

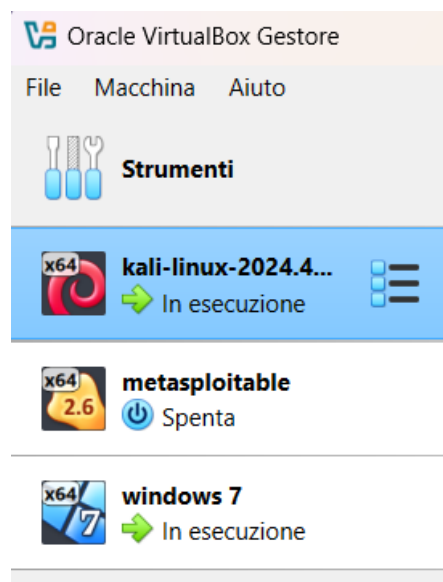
Traccia

Simulare, in un ambiente virtuale, un'architettura client server in cui un client con indirizzo (192.168.32.101) Windows 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.

Svolgimento passo dopo passo

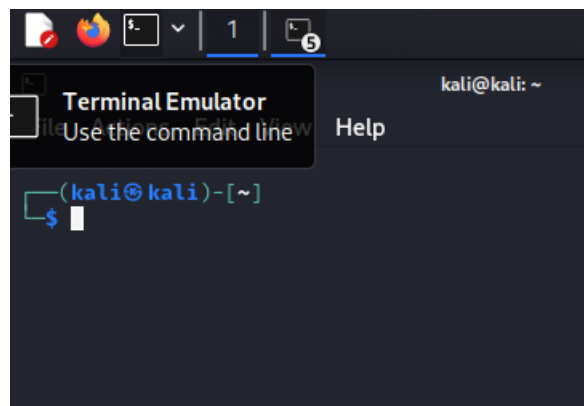
Step1

Aprire da “**Oracle Virtual Box**” le macchine virtuali di cui abbiamo bisogno in questo caso, in questo caso avremo bisogno di “**Windows**” e “**Kali Linux**”



Una volta aperte entrambe le macchine vado a settare gli “**IP**” di entrambe, windows (192.168.32.101) e Kali Linux (192.168.32.101)

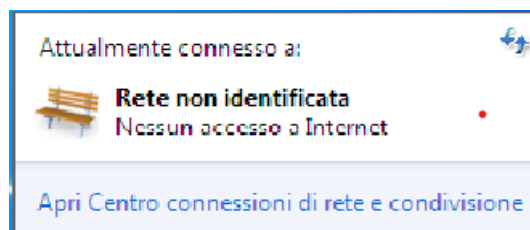
Per fare questo su **“Kali Linux”** vado ad aprire il **“prompt”**



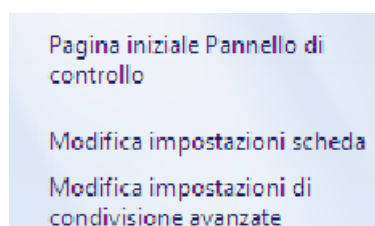
Dopodichèandrò a dare il comando **“sudo nano /etc/network/interfaces”**
in questo modo



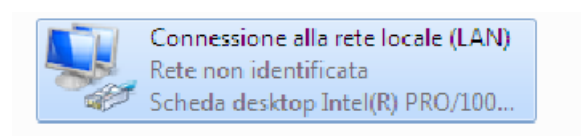
Per windows la procedura è differente, perchè una volta aperto la VM di
Windows vado su **“Apri Centro connessioni di rete”**



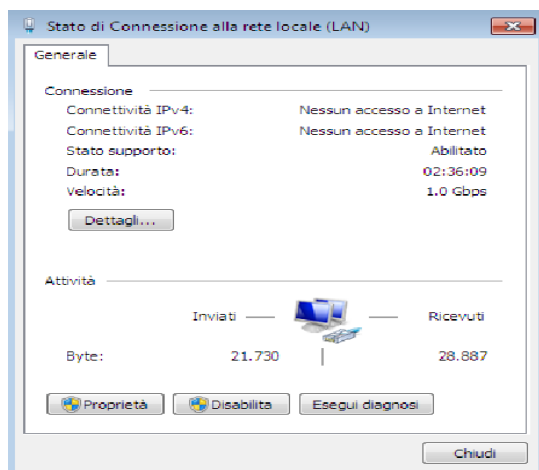
Una volta aperto vado a cliccare su **“Modifica impostazioni scheda”**



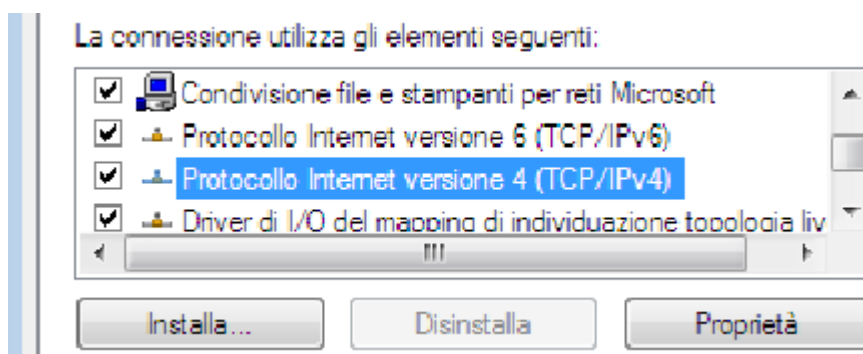
Dopodichè doppio click su **“Connessione alla rete locale”**



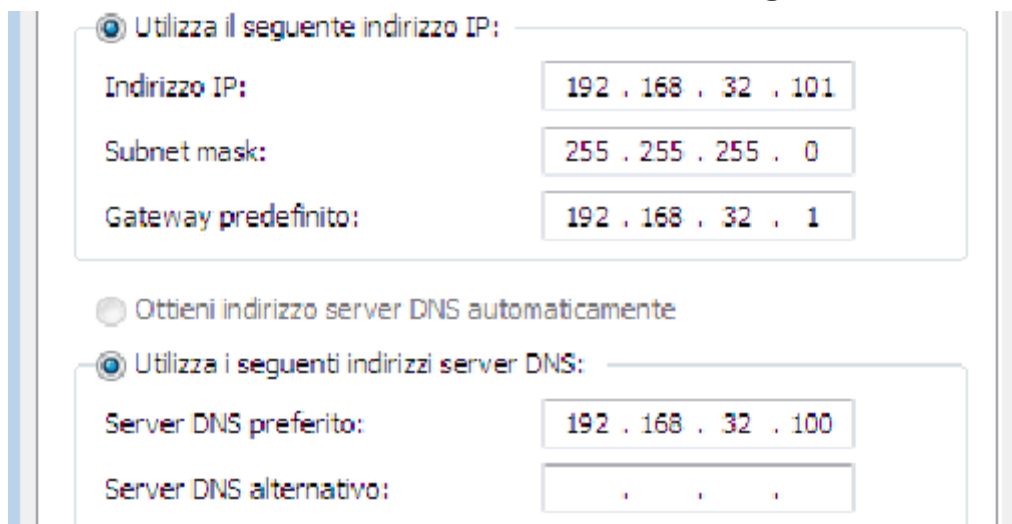
Una volta aperto vado su **“Proprietà”**



Aperta la finestra di **“Proprietà”** clicco su **“Protocollo Internet versione 4”**



E vado a settare la macchina come nell'immagine sottostante



Dopo settate entrambe le macchine controllo che entrambe comunichino tra di loro, per fare ciò riapro il **“Terminale”** di **“Kali Linux”** e vado a dare il comando **“ ping 192.168.32.101 ”** fatto questo ultimo passaggio il terminale mi mostrerà che la macchina di **“Kali”** pingherà con quella di windows

```
(kali㉿kali)-[~]
└─$ ping 192.168.32.101
PING 192.168.32.101 (192.168.32.101) 56(84) bytes of data.
64 bytes from 192.168.32.101: icmp_seq=1 ttl=128 time=7.42 ms
64 bytes from 192.168.32.101: icmp_seq=2 ttl=128 time=1.35 ms
64 bytes from 192.168.32.101: icmp_seq=3 ttl=128 time=2.43 ms
64 bytes from 192.168.32.101: icmp_seq=4 ttl=128 time=1.40 ms
^C
  _ 192.168.32.101 ping statistics _
```

Stesso procedimento sulla macchina “Windows”, apro il terminale e vado a adre lo stesso comando che ho dato a “Kali” quindi “ ping 192.168.32.100”

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

Esecuzione di Ping 192.168.32.100 con 32 byte di dati:
Risposta da 192.168.32.100: byte=32 durata=2ms TTL=64
Risposta da 192.168.32.100: byte=32 durata=1ms TTL=64
Risposta da 192.168.32.100: byte=32 durata=1ms TTL=64
Risposta da 192.168.32.100: byte=32 durata=1ms TTL=64

Statistiche Ping per 192.168.32.100:
    Pacchetti: Trasmessi = 4, Ricevuti = 4,
    Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
        Minimo = 1ms, Massimo = 2ms, Medio = 1ms
```

Step 2

Dopo aver appurato che entrambe le macchine comunicano tra di loro faccio sì che la macchina di “Kali” funga da “DNS” (Domain Name System) per fare ciò vado a riaprire di nuovo il terminale e vado a dargli il seguente comando “**dns --fakedomains epicode.internal --fakeip 192.168.32.100 --nameservers 192.168.32.100 --interface 192.168.32.100**”

```
(kali㉿kali)-[~]
$ dnschef --fakedomains epicode.internal --fakeip 192.168.32.100 --nameservers 192.168.32.100 --interface 192.168.32.100
```

Dato il comando e apparsa la schermata sottostante avremo il “DNS” attivo

```
(kali㉿kali)-[~]
$ dnschef --fakedomains epicode.internal --fakeip 192.168.32.100 --nameservers 192.168.32.100 --interface 192.168.32.100
/usr/bin/dnschef:453: SyntaxWarning: invalid escape sequence '\/'
header += " / _ | ' _ V _ | / _ | ' \ / _ _ | _ |\n"
/usr/bin/dnschef:454: SyntaxWarning: invalid escape sequence '\'
header += " | | | | | | | | | | | | | | | | | |\n"
/usr/bin/dnschef:455: SyntaxWarning: invalid escape sequence '\'
header += " \_,_| |_|_|_|_\|_|_|_|_|_|_|_|_|_|_|_|\n"

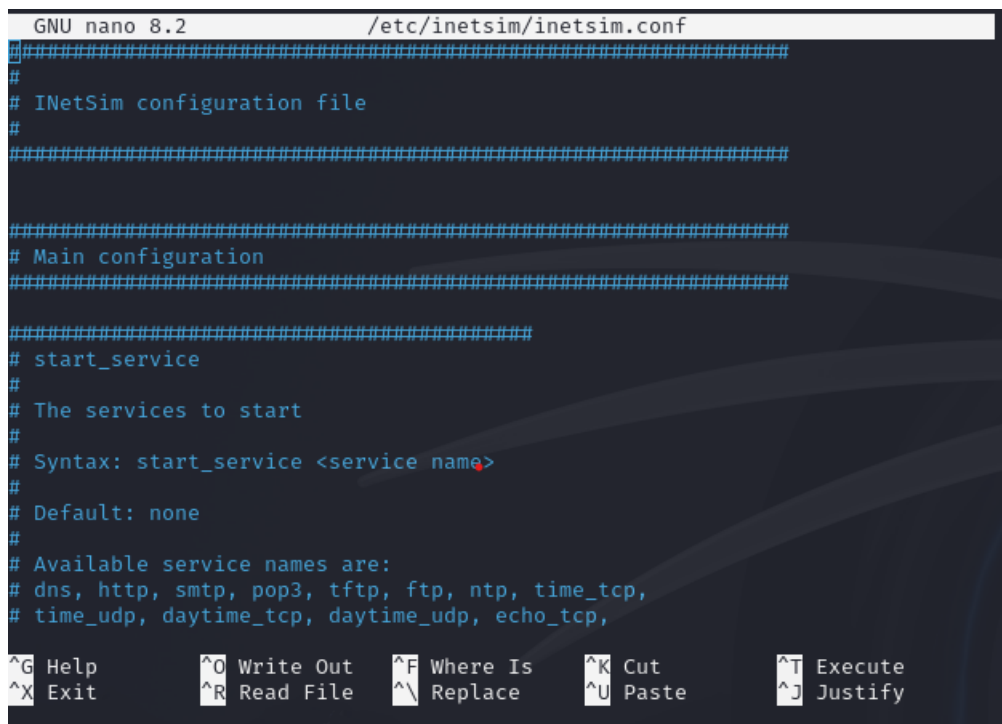
          version 0.4
      _____
     d n s c h e f
    iphelix@thesprawl.org

(09:24:27) [*] DNSChef started on interface: 192.168.32.100
(09:24:27) [*] Using the following nameservers: 192.168.32.100
(09:24:27) [*] Cooking A replies to point to 192.168.32.100 matching: epicode.internal
```

Step 3

Dopo aver completato i precedenti Step vado a settare, sempre dal terminale di “Kali”, l’**“Inetsim”** ossia un tool che mi permetterà di simulare servizi Internet standard come ad esempio **“DNS”** **“HTTP”** e **“HTTPS”**.

Per fare ciò vado a dare il comando **“sudo nano /etc/inetsim/inetsim.conf”**. Una volta dato il comando mi si è aperta l’interfaccia sottostante



```
GNU nano 8.2 /etc/inetsim/inetsim.conf
#####
#
# INetSim configuration file
#
#####
# Main configuration
#
#####
# 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,
^G Help      ^O Write Out  ^F Where Is  ^K Cut       ^T Execute
^X Exit      ^R Read File  ^N Replace   ^U Paste     ^J Justify
```

Appena aperta vado a cercare la sezione dove sono presenti tutti i protocolli e vado a inserire il simbolo del cancelletto “#” avanti a tutte le stringhe per farsì che **“inetsim”** legga solo **“start_service http”**

```
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
#start_service time_tcp
#start_service time_udp
#start_service daytime_tcp
#start_service daytime_udp
#start_service echo_tcp
#start_service echo_udp
#start_service discard_tcp
```

Un'altra modifica che vado ad apportare al terminale si trova poco sotto la serie di protocolli

```
# service_bind_address
#
# IP address to bind services to
#
# Syntax: service_bind_address <IP address>
#
# Default: 127.0.0.1
service_bind_address 192.168.32.100
#####
```

La modifica effettuata è stata eliminare il cancelletto in modo tale che “**inetsim**” eseguisse il comando “**service_bind_address**” e ho modificato l'ip successivo con “**192.168.32.100**”.

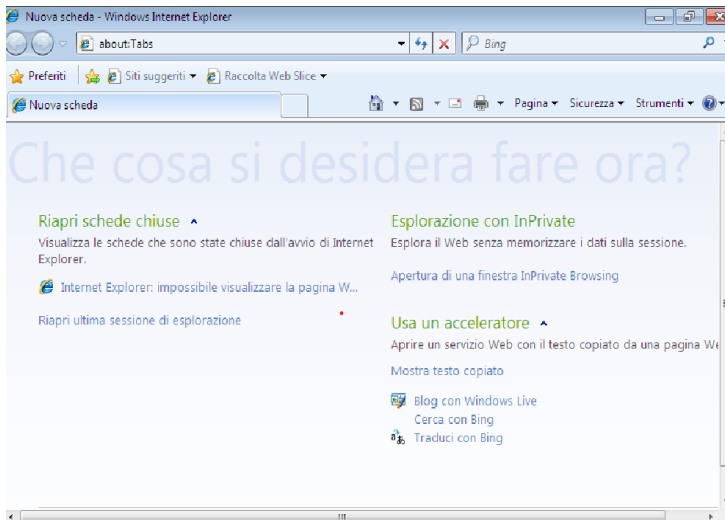
Una volta concluse le modifiche salviamo e ritrovati sul terminale andremo ad inserire il comando “**sudo inetsim**” e automaticamente ci ritroveremo un output come nell'immagine sottostante

```
(kali㉿kali)-[~]
$ sudo inetsim
[sudo] password for kali:
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 84341) ==
Session ID: 84341
Listening on: 192.168.32.100
Real Date/Time: 2025-03-22 12:24:23
Fake Date/Time: 2025-03-22 12:24:23 (Delta: 0 seconds)
Forking services...
* http_80_tcp - started (PID 84351)
done.
Simulation running.
```

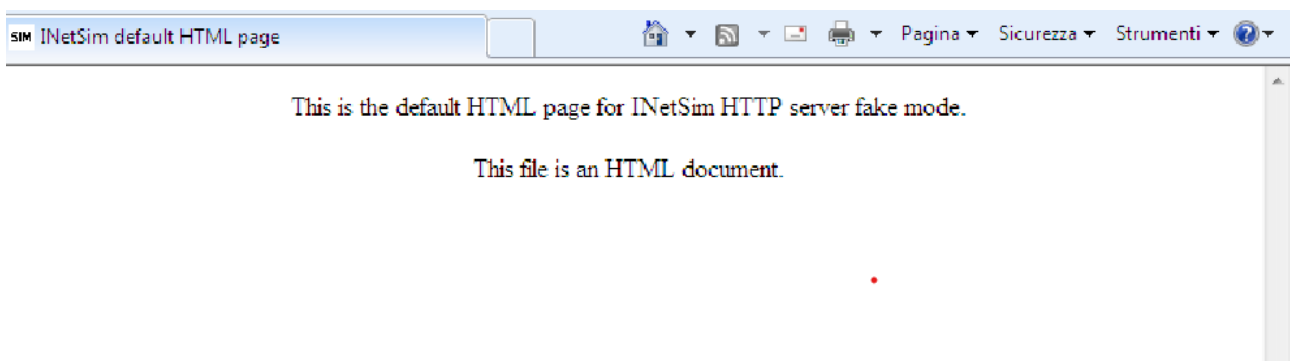
Dalla terzultima riga possiamo notare che il servizio “**HTTP**” è in ascolto sulla porta 80 del localhost.

Step 4

Una volta svolte tutte le modifiche e avviato il tool non mi resta che controllare che tutto funzioni, per fare ciò, vado sulla macchina virtuale di windows e apro il browser

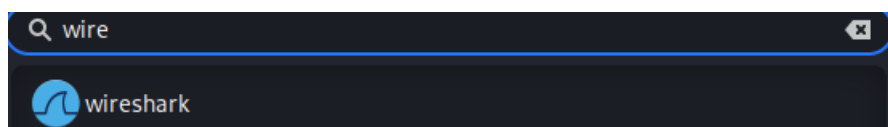


E nella barra di ricerca vado a scrivere “**epicode.internal**”, una volta aperto ritrovandomi nella pagina sottostante capisco che tutto sta funzionando correttamente.

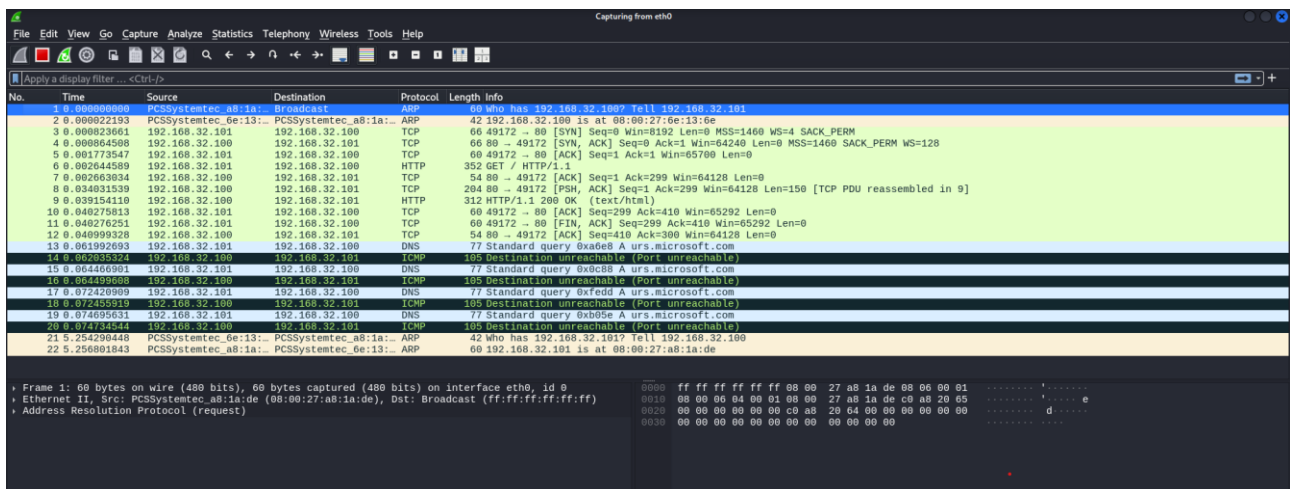


Step 5

Una volta accertatomi che tutto funziona vado ad aprire un tool presente su “Kali” ossia “**Wireshark**”



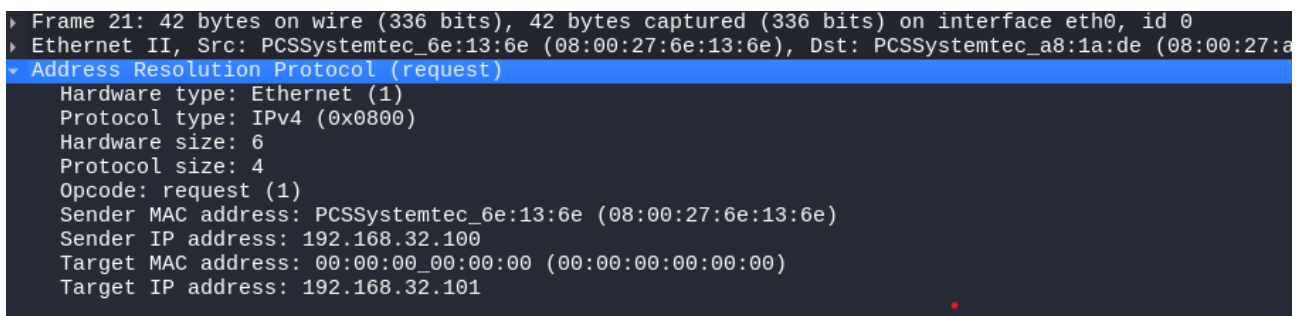
Aperto Wireshark vado a cliccare su “eth0” e mi ritrovo la schermata sottostante



Da questa vado a selezionare un protocollo “ARP”

21 5.254290448 PCSSystemtec 6e:13:... PCSSystemtec a8:1a:... ARP

E successivamente vado a cliccare su “Address Resolution Protocol”



E da qui posso vedere il “Sender MAC Address” e il “Target MAC Address”.

Step 6

Una volta visti i “MAC” vado a cambiare le impostazioni di “inetsim”, riaprendo il terminale di “Kali” andando nella lista dei protocolli e togliere il cancelletto a “HTTPS” e metterlo a “HTTP”

```
start_service http
#start_service https
```

```
#start_service http
start_service https
```


Fatto questo salviamo e andiamo di nuovo sul motore di ricerca per ricaricare la pagina “**epicode.internal**” ma stavolta con protocollo “**HTTPS**” una volta svolta, riapriamo Wireshark e possiamo vedere che

```
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: PCSSystemtec_a8:1a:de (08:00:27:a8:1a:de)
  Sender IP address: 192.168.32.101
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.32.100
```

I “**MAC**” address sono cambiati, questo perché il “**MAC**” target in “**HTTPS**” essendo criptato non ci è permesso guardarlo così come i pacchetti inviati.

*MAC = Indirizzo fisico unico che viene assegnato alle schede di rete dei computer

*HTTP = è un protocollo a livello applicativo usato come principale sistema per la trasmissione d'informazioni sul web

*HTTPS = è un protocollo per la comunicazione sicura attraverso una rete di computer utilizzato su internet

*Inetsim = è un simulatore di servizi internet

*wireshark = serve per la cattura dei pacchetti e l'analisi del contenuto

*ARP = si intende un protocollo di rete appartenente alla suite del protocollo internet (IP) e operante a livello accesso di rete

*DNS = indica un sistema per assegnare nomi ai nodi della rete (in inglese host)