol  
Line-based text data: text/html (10 lines)

Frame (312 bytes)    Reassembled TCP (408 bytes)

In questa fase è stata effettuata la cattura del traffico di rete per osservare in modo diretto le comunicazioni tra la macchina Windows e la macchina Kali Linux.

Dopo aver avviato Wireshark sull'interfaccia di rete corretta, sono state generate due tipologie di traffico: la prima attraverso una semplice sequenza di ping, utile per verificare la raggiungibilità del server e l'invio di pacchetti ICMP, e la seconda tramite l'accesso dal browser al servizio HTTP simulato da INetSim.

La cattura dei pacchetti ha evidenziato correttamente lo scambio di messaggi ICMP tra le due macchine, confermando la stabilità della connessione e l'assenza di perdita di pacchetti.

Successivamente filtrando il traffico HTTP, sono stati individuati i pacchetti relativi alla richiesta GET proveniente dal client Windows e alla risposta fornita dal server Kali che conteneva la pagina di default del servizio simulato.

Questa analisi ha permesso di visualizzare concretamente la sequenza di comunicazione e di comprendere come il traffico si articoli nei diversi protocolli, dimostrando la piena riuscita della simulazione e la corretta attività di sniffing.