

Configurazione IP statici di server/client

Procediamo alla modifica dell'IP statico di Kali con il comando SUDO NANO /ETC/NETWORK/INTERFACES. All'interno del testo inseriamo il nuovo IP di KALI 192.168.32.100

```
GNU nano 7.2
# 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/*

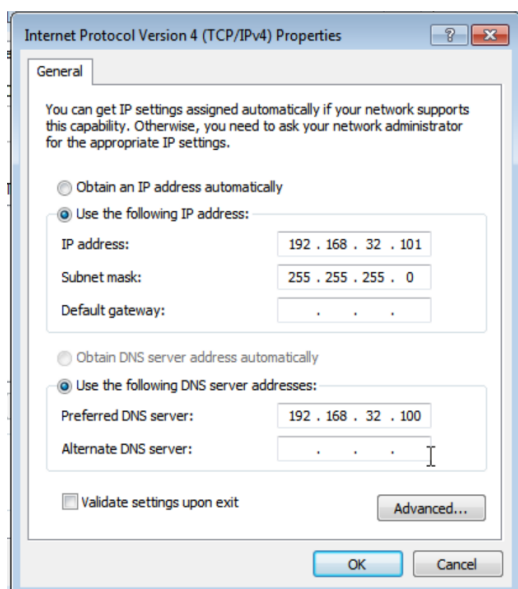
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.32.100
netmask 255.255.255.0
```

Eseguiamo la stessa operazione su WINDOWS7 ,seguendo il percorso :

Start/digitare nella barra di ricerca CONNESSIONI/selezionare Visualizza connessioni di rete/seleziona con il tasto destro del mouse sulla connessione LAN/seleziona la voce IPv4 / e clicca su proprietà.

Comparirà la seguente schermata :



nella quale si dovrà impostare l'IP di WINDOWS :192.168.32.101 , anche la voce DNS dovrà essere impostata inserendo in questo caso l'IP di KALI: 192.168.32.100

Configurazione HTTPS /DNS

Su Kali eseguiamo il comando SUDO NANO /ETC/INETSIM/INETSIM.CONF.

Poco più in basso alla lista troveremo tutta una serie di servizi che possono essere simulati da inetsim . Cancelliamo # dalle voci HTTPS e DNS per renderle attive

```
# 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
#start_service smtps
#start_service pop3
```

In basso a questa lista impostiamo anche l'IP di KALI che abbiamo cambiato in precedenza

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

Non resta altro che configurare il DNS.

Nella stessa lista scorriamo in basso finche non troviamo i settaggi DNS.

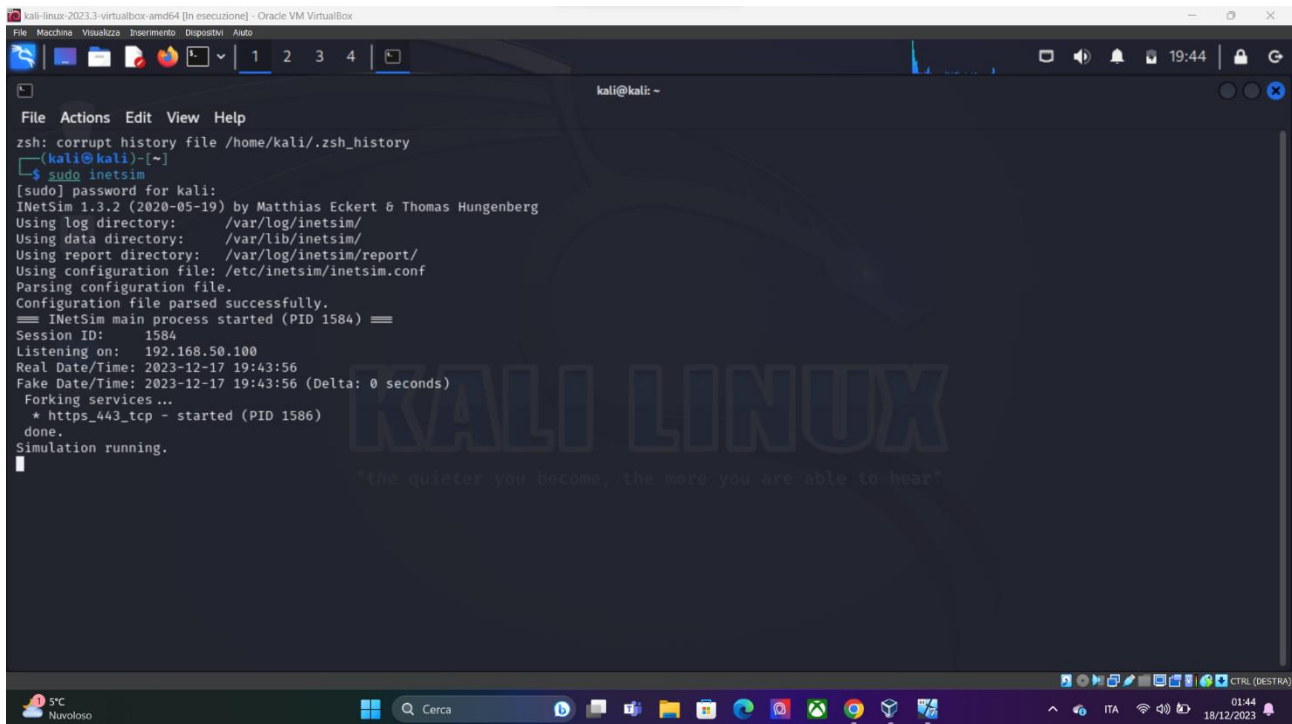
Ai settaggi impostiamo il dominio EPICODE.INTERNAL e attiviamolo cancellando #

```
#####  
# dns_default_domainname  
#  
# Default domain name to return with DNS replies  
#  
# Syntax: dns_default_domainname <domain name>  
#  
# Default: inetsim.org  
#  
dns_default_domainname epicode.internal
```

Settiamo anche l'IP STATIC a finche il nostro dominio risponda a un determinato IP (in questo caso l'IP di KALI)

```
#####  
# dns_static  
#  
# Static mappings for DNS  
#  
# Syntax: dns_static <fqdn hostname> <IP address>  
#  
# Default: none  
#  
dns_static epicode.internal 192.168.32.100  
#dns_static ns1.foo.com 10.70.50.30  
#dns_static ftp.bar.net 10.10.20.30
```

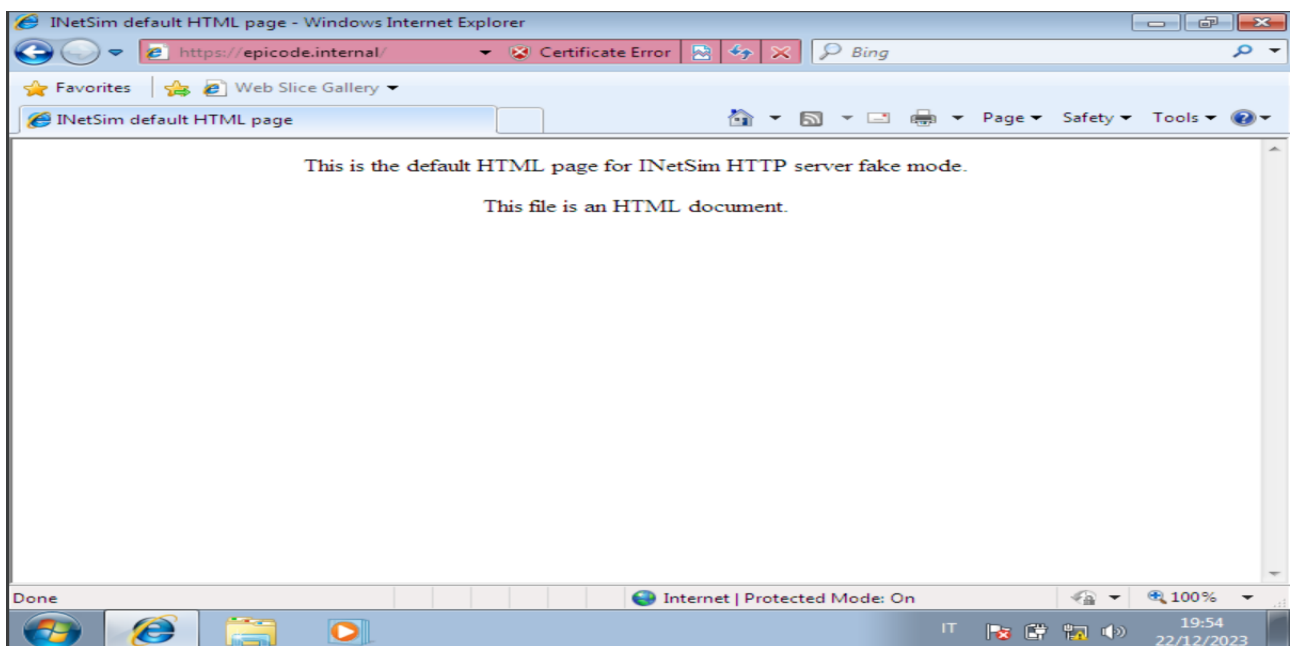
Salviamo le nuove impostazioni e facciamo partire INETSIM con il comando SUDO INETSIM



```
kali@kali: ~  
File Actions Edit View Help  
zsh: corrupt history file /home/kali/.zsh_history  
~  
$ 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 1584) ==  
Session ID: 1584  
Listening on: 192.168.50.100  
Real Date/Time: 2023-12-17 19:43:56  
Fake Date/Time: 2023-12-17 19:43:56 (Delta: 0 seconds)  
Forking services ...  
* https_443_tcp - started (PID 1586)  
done.  
Simulation running.
```

(vediamo che il servizio https utilizzerà la porta 443)

Mettiamo in ascolto Wireshark e successivamente su Windows 7 utilizzando Internet Explorer proviamo ad accedere al sito : <https://epicode.internal> , ci comparirà una pagina fittizia creata da INETSIM



Vediamo cosa ha registrato wireshark se si utilizza https come protocollo

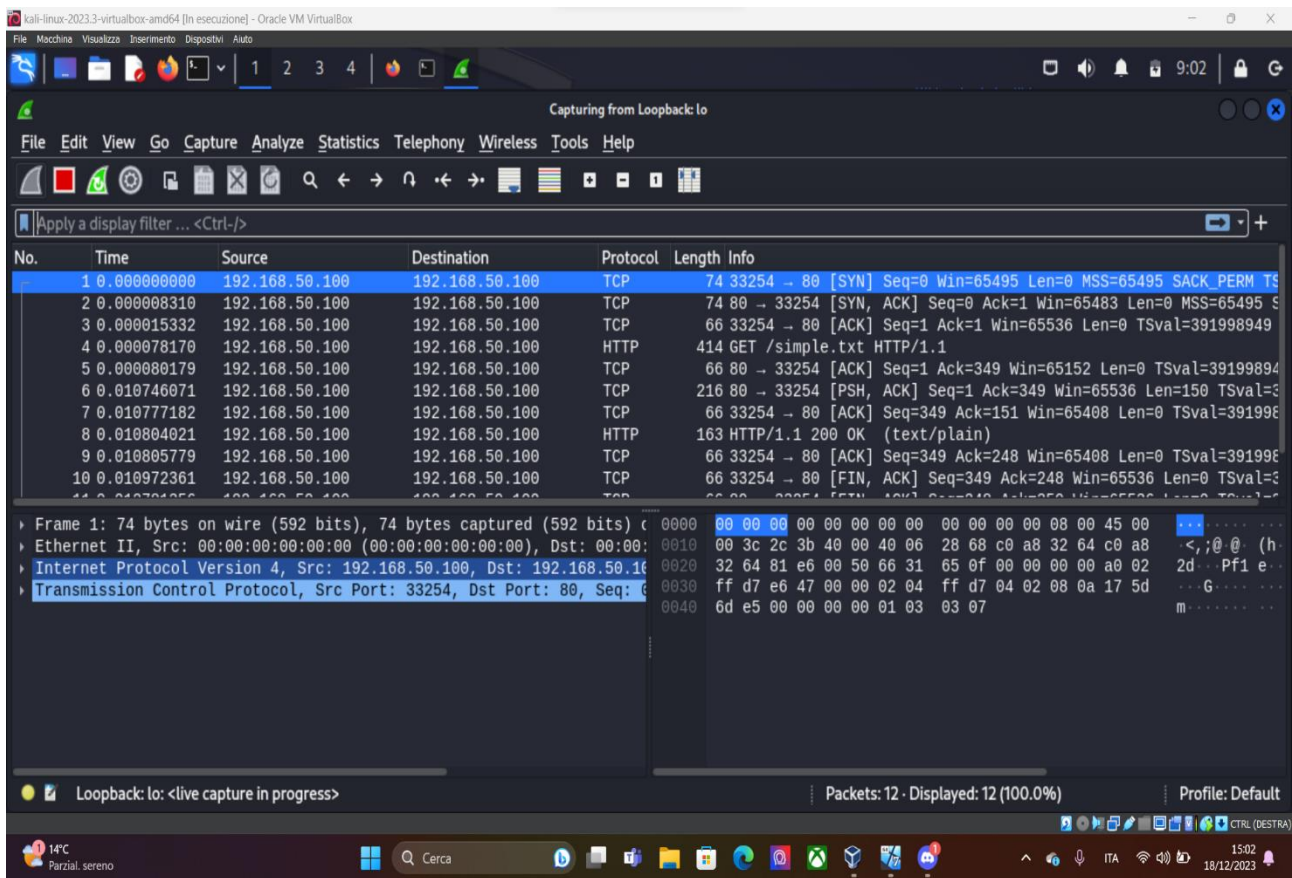
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::28d2:fbe9:deb...	ff02::2	ICMPv6	62	Router Solicitation
2	10.339150069	08:00:27:47:7a:31	Broadcast	ARP	60	Who has 192.168.32.100? Tell 192.168.32.101
3	10.339177137	08:00:27:cb:7e:f5	08:00:27:47:7a:31	ARP	42	192.168.32.100 is at 08:00:27:cb:7e:f5
4	10.339343641	192.168.32.101	192.168.32.100	TCP	66	49188 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM
5	10.339366715	192.168.32.100	192.168.32.101	TCP	66	443 → 49188 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
6	10.339641082	192.168.32.101	192.168.32.100	TCP	60	49188 → 443 [ACK] Seq=1 Ack=1 Win=65700 Len=0
7	10.340921688	192.168.32.101	192.168.32.100	TLSv1	183	Client Hello
8	10.340931287	192.168.32.100	192.168.32.101	TCP	54	443 → 49188 [ACK] Seq=1 Ack=130 Win=64128 Len=0
9	10.369108923	192.168.32.100	192.168.32.101	TLSv1	1373	Server Hello, Certificate, Server Key Exchange, Server Hello Done
10	10.372501297	192.168.32.101	192.168.32.100	TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
11	10.372862373	192.168.32.100	192.168.32.101	TLSv1	113	Change Cipher Spec, Encrypted Handshake Message

Frame 2: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0	0000	ff ff ff ff ff 08 00 27 47 7a 31 08 06 00 01 'Gz1.. ..
Ethernet II, Src: 08:00:27:47:7a:31 (08:00:27:47:7a:31), Dst: Broadcast	0010	08 00 06 04 00 01 08 00 27 47 7a 31 c0 a8 20 65 'Gz1.. e
Address Resolution Protocol (request)	0020	00 00 00 00 00 00 c0 a8 20 64 00 00 00 00 00 00 d.....
Hardware type: Ethernet (1)	0030	00 00 00 00 00 00 00 00 00 00 00 00
Protocol type: IPv4 (0x0800)			
Hardware size: 6			
Protocol size: 4			
Opcode: request (1)			
Sender MAC address: 08:00:27:47:7a:31 (08:00:27:47:7a:31)			
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			

Rifacciamo la stessa operazione ma questa volta settiamo INETSIM con http invece di https :

```
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
#start_service discard_udp
#start_service quotd_tcp
#start_service quotd_udp
#start_service chargen_tcp
```

Vediamo come sarà la lettura di Wireshark se si utilizza HTTP



Osservazioni:

come possiamo notare dalle due letture, la differenza sostanziale e la creazione di un tunnel cifrato per la trasmissione dati “sicura” se si utilizza il protocollo https (TLS) mentre con il protocollo http standard si avranno tutti i dati in chiaro .