

PROGETTO FINALE

MODULO 1

1) REQUISITI E SERVIZI:

#KALI LINUX IP: 192.168.32.100

#WINDOWS 7 IP: 192.168.32.101

HTTPS SERVER: attivo

Servizio DNS per risoluzione dominio: attivo

Simulare, in ambiente di laboratorio virtuale, un'architettura client server in cui un client con indirizzo 192.168.32.101 (Windows 7) richiede tramite web browser una risorsa all'hostname epicode.internal che risponde all'indirizzo 192.168.32.100 (Kali). Si intercetti poi la comunicazione con Wireshark, evidenziando i MAC address di sorgente e destinazione ed il contenuto della richiesta HTTPS. Ripetere l'esercizio, sostituendo il server HTTPS, con un server HTTP. Si intercetti nuovamente il traffico, evidenziando le eventuali differenze tra il traffico appena catturato in HTTP ed il traffico precedente in HTTPS. Spiegare, motivandole, le principali differenze se presenti.

INIZIO IMPOSTANDO I NUOVI IP SU VM KALI E VM WINDOWS 7

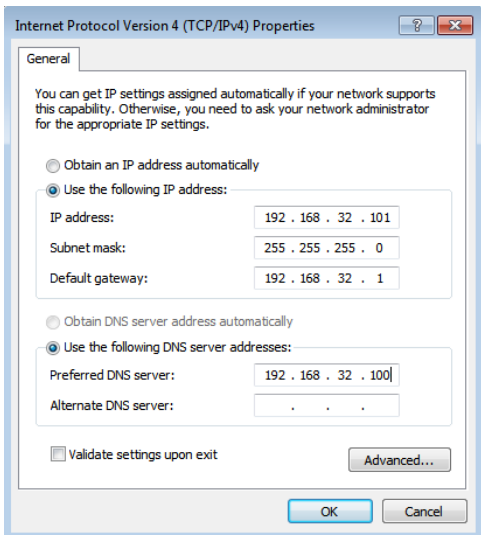
KALI -> TERMINALE -> SUDO NANO /ETC/NETWORK/INTERFACES:

```
django@kali: ~  
File Actions Edit View Help  
GNU nano 7.2 /etc/network/interfaces *  
# This file describes the network interfaces available on your system  
# and how to activate them. For more information, see interfaces(5).  
  
source /etc/network/interfaces.d/*  
  
# The loopback network interface  
auto lo  
iface lo inet loopback  
  
auto eth0  
iface eth0 inet static  
address 192.168.32.100  
netmask 255.255.255.0  
network 192.168.32.0  
broadcast 192.168.32.255  
gateway 192.168.32.1
```

VERIFICO CON IFCONFIG:

```
(django@kali)~  
$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 192.168.32.100 netmask 255.255.255.0 broadcast 192.168.32.255
```

WINDOWS7 -> CONFIGURAZIONE IP & DNS



WINDOWS 7 -> PANNELLO DI CONTROLLO -> IMPOSTAZIONI DI RETE. WINDOWS 7 DOVRÀ RICHIEDERE UN SERVIZIO WEB AL DNS DI KALI CHE CORRISPONDERÀ AL SUO IP. IL RECORD CHE VI VERRÀ ASSOCIATO SARÀ epicode.internal (hostname).

VERIFICO CON IPCONFIG /ALL:

```
Windows 7 x64 [In esecuzione] - Oracle VM VirtualBox
File Macchina Visualizza Inserimento Dispositivi Aiuto

C:\Users\Win7>ipconfig /all

Windows IP Configuration

Host Name . . . . . : Win7-PC
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . : 
   Description . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
   Physical Address. . . . . : 08-00-27-D3-39-4A
   DHCIPv6 Enabled. . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::206c::467d::164::%11(Preferred)
   IPv4 Address. . . . . : 192.168.32.101(Preferred)
   Subnet Mask . . . . . : 255.255.255.0
   Default Gateway . . . . . : 192.168.32.1
   DHCPv6 IAID . . . . . : 235405351
   DHCPv6 Client DUID. . . . . : 00-01-00-01-2C-CF-ED-A1-08-00-27-D3-39-4A

   DNS Servers . . . . . : 192.168.32.100
   NetBIOS over Tcpip. . . . . : Enabled
```

PING A KALI:

```
C:\Users\Win7>ping 192.168.32.100

Pinging 192.168.32.100 with 32 bytes of data:
Reply from 192.168.32.100: bytes=32 time<1ms TTL=64
Reply from 192.168.32.100: bytes=32 time=1ms TTL=64
Reply from 192.168.32.100: bytes=32 time=1ms TTL=64
Reply from 192.168.32.100: bytes=32 time=1ms TTL=64
```

#INETSIM

InetSim È UN SOFTWARE CHE SIMULA LE FEATURES DELLA RETE/SERVIZI INTERNET, DA CONFIGURARE ALL'INTERNO DELLA VM DI KALI LINUX. ALL'INTERNO DI "NANO" (EDITOR DI TESTO CHE CONSENTE DI MODIFICARE FILE) COMMENTEREMO CON "#" DAVANTI I SERVIZI CHE NON UTILizzerEMO LASCIANDO INTATTI HTTP/HTTPS/DNS. ACCEDEREMO SUCCESSIVAMENTE DAL BROWSER DI WINDOWS AD UN SITO FAKE RICHIEDENDO UNA RISORSA WEB SULL'HOST epicode.internal, CON CONNESSIONE INTERNET PROTETTA (HTTPS) E NON PROTETTA (HTTP).

PASSAGGI DI CONFIGURAZIONE DI INETSIM:

- cd /etc/inetsim (dir)
- ls (file presenti nella dir)
- sudo nano inetsim.conf
- # per commentare i servizi da "spegnere" (tranne http/https)
- nella riga "service_bind_address" inseriamo il local host e sovrascriviamo con ^O
- sudo inetsim (per avviare la simulazione)

```
#####
# start_service
#
# The services to start
#
# Syntax: start_service <service name>
#
# Default: none
#
# 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
```

IP DI ACCESSO AL SITO INTERNET SIMULATO (0.0.0.0 PER TUTTE LE INTERFACCIE):

```
#####  
# service_bind_address  
#  
# IP address to bind services to  
#  
# Syntax: service_bind_address <IP address>  
#  
# Default: 0.0.0.0  
#  
service_bind_address 0.0.0.0
```

DNS DI DEFAULT E DNS STATICO CON RECORD DA ASSOCIARE:

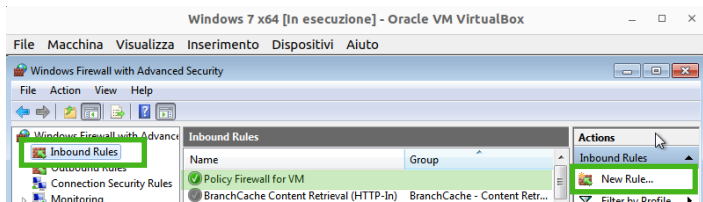
```
#####  
# dns_default_ip  
#  
# Default IP address to return with DNS replies  
#  
# Syntax: dns_default_ip <IP address>  
#  
# Default: 192.168.32.100  
#  
dns_default_ip 192.168.32.100
```

```
#####  
# dns_static  
#  
# Static mappings for DNS  
#  
# Syntax: dns_static <fqdn hostname> <IP address>  
#  
# Default: none  
#  
dns_static epicode.internal 192.168.32.100
```

SIMULAZIONE AVVIATA CORRETTAMENTE:

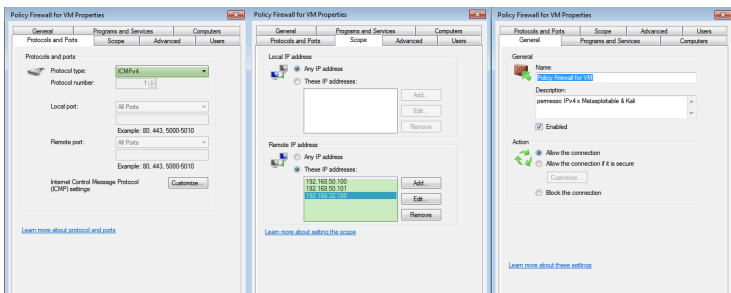
```
(django@kali)-[~]  
$ sudo inetsim  
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg  
Using log directory: /var/log/inetsim/  
Using data directory: /var/lib/inetsim/  
Using report directory: /var/log/inetsim/report/  
Using configuration file: /etc/inetsim/inetsim.conf  
Parsing configuration file.  
Configuration file parsed successfully.  
== INetSim main process started (PID 19514) ==  
Session ID: 19514  
Listening on: 0.0.0.0  
Real Date/Time: 2023-11-20 10:12:13  
Fake Date/Time: 2023-11-20 10:12:13 (Delta: 0 seconds)  
Forking services...  
 * dns_53_tcp_udp - started (PID 19524)  
print() on closed filehandle MLOG at /usr/share/perl5/Net/DNS/Nameserver.pm line 399.  
 * https_443_tcp - started (PID 19526)  
 * http_80_tcp - started (PID 19525)  
done.  
Simulation running.
```

SETTAGGIO POLICY FIREWALL WINDOWS 7:

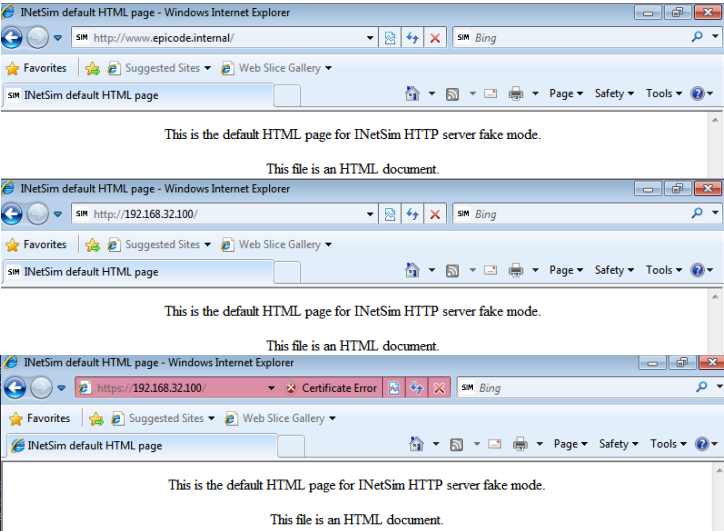


IMPOSTAZIONI AVANZATE RETE -> INBOUND RULES -> NEW
#PROTOCOLS AND PORTS SETTANDO IL CAMPO "PROTOCOL TYPE" SU ICMPv4

#SCOPE SETTANDO GLI IP A CUI CONCEDIAMO DI INVIARE PING
#GENERAL (FACOLTATIVO) INSERENDO UN RIFERIMENTO AL PROTOCOLLO PER DISTINGUERLO DA ALTRI

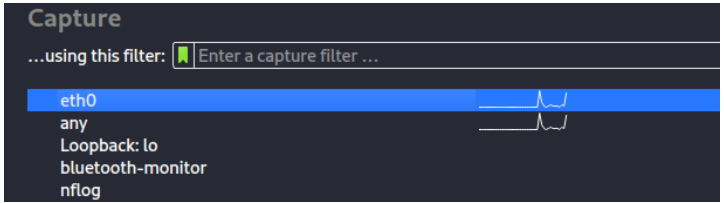


RAGGIUNGIBILITÀ DEL SERVER SU HOSTNAME/IP (HTTP/HTTPS):

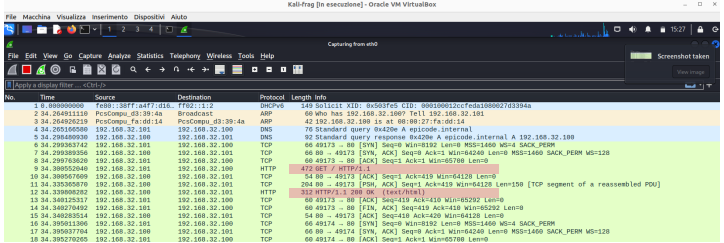


#WIRESHARK

SI UTILIZZA PER SNIFFARE, OVVERO CATTURARE ED ANALIZZARE I PACCHETTI IN TRANSITO SU UNA DETERMINATA RETE. ATTRAVERSO LA SELEZIONE DI UNA SCHEDA DI RETE NE VIENE CATTURATO IL TRAFFICO CHE SI PUÒ SUCESSIVAMENTE FILTRARE PER VARIE NECESSITÀ/INTERESSE (IP / DNS / PORTA, ECC). eth0 - stessa rete (LA NOSTRA VM)



HTTP: PORTA 80 - CONNESSIONE NON PROTETTA

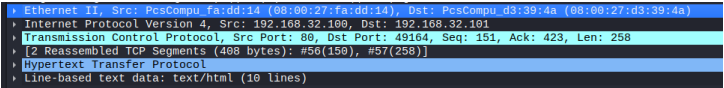


PASSAGGIO 2/3: PROTOCOLLO ARP (LVL2) - LE MACCHINE COMUNICANO TRAMITE L'INDIRIZZO FISICO, SI RICERCA LA CORRISPONDENZA TRA IP E MAC ADDRESS, KALI RISPONDE TROVANDOLO.

PASSAGGIO 3/4: DNS - CONFERMATA LA CORRISPONDENZA DELL'IP DI KALI AL RECORD epicode.internal con la query

SUCESSIVAMENTE AVVIENE LA RICHIESTA SULLA PORTA 80 (HTTP) DA "GET" A "OK 200" DEL SERVER CON SEQUENZA DI THREE-WAY-HANDSHAKE. STRINGHE DA SYN/SYN ACK fino a FIN/ACK ACK.

PACCHETTO IN TRANSITO "TEXT/PLAIN".



MAC ADDRESS SORGENTE (WINDOWS 7):
08:00:27:D3:39:4A

MAC ADDRESS DESTINAZIONE (KALI):
08:00:27:fa:dd:14

HTTPS: PORTA 443 - CONNESSIONE PROTETTA

RICHIESTA SULLA PORTA 443 (HTTPS) DA “Client Hello” A “Server Hello” (su TLS - PROTOCOLLO CHE SERVE A PROTEGGERE DATI) DOPO LA SEQUENZA DI THREE-WAY-HANDSHAKE. HTTPS CRIPTA I DATI DURANTE LA COMUNICAZIONE CLIENT/SERVER.

1 0.000000000	PcsCompu_d3:39:4a	Broadcast	ARP	60 Who has 192.168.32.100? Tell 192.168.32.101
2 0.000014166	PcsCompu_fa:dd:14	PcsCompu_d3:39:4a	ARP	42 192.168.32.100 is at 08:00:27:fa:dd:14
3 0.000237618	192.168.32.101	192.168.32.100	DNS	76 Standard query 0xb4fd A epicode.internal
4 0.038546242	192.168.32.100	192.168.32.101	DNS	92 Standard query response 0xb4fd A epicode.internal A 192.168.32.100
5 0.039378704	192.168.32.101	192.168.32.100	TCP	66 49165 -> 443 [SYN] Seq=0 Win=0 Len=0 MSS=1460 WS=4 SACK_PERM
6 0.039401951	192.168.32.100	192.168.32.101	TCP	66 443 -> 49165 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_P
7 0.039614128	192.168.32.101	192.168.32.100	TCP	60 49165 -> 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
8 0.044513958	192.168.32.101	192.168.32.100	TLSv1	183 Client Hello
9 0.044534448	192.168.32.100	192.168.32.101	TCP	54 443 -> 49165 [ACK] Seq=1 Ack=130 Win=64128 Len=0
10 0.171339163	192.168.32.100	192.168.32.101	TLSv1	1373 Server Hello, Certificate, Server Key Exchange, Server Hello Done
11 0.190778706	192.168.32.101	192.168.32.100	TLSv1	188 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Messa
12 0.190825065	192.168.32.100	192.168.32.101	TCP	54 443 -> 49165 [ACK] Seq=1320 Ack=264 Win=64128 Len=0
13 0.191057550	192.168.32.100	192.168.32.101	TLSv1	119 Change Cipher Spec, Encrypted Handshake Message
14 0.294006202	PcsCompu_d3:39:4a	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101
15 0.397024202	192.168.32.100	192.168.32.101	TCP	113 [TCP Retransmission] 443 -> 49165 [PSH, ACK] Seq=1320 Ack=264 Win=0
16 0.397482599	192.168.32.101	192.168.32.100	TCP	60 49165 -> 443 [ACK] Seq=264 Ack=1379 Win=64320 Len=0
17 0.398539317	192.168.32.101	192.168.32.100	TCP	66 [TCP Dup ACK 16w] 49165 -> 443 [ACK] Seq=264 Ack=1379 Win=64320 Le
18 1.225785408	PcsCompu_d3:39:4a	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101
19 2.226495621	PcsCompu_d3:39:4a	Broadcast	ARP	60 Who has 192.168.32.1? Tell 192.168.32.101

#DIFFERENZE HTTP/HTTPS

HTTP	HTTPS
ATTUA UNO SCAMBIO 3 WAY HANDSHAKE E I PACCHETTI IN TRANSITO SONO LEGGIBILI, FORNISCE PIÙ INFO/ELEMENTI DURANTE LA CATTURA CON WIRESHARK	SCAMBIO 3 WAY HANDSHAKE E POI CRIPTA CON TLS. ATTUA UNA CIFRATURA DEL CANALE IN 3 FASI: NEGOZIAZIONE DELL’ALGORITMO DA USARE TRA LE PARTI COINVOLTE, SCAMBIO DELLE CHIAVI DI AUTENTICAZIONE CON CIFRATURA ASIMMETRICA, CIFRATURA SIMMETRICA
CONTENUTO INTERCETTABILE	CONTENUTO NON INTERCETTABILE
DA “GET” a “ 200 OK” su porta 80	DA “Client Hello” a “ Server Hello” su porta 443

ESEMPIO HTTP: SAPPIAMO COSA CONTIENE IL PACCHETTO

Frame 33: 312 bytes on wire (2496 bits), 312 bytes captured (2496 bits) on interface eth0, id 0
Ethernet II, Src: PcsCompu_fa:dd:14 (08:00:27:fa:dd:14), Dst: PcsCompu_d3:39:4a (08:00:27:d3:39:4a)
Internet Protocol Version 4, Src: 192.168.32.100, Dst: 192.168.32.101
Transmission Control Protocol, Src Port: 80, Dst Port: 49165, Seq: 151, Ack: 419, Len: 258
[2 Reassembled TCP Segments (408 bytes): #32(150), #33(258)]
Hypertext Transfer Protocol
HTTP/1.1 200 OK\r\n
[Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
Response Version: HTTP/1.1
Status Code: 200
[Status Code Description: OK]
Response Phrase: OK
Date: Mon, 20 Nov 2023 16:50:56 GMT\r\n
Content-Type: text/html\r\n
Server: INetSim HTTP Server\r\n
Connection: Close\r\n
Content-Length: 258\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.033215699 seconds]
[Request in frame: 30]
[Request URI: http://epicode.internal/]
File Data: 258 bytes
Line-based text data: text/html (10 lines)
<html>\r\n
<head>\r\n
<title>INetSim default HTML page</title>\r\n
</head>\r\n
<body>\r\n
<p>\r\n
<p align="center">This is the default HTML page for INetSim HTTP server fake mode.</p>\r\n
<p align="center">This file is an HTML document.</p>\r\n
</body>\r\n
</html>\r\n