

SIMULAZIONE DI UN'ARCHITETTURA CLIENT SERVER

Nell'esercitazione di oggi abbiamo simulato, in un laboratorio virtuale, un'architettura client server tra due dispositivi tramite web browser.

FASE 1

Nella Fase 1 abbiamo dovuto configurare i dispositivi per poterli far comunicare. Una macchina in ambiente windows con IP ADDRESS **192.168.32.101** e l'altra in ambiente Linux con IP ADDRESS **192.168.32.100**. Una volta impostati tutti i criteri base abbiamo iniziato a configurare il tool **inetsim** presente sulla macchina in ambiente Linux.

Tramite il comando **/etc/inetsim/inetsim.conf** abbiamo potuto cambiare i parametri (fig.1) del tool per avviare poi il collegamento tramite web browser

```
#####  
# service_bind_address  
#  
# IP address to bind services to  
#  
# Syntax: service_bind_address <IP address>  
#  
# Default: 127.0.0.1  
#  
#service_bind_address 10.10.10.1  
service_bind_address 192.168.32.100  
  
# dns_static  
#  
# Static mappings for DNS  
#  
# Syntax: dns_static <fqdn hostname> <IP address>  
#  
# Default: none  
#  
#dns_static www.foo.com 10.10.10.10  
#dns_static ns1.foo.com 10.70.50.30  
#dns_static ftp.bar.net 10.10.20.30  
dns_static epicode.internal 192.168.32.100
```

FIG.1

Possiamo notare il cambio di IP ADDRESS e dei DNS. Abbiamo anche specificato il nome della pagina web che dovevamo cercare per riuscire a comunicare. In questo caso **epicode.internal**. Questo ci ha permesso di far cominucare il dispositivo in ambiente Windows tramite web browser con il dispositivo in ambiente Linux.

Per fare questo però abbiamo dovuto apportare delle modifiche alla scheda di rete del dispositivo Windows (fig.2)

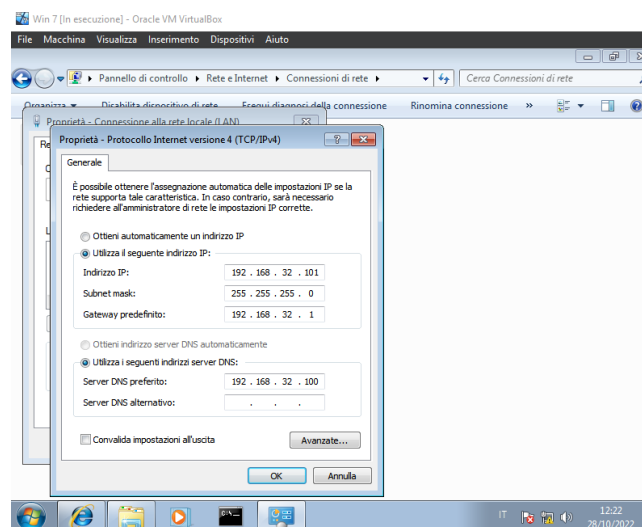


FIG.2

Inserendo i DNS abbiamo potuto comunicare

FASE 2

Nella Fase 2 abbiamo utilizzato, nel dispositivo in ambiente Window, il web browser presente di default per accedere alla pagina web presente invece sul dispositivo Linux.

Inserendo nella barra di ricerca internal.epicode siamo riusciti a comunicare tra i due dispositivi(fig.3)

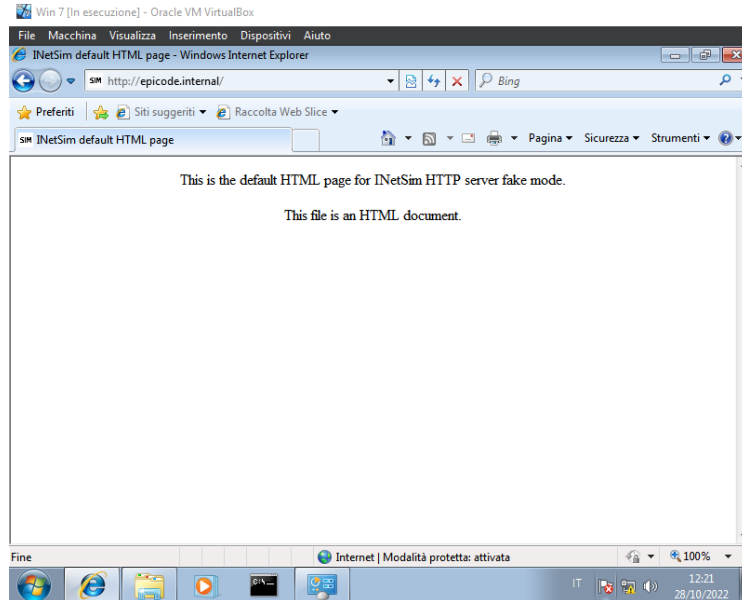


FIG.3

FASE 3

Nella Fase 3 abbiamo usato il tool presente sul dispositivo Linux, Wireshark che ci ha permesso di vedere ed analizzare il “traffico di dati” tra i due dispositivi. Un traffico era con protocollo **HTTP**(fig.4) e l'altro in protocollo **HTTPS**(fig.5)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.101	192.168.32.100	TCP	66	49165 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
2	0.000018128	192.168.32.100	192.168.32.101	TCP	66	80 → 49165 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=4
3	0.000293463	192.168.32.101	192.168.32.100	TCP	60	49165 → 80 [ACK] Seq=1 Ack=1 Win=65700 Len=0
4	0.000350707	192.168.32.101	192.168.32.100	HTTP	340	GET / HTTP/1.1
5	0.000355986	192.168.32.100	192.168.32.101	TCP	54	80 → 49165 [ACK] Seq=1 Ack=287 Win=64128 Len=0
6	0.015923110	192.168.32.100	192.168.32.101	TCP	284	80 → 49165 [PSH, ACK] Seq=1 Ack=287 Win=64128 Len=150 [TCP segment of a r...
7	0.018168651	192.168.32.100	192.168.32.101	HTTP	312	HTTP/1.1 200 OK (text/html)
8	0.018418054	192.168.32.101	192.168.32.100	TCP	60	49165 → 80 [ACK] Seq=287 Ack=410 Win=65292 Len=0
9	0.018478115	192.168.32.101	192.168.32.100	TCP	60	49165 → 80 [FIN, ACK] Seq=287 Ack=410 Win=65292 Len=0
10	0.018485538	192.168.32.100	192.168.32.101	TCP	54	80 → 49165 [ACK] Seq=410 Ack=288 Win=64128 Len=0

FIG.4

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.32.101	192.168.32.100	TCP	66	49197 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
2	0.000017013	192.168.32.100	192.168.32.101	TCP	66	443 → 49197 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=4
3	0.000267965	192.168.32.101	192.168.32.100	TCP	60	49197 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
4	0.000509824	192.168.32.101	192.168.32.100	TLSv1	215	Client Hello
5	0.000516166	192.168.32.100	192.168.32.101	TCP	54	443 → 49197 [ACK] Seq=1 Ack=162 Win=64128 Len=0
6	0.012902822	192.168.32.100	192.168.32.101	TLSv1	1373	Server Hello, Certificate, Server Key Exchange, Server Hello Done
7	0.018765363	192.168.32.101	192.168.32.100	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
8	0.018786856	192.168.32.100	192.168.32.101	TCP	54	443 → 49197 [ACK] Seq=1320 Ack=296 Win=64128 Len=0
9	0.019448851	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message

FIG.5

La differenza tra i due sono i protocolli **TCP**, per http e il protocollo aggiuntivo **TLS** per HTTPS.

Piu nello specifico andando a selezionare una voce possiamo notare i mac address dei dispositivi che hanno comunicato(fig. 5 6)

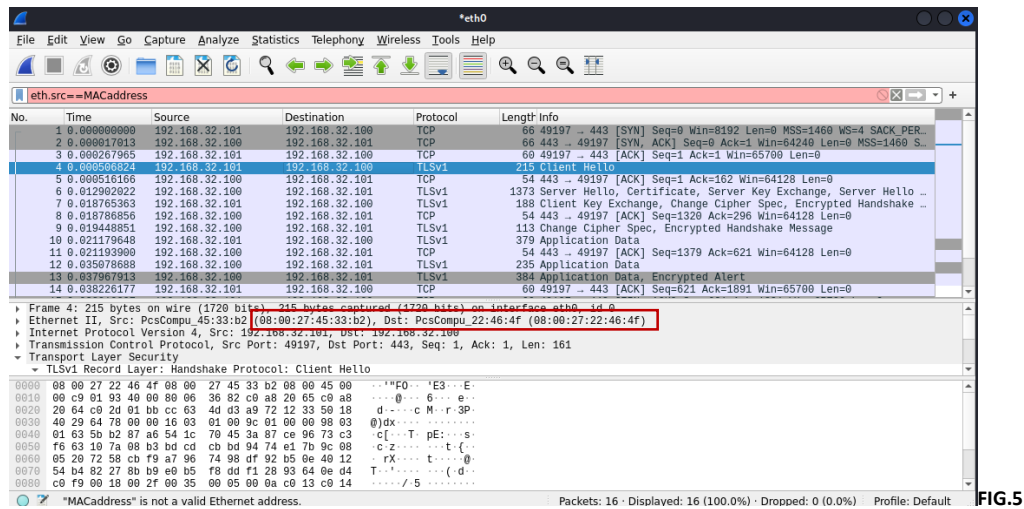


FIG.5

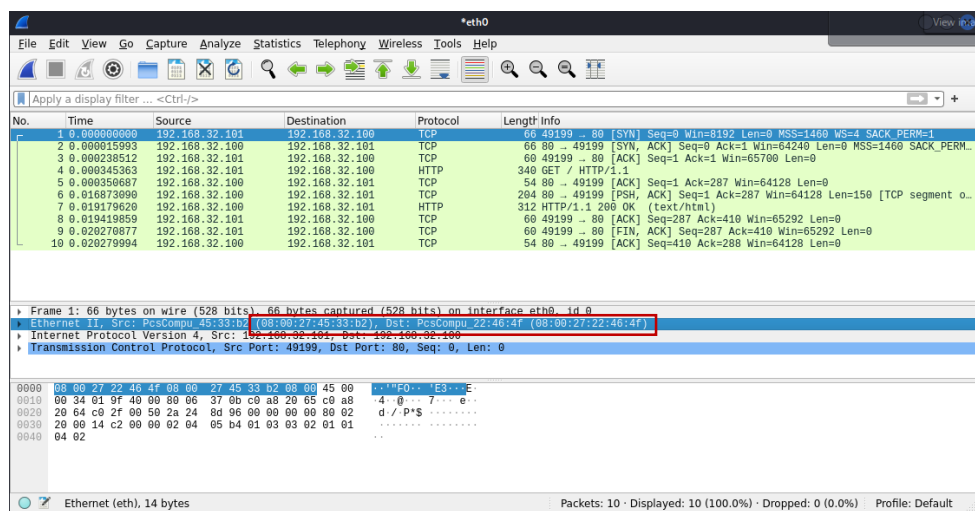


FIG.6

CONCLUSIONI

Oltre alla comunicazione dei due dispositivi possiamo notare un protocollo aggiuntivo su HTTPS – TLS Transport Layer Security (fig.7).



