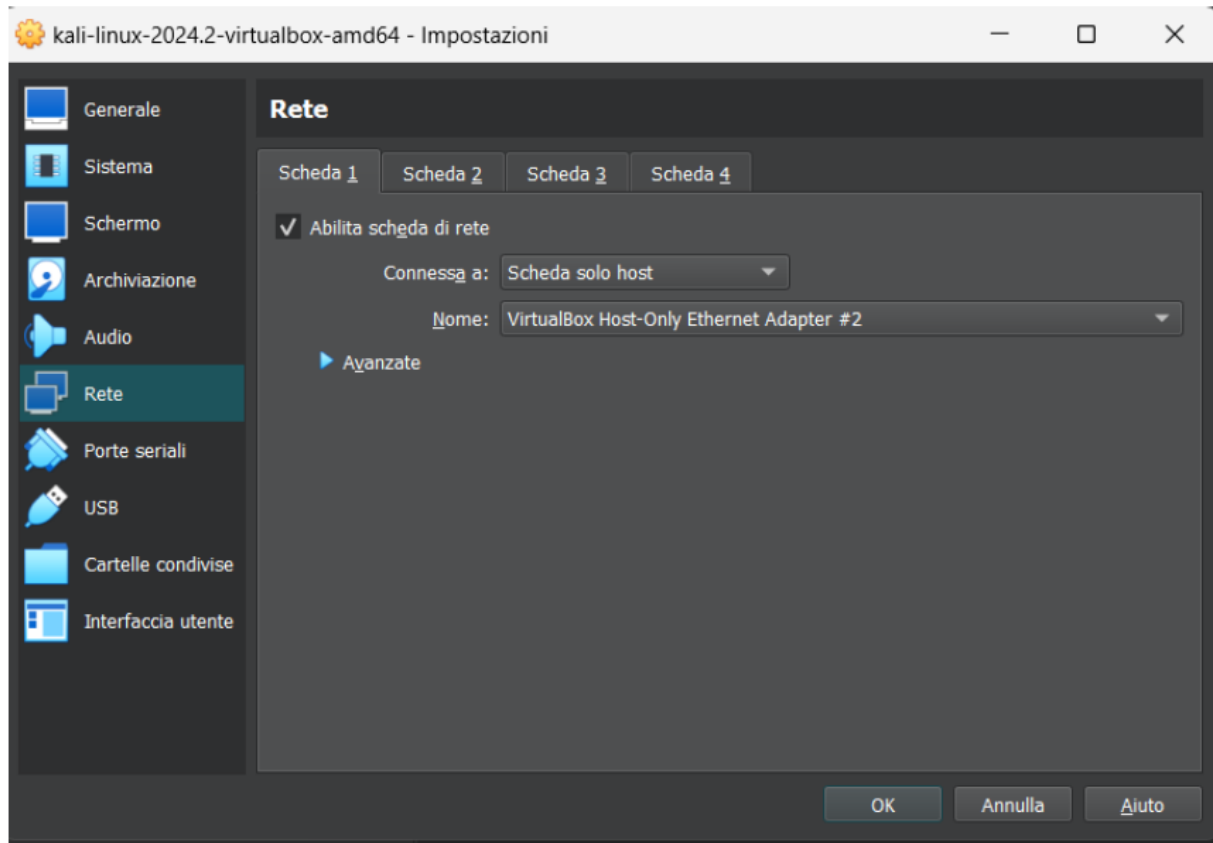


2. A questo punto devo prima impostare la rete solo host AD ENTRAMBE LE VM !!



3. poi bisogna dare un IP statico alle 2 VM

3.1 Kali:

```
sudo vim /etc/network/interfaces
```

```
auto eth0
iface eth0 inet static
address 192.168.32.100
gateway 192.168.32.1
netmask 255.255.255.0
```

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo vim /etc/network/interfaces  
[sudo] password for kali:  
  
(kali@kali)-[~]  
$ sudo cat /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  
gateway 192.168.32.1  
netmask 255.255.255.0  
  
(kali@kali)-[~]  
$
```

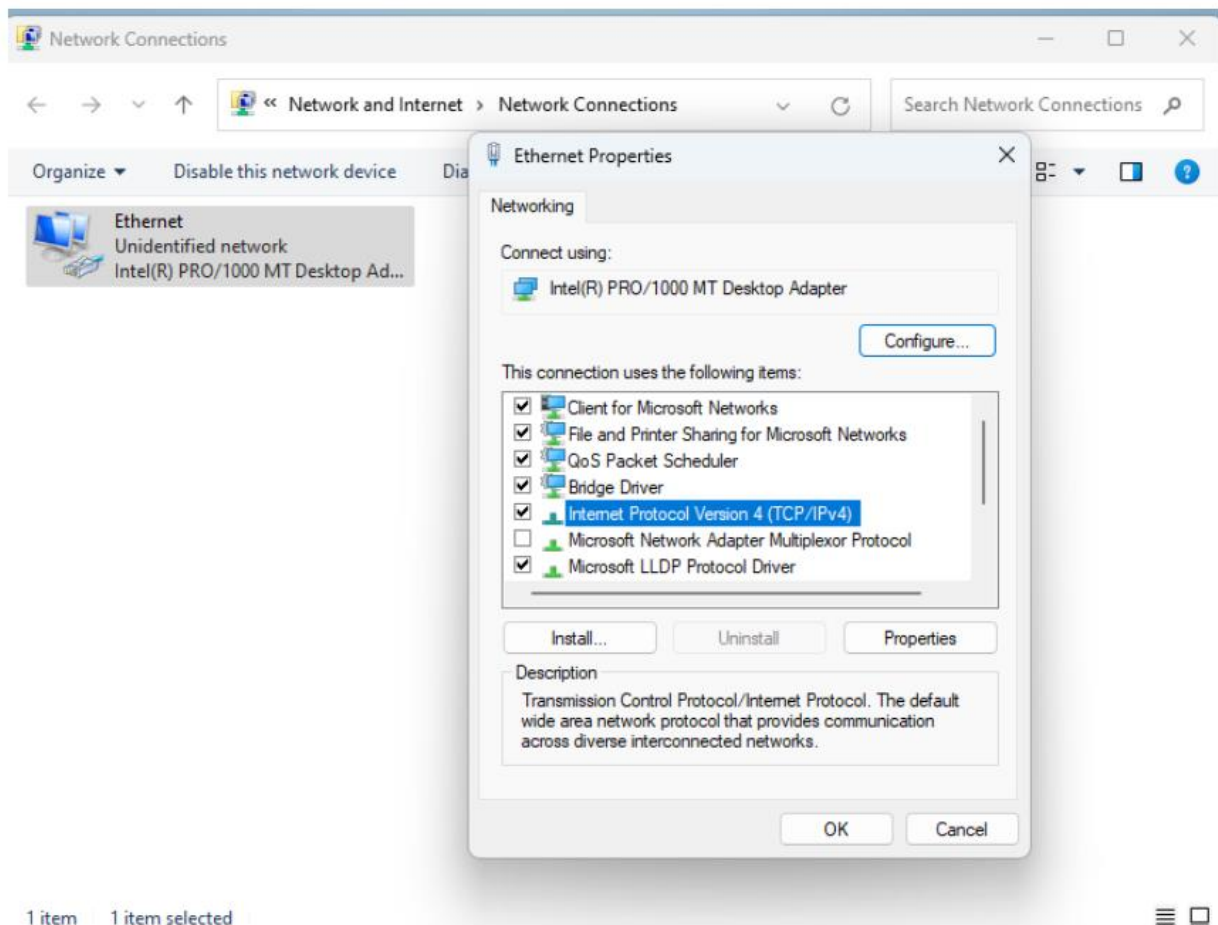
```
sudo systemctl restart networking.service  
sudo systemctl restart NetworkManager.service
```

infine ifconfig per verifica

```
kali@kali: ~  
File Actions Edit View Help  
$ sudo systemctl restart networking.service  
  
(kali@kali)-[~]  
$ sudo systemctl restart NetworkManager.service  
  
(kali@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  
    ether 08:00:27:d2:26:79 txqueuelen 1000 (Ethernet)  
    RX packets 71 bytes 4860 (4.7 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 64 bytes 11635 (11.3 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 8 bytes 480 (480.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 8 bytes 480 (480.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
(kali@kali)-[~]  
$
```

### 3.2. Win

Network connection -> Properties -> IPv4 -> Properties



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

☐ Obtain an IP address automatically

☒ Use the following IP address:

IP address: 192 . 168 . 32 . 101

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 32 . 1

☐ Obtain DNS server address automatically

☒ Use the following DNS server addresses:

Preferred DNS server: . . .

Alternate DNS server: . . .

☐ Validate settings upon exit

Advanced...

OK Cancel

In questa sezione dopo andremo a settare anche il DNS

Per verifica:

cmd -> ipconf

```
Command Prompt

C:\Users\User>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

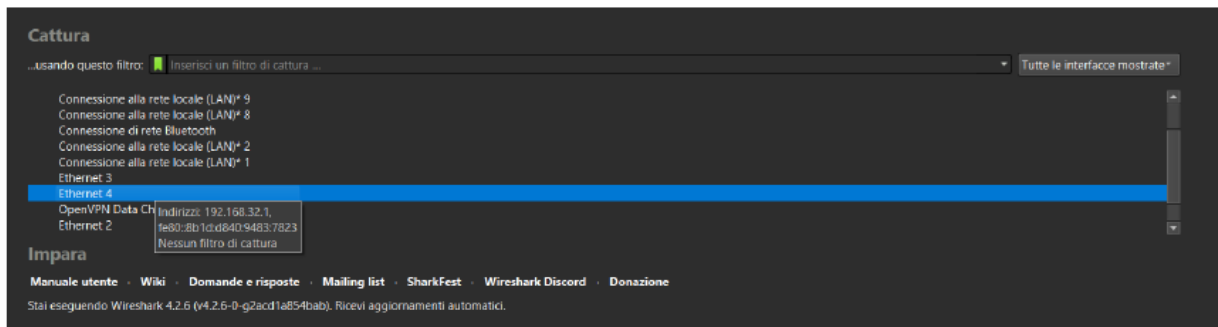
    Connection-specific DNS Suffix  . : 
    IPv4 Address. . . . . : 192.168.32.101
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.32.1

C:\Users\User>
```

Ora sul mio win host (quello che esegue le VM) vado a lanciare wireshark, per vedere la rete che sto usando su VB, vado a cercare le reti attualmente attive nelle impostazioni delle schede di rete dal pannello di controllo, devo cercare quella che finisce con #2

Organizza	Disabilita dispositivo di rete	Esegui diagnosi della connessione	Rinomina connessione	Visualizza stato della connessione	Cambia
Nome	Stato		Nome dispositivo		Connettività
Connessione di rete Bluetooth	Non connesso		Bluetooth Device (Personal Area Network)		
Ethernet	Cavo di rete scollegato		Intel(R) Ethernet Connection (23) I219-LM		
Ethernet 2	Cavo di rete scollegato		TAP-NordVPN Windows Adapter V9		
Ethernet 3	Abilitato		VirtualBox Host-Only Ethernet Adapter		
Ethernet 4	Abilitato		VirtualBox Host-Only Ethernet Adapter #2		
OpenVPN Data Channel Offload for NordVPN	Cavo di rete scollegato		OpenVPN Data Channel Offload		
Wi-Fi	Xiaomi_A682_5G		Intel(R) Wi-Fi 6E AX211 160MHz		Accesso a Internet

Ora lancio wireshark e devo selezionare la stessa rete



Ora devo avviare il servizio http(s) e creare una pagina di test da far richiedere al client windows

Da Kali

```
sudo service apache2 start
```

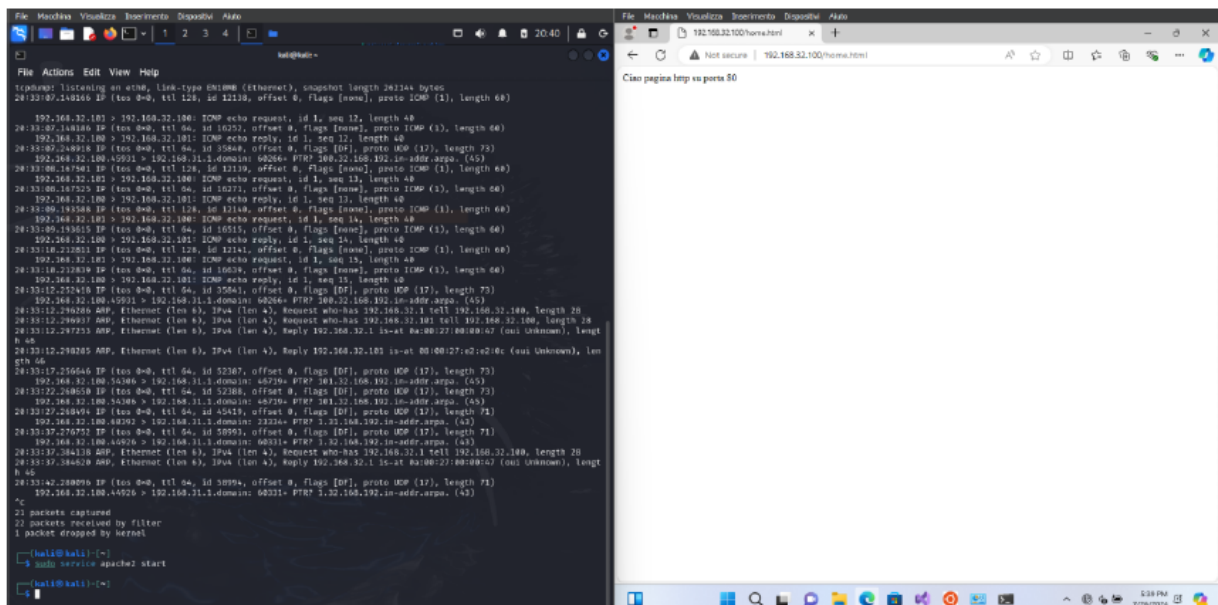
Di default il servizio è in http su porta 80,

le pagine html sono salvate in /var/www/html/

creo una nuova pagina che si chiama home.html in questa cartella e inserisco del testo random

Da Win

non devo fare altro che aprire il browser e cercare <http://192.168.32.100/home.html>



Ora devo fare due operazioni:

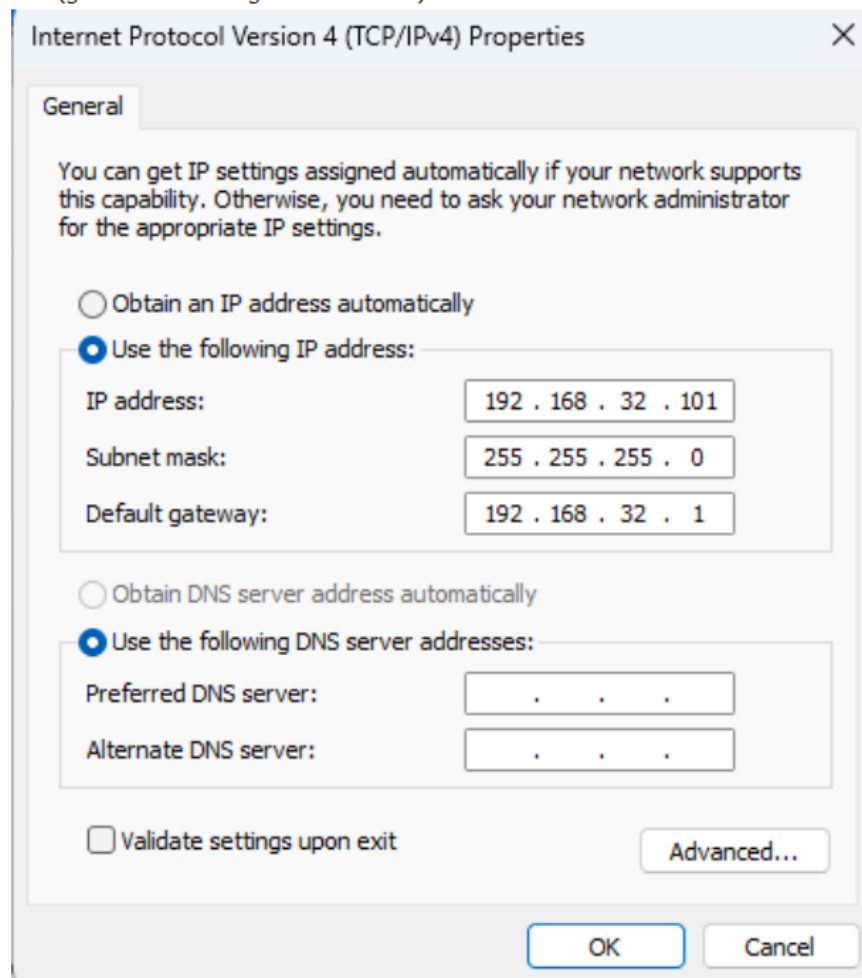
1. aprire un servizio dns su kali per associare il suo IP con l'hostname epicode.internal
2. attivare il servizio https e poi disattivarlo

1. per il primo punto ho due opzioni

1.1 server dns su kali -> associazione hostname ip -> esecuzione servizio -> setting del dns custom su



windows (guarda screen degli IP statici win)



Pro: situazione molto realistica, failsefe e utile per future implementazioni

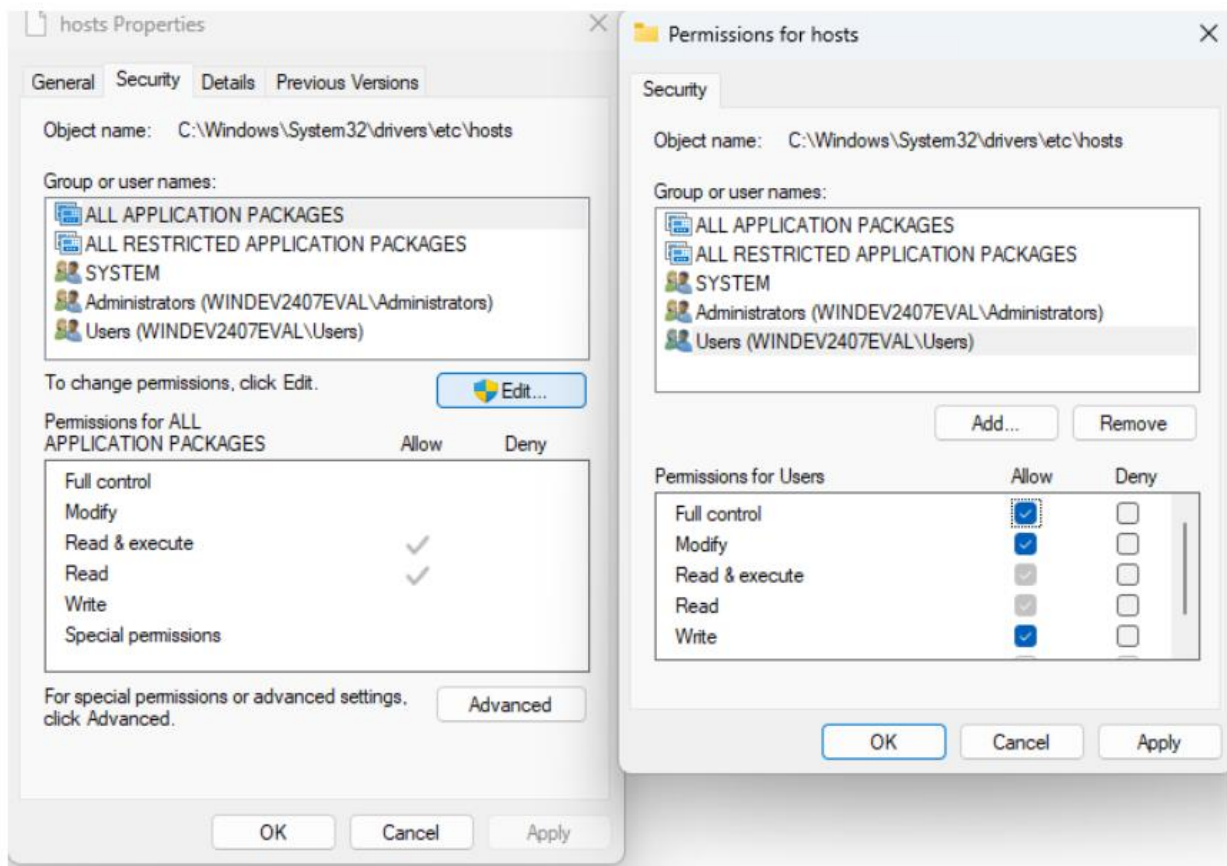
Contro: è facile sbagliare, in caso di esercizio pratico fa perdere un po' tempo

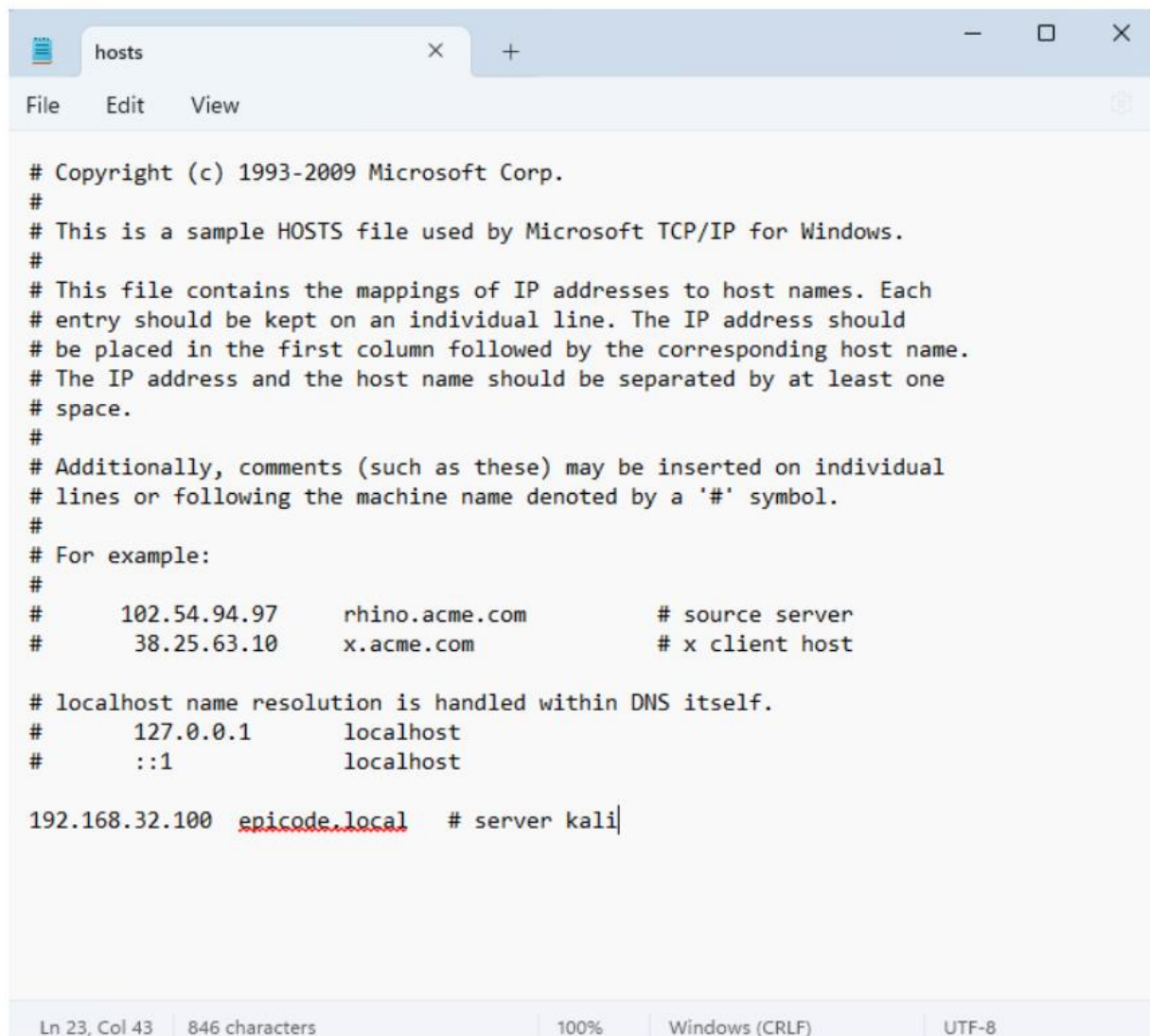
TIPS: visto che stiamo usando Kali, possiamo utilizzare un tool già presente, dnscf

1.2 posso semplicemente modificare il file host di windows associando hostname e ip (in questo caso seguiamo la prima opzione)

navigo su "c:\Windows\System32\Drivers\etc\hosts"

cambio i permessi per il file





```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com          # source server
#       38.25.63.10       x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1         localhost
#       ::1               localhost

192.168.32.100 epicode.local # server kali
```

Ln 23, Col 43 | 846 characters | 100% | Windows (CRLF) | UTF-8

1.1 (parte2) Usiamo dnscchef con i seguenti parametri

```
sudo dnscchef --fakedomains=epicode.local --fakeip=192.168.32.100 --interface=192.168.32.100
```



Ora per completare l'esercizio devo eseguire la connessione prima in https e poi in http catturando i pacchetti con wireshark

Per prima cosa devo crearmi il certificato self-signed da usare su apache e poi connettermi in https su porta 443, tornando poi su http su porta 80

Genero il certificato self-signed

Creo una chiave privata

```
sudo openssl genrsa -aes128 -out privata.key 2048
```

(per comodità uso la pw kali)

14/38

creo richiesta certificato csr (in questa fase dovrò specificare da shell il nome host epicode.internal

```
sudo openssl req -new -days 365 -key privata.key -out richiesta.csr
```

infine genero il certificato da esporre su apache

```
sudo openssl x509 -in richiesta.csr -out certificato.crt -req -signkey privata.key -days 365
```

```
(kali@kali)-[~/Desktop]
$ sudo openssl genrsa -aes128 -out privata.key 2048
[sudo] password for kali:
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:

(kali@kali)-[~/Desktop]
$ sudo openssl req -new -days 365 -key privata.key -out richiesta.csr
Ignoring -days without -x509; not generating a certificate
Enter pass phrase for privata.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IT
State or Province Name (full name) [Some-State]:Rome
Locality Name (eg, city) []:Rome
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Test
Organizational Unit Name (eg, section) []:Test
Common Name (e.g. server FQDN or YOUR name) []:epicode.local
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:

(kali@kali)-[~/Desktop]
$ sudo openssl x509 -in richiesta.csr -out certificato.crt -req -signkey privata.key -days 365
Enter pass phrase for privata.key:
Certificate request self-signature ok
subject=C=IT, ST=Rome, L=Rome, O=Test, OU=Test, CN=epicode.local

(kali@kali)-[~/Desktop]
$
```

Ora devo salvare il certificato.csr in apache (le istruzioni si possono tranquillamente trovare nel web)

Modifico il virtualhost (devo modificare poche impostazioni)



```

<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    ServerName epicode.local

    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile /home/kali/Desktop/certificato.crt
    SSLCertificateKeyFile /home/kali/Desktop/privata.key

    # Server Certificate Chain:
    # Point SSLCertificateChainFile at a file containing the
    # concatenation of PEM encoded CA certificates which form the
    # certificate chain for the server certificate. Alternatively
    # the referenced file can be the same as SSLCertificateFile
    # when the CA certificates are directly appended to the server
    # certificate for convinience.
    #SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt

    # Certificate Authority (CA):
    # Set the CA certificate verification path where to find CA
    # certificates for client authentication or alternatively one
    # huge file containing all of them (file must be PEM encoded)
    # Note: Inside SSLCACertificatePath you need hash symlinks
    # to point to the certificate files. Use the provided
    # Makefile to update the hash symlinks after changes.

```

-- INSERT --

Abilita i moduli mod\_ssl e mod\_headers  
 sudo a2enmod ssl

Abilita la lettura della configurazione SSL

```
sudo a2enconf ssl-params
```

Abilita il Virtual Host SSL di default

```
sudo a2ensite default-ssl
```

Verifica conf

```
sudo apache2ctl configtest
```

Se appare syntax ok posso riavviare apache

```
sudo systemctl restart apache2
```

Usando un host windows, il problema di fondo è il firewall che blocca determinate connessioni, di default sono tutte bloccate

la soluzione è una terza vm che fa da sniffer,

in questo modo posso collegare tutte e 3 alla rete solo host assegnandole un IP libero

NON DIMENTICARE DI IMPOSTARE LA SCHEDA DI RETE DELLA MACCHINA SNIFFER IN MODALITÀ PROMISCUA

```
(kali㉿kali)-[~]
$ cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

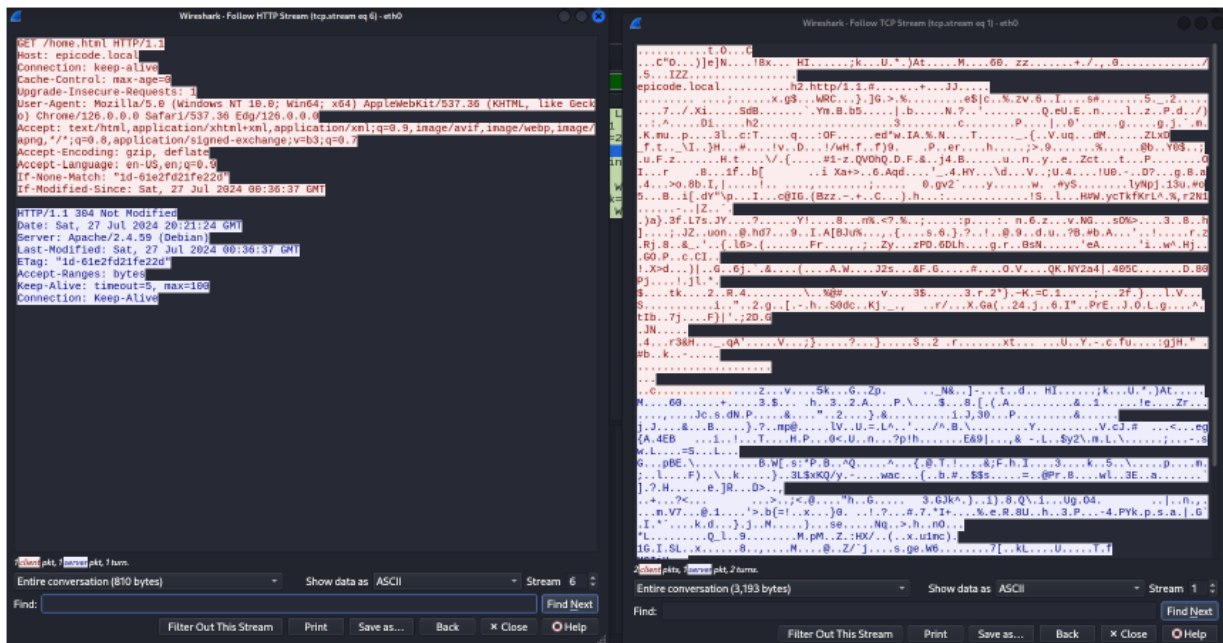
source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

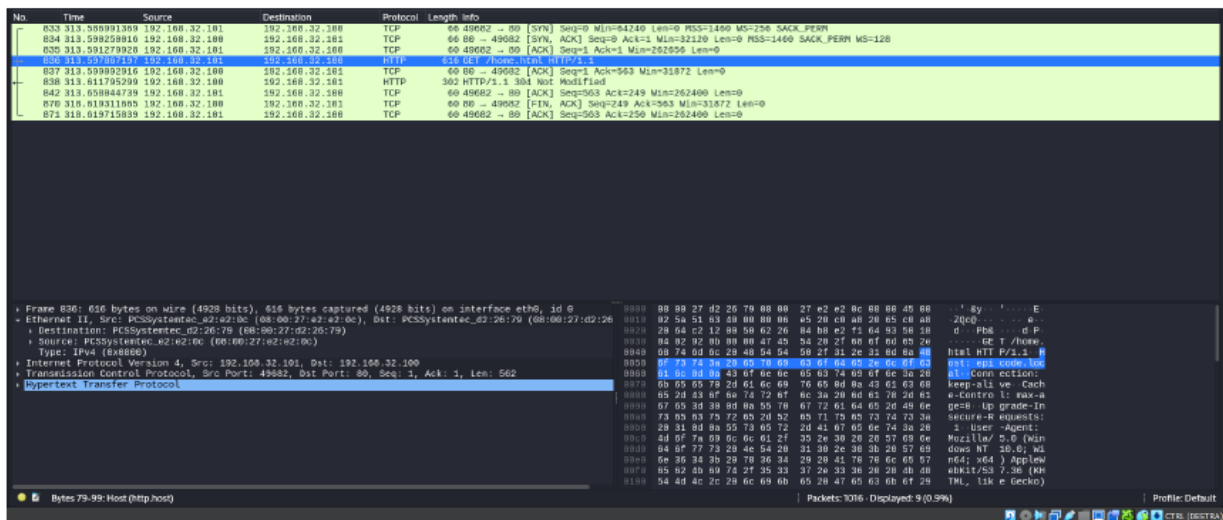
auto eth0
iface eth0 inet static
address 192.168.32.111
gateway 192.168.32.1
netmask 255.255.255.0
```

Destination	Protocol	Length	Info
192.168.32.100	TLSv1.3	84	Change
192.168.32.101	TCP	60	443 -
192.168.32.100	TCP	60	49697
192.168.32.101	TCP	60	443 -
192.168.32.100	TCP	60	49697
192.168.32.100	TLSv1.3	1809	Client
192.168.32.101	TCP	60	443 -
192.168.32.100	TLSv1.3	1494	Server
192.168.32.100	TLSv1.3	118	Change
192.168.32.100	TLSv1.3	788	Applic
192.168.32.101	TLSv1.3	133	Applic
192.168.32.101	TLSv1.3	467	Applic
192.168.32.101	TCP	60	49697
192.168.32.100	TCP	60	49697
192.168.32.100	TCP	60	49697
192.168.32.100	TLSv1.3	78	Applic
192.168.32.100	TCP	60	443
192.168.32.101	TCP	60	49697

Di seguito gli screen delle connessioni http e https



## Screen con MAC-Address



## Screen MAC-Address con connessione cifrata



Wireshark interface showing a network capture. The packet list on the left displays a series of TCP and TLSv1.3 packets between 192.168.32.101 and 192.168.32.100. The selected packet (No. 61) is a TLSv1.3 Client Hello. The packet details pane on the right shows the structure of the Client Hello, including the TLS Version (3), Random (60 bytes), Session ID (0), Cipher Suites (8), Compression Methods (0), and Extensions (0).

No.	Time	Source	Destination	Protocol	Length	Info
60	13.32599755	192.168.32.101	192.168.32.100	TCP	60	443 → 49677 [ACK] Seq=0 Ack=1 Min=32120 Len=0 MSS=1460 SACK_PERM WS=128
61	13.32599756	192.168.32.101	192.168.32.100	TCP	60	49677 → 443 [ACK] Seq=1 Ack=1 Min=262656 Len=0
62	13.326988757	192.168.32.101	192.168.32.100	TLSv1.3	1777	Client Hello (SN=splcode.local)
63	13.327285233	192.168.32.100	192.168.32.101	TCP	60	443 → 49677 [ACK] Seq=1 Ack=1724 Min=31872 Len=0
64	13.330211941	192.168.32.100	192.168.32.101	TLSv1.3	1494	Server Hello, Change Cipher Spec, Application Data, Application Data, Application Data, Application Data
65	13.330457392	192.168.32.101	192.168.32.100	TLSv1.3	84	Change Cipher Spec, Application Data
66	13.330490108	192.168.32.101	192.168.32.100	TCP	60	49677 → 443 [FIN, ACK] Seq=1754 Ack=1441 Min=261120 Len=0
67	13.337898958	192.168.32.101	192.168.32.100	TCP	60	443 → 49677 [FIN, ACK] Seq=1441 Ack=1755 Min=31872 Len=0
68	13.339486938	192.168.32.101	192.168.32.100	TCP	60	49677 → 443 [ACK] Seq=1755 Ack=1441 Min=261120 Len=0
69	13.312784879	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [SYN] Seq=0 Min=84240 Len=0 MSS=1460 WS=256 SACK_PERM
70	13.312785835	192.168.32.100	192.168.32.101	TCP	60	443 → 49676 [SYN, ACK] Seq=0 Ack=1 Min=32120 Len=0 MSS=1460 SACK_PERM WS=128
71	13.414822388	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [ACK] Seq=1 Ack=1 Min=262656 Len=0
72	13.425089788	192.168.32.101	192.168.32.100	TLSv1.3	1841	Client Hello (SN=splcode.local)
73	13.426539274	192.168.32.100	192.168.32.101	TCP	60	443 → 49676 [ACK] Seq=1 Ack=1788 Min=31872 Len=0
74	13.442734748	192.168.32.100	192.168.32.101	TLSv1.3	1494	Server Hello, Change Cipher Spec, Application Data, Application Data, Application Data, Application Data
75	13.445086102	192.168.32.101	192.168.32.100	TLSv1.3	84	Change Cipher Spec, Application Data
76	13.448386187	192.168.32.100	192.168.32.101	TCP	60	443 → 49676 [FIN, ACK] Seq=1441 Ack=1818 Min=31872 Len=0
77	13.451124826	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [ACK] Seq=1818 Ack=1441 Min=261120 Len=0
78	13.450683769	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [FIN, ACK] Seq=1818 Ack=1441 Min=261120 Len=0
79	13.450553539	192.168.32.100	192.168.32.101	TCP	60	443 → 49676 [ACK] Seq=1442 Ack=1819 Min=31872 Len=0
80	13.080330314	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [SYN] Seq=0 Min=84240 Len=0 MSS=1460 WS=256 SACK_PERM
81	13.080337174	192.168.32.101	192.168.32.100	TCP	60	49600 → 443 [SYN] Seq=0 Min=84240 Len=0 MSS=1460 WS=256 SACK_PERM
82	13.080337278	192.168.32.100	192.168.32.101	TCP	60	443 → 49676 [SYN, ACK] Seq=0 Ack=1 Min=32120 Len=0 MSS=1460 SACK_PERM WS=128
83	13.080337386	192.168.32.101	192.168.32.100	TCP	60	49676 → 443 [ACK] Seq=1 Ack=1 Min=262656 Len=0
84	13.080337688	192.168.32.101	192.168.32.100	TLSv1.3	1777	Client Hello (SN=splcode.local)
85	13.080337688	192.168.32.101	192.168.32.100	TCP	60	443 → 49676 [FIN, ACK] Seq=0 Ack=1 Min=32120 Len=0 MSS=1460 SACK_PERM WS=256 SACK_PERM

Frame 61: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0

- Ethernet II, Src: PCSSystemec\_02:e2:0c (08:00:27:e2:0c), Dst: PCSSystemec\_02:26:79 (08:00:27:d2:26)
  - Destination: PCSSystemec\_02:26:79 (08:00:27:d2:26:79)
  - Source: PCSSystemec\_02:e2:0c (08:00:27:e2:0c:0c)
  - Type: IPv4 (0x8000)
  - Padding: 000000000000
- Internet Protocol Version 4, Src: 192.168.32.101, Dst: 192.168.32.100
- Transmission Control Protocol, Src Port: 49677, Dst Port: 443, Seq: 1, Ack: 1, Len: 0

Transmission Control Protocol: Protocol | Packets: 1027 | Displayed: 83 (8.1%) | Profile: Default