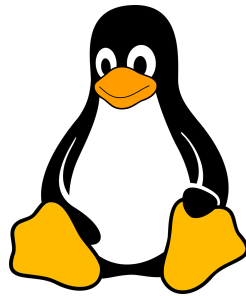


Laboratorio e Amministrazione di Sistema



Configurazione Server OpenVPN



Cogotti Giulia

884383

29/05/2023

Indice

1. Introduzione.....	2
2. Configurazione.....	2
2.1. Autorità Certificante (CA).....	2
2.2. Server.....	4
2.3. Firma del certificato da parte della CA.....	6
2.4. Client.....	9
2.5. Configurazione server OpenVPN.....	11
2.6. Configurazione client OpenVPN.....	14
3. Testing.....	18
3.1. Test 1: VM server e client nella stessa LAN.....	18
3.2. Test 2: VM server isolata e il mio pc come client.....	20
3.3. Test 3: reindirizzamento di tutto il traffico sulla VPN.....	22
4. Riferimenti.....	25

1. Introduzione

Una VPN, acronimo di Virtual Private Network, ovvero rete privata virtuale, è uno strumento che consente agli utenti di collegarsi alla rete in modo sicuro e anonimo tramite una connessione criptata.

OpenVPN è un protocollo di connessione open source che stabilisce una connessione privata, o tunnel, tra un client VPN e un server VPN.

2. Configurazione

2.1. Autorità Certificante (CA)

Un'autorità di certificazione (CA) è un'entità responsabile dell'emissione di certificati digitali per verificare le identità su Internet. Essa consente di configurare, testare ed eseguire programmi che richiedono connessioni crittografate tra client e server.

Per configurarla ho creato una nuova macchina virtuale con hostname **certificate-auth**.

ATTENZIONE: come specificato da *openvpn*, un errore da non fare è quello di posizionare i file CA sul server OpenVPN. Una CA richiede una chiave privata che viene utilizzata per firmare i certificati usati da client e server. Se si perde il controllo di questa chiave privata, allora non ci si può più fidare di alcun certificato della CA. Chiunque abbia accesso a questa chiave privata CA può firmare nuovi certificati, che possono quindi connettersi al server OpenVPN senza dover modificare nulla sul server VPN. Posizionare i file su un'altra VM permette di accenderla solo quando necessario e tenerla offline il resto del tempo.

Il primo passaggio da fare è quello di installare il pacchetto *easy-rsa*, uno strumento di gestione della CA utilizzato per generare una chiave privata e un certificato root pubblico che in seguito verranno usati per firmare le richieste del client e del server.

```
cogotti-giulia@certificate-auth:~$ sudo apt install easy-rsa
```

ATTENZIONE: eseguire i prossimi comandi senza usare *sudo* perchè un utente normale deve interagire con la CA senza necessitare di particolari privilegi.

Creazione di una directory *easy-rsa* nella home dell'utente (non root) per creare un'infrastruttura a chiave pubblica (PKI). Questa cartella sarà usata per creare

collegamenti simbolici che puntano ai file installati del pacchetto easy-rsa, in questo modo ogni modifica al pacchetto sarà rispecchiata nella cartella appena creata.

```
cogotti-giulia@certificate-auth:/$ mkdir ~/easy-rsa
cogotti-giulia@certificate-auth:/$ ln -s /usr/share/easy-rsa/* ~/easy-rsa/
```

Assegnamento dei permessi solo al proprietario

```
cogotti-giulia@certificate-auth:~$ chmod 700 /home/cogotti-giulia/easy-rsa
```

Inizializzazione della PKI all'interno della directory easy-rsa

```
cogotti-giulia@certificate-auth:~$ cd ~/easy-rsa
cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa init-pki

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /home/cogotti-giulia/easy-rsa/pki
```

A questo punto è stata creata una directory pki contenente tutti i file necessari alla creazione della CA.

Ora è necessario creare un file `vars` contenente alcuni valori di default

```
set_var EASYRSA_REQ_COUNTRY    "EU"
set_var EASYRSA_REQ_PROVINCE   "Sud Sardegna"
set_var EASYRSA_REQ_CITY       "Cagliari"
set_var EASYRSA_REQ_ORG        "My Organization"
set_var EASYRSA_REQ_EMAIL      "me@example.it"
set_var EASYRSA_REQ_OU         "My Organizational Unit"
```

```
# Choose a size in bits for your keypairs. The recommended value is 2048. Using
# 2048-bit keys is considered more than sufficient for many years into the
# future. Larger key sizes will slow down TLS negotiation and make key/DH param
# generation take much longer. Values up to 4096 should be accepted by most
# software. Only used when the crypto alg is rsa (see below.)
```

```
set_var EASYRSA_KEY_SIZE       2048
```

```
# The default crypto mode is rsa; ec can enable elliptic curve support.
# Note that not all software supports ECC, so use care when enabling it.
# Choices for crypto alg are: (each in lower-case)
# * rsa
# * ec
# * ed
```

```
set_var EASYRSA_ALGO           ec
```

```
# Cryptographic digest to use.
# Do not change this default unless you understand the security implications.
# Valid choices include: md5, sha1, sha256, sha224, sha384, sha512
```

```
set_var EASYRSA_DIGEST         "sha512"
```

Infine va eseguito il comando `./easyrsa build-ca` il quale creerà due importanti file: `ca.crt` e `ca.key`.

ca.crt è il file del certificato pubblico della CA. Utenti, server e client lo utilizzano per verificare che fanno parte della stessa rete. Ogni utente e server che utilizza la CA dovrà avere una copia di questo file, ci si farà riferimento per assicurarsi che qualcuno non stia impersonando un sistema e non stia eseguendo un attacco man-in-the-middle.

ca.key è la chiave privata utilizzata dalla CA per firmare i certificati per server e client. Se un utente malintenzionato ottiene l'accesso alla CA e quindi a questo file, sarà necessario distruggere la CA. Per questo motivo, come detto in precedenza, il file **ca.key** dovrà essere solo nella macchina CA ed essa dovrebbe idealmente essere tenuta offline fino a quando non serve che essa firmi una richiesta.

```
cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa build-ca
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

Enter New CA Key Passphrase:
Re-Enter New CA Key Passphrase:
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.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:certificate-auth

CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
/home/cogotti-giulia/easy-rsa/pki/ca.crt
```

2.2. Server

Per il server ho creato un ulteriore macchina virtuale con hostname **cgt-server**. Il primo passo da fare è quello di installare i pacchetti **openvpn** e **easy-rsa**. In seguito, come per la CA, creo la cartella **easy-rsa** nella home dell'utente non root e il link simbolico al pacchetto installato.

```
cogotti-giulia@cgt-server:~$ mkdir ~/easy-rsa
cogotti-giulia@cgt-server:~$ ln -s /usr/share/easy-rsa/* ~/easy-rsa/

cogotti-giulia@cgt-server:~$ sudo chown cogotti-giulia ~/easy-rsa
cogotti-giulia@cgt-server:~$ chmod 700 ~/easy-rsa
```

Creazione di un PKI per il server OpenVPN, usata per gestire i certificati del server e del client invece di mandarli direttamente al server CA. Anche in questo caso ho bisogno di un file **vars** contenente solo due righe che garantiscono che le chiavi private e le richieste di certificato siano configurate per usare la crittografia a curva

ellittica (ECC) per generare chiavi e firme sicure per client e server. La scelta di questo algoritmo permette di aumentare la velocità rispetto al classico RSA.

```
GNU nano 6.2
set_var EASYRSA_ALGO "ec"
set_var EASYRSA_DIGEST "sha512"
```

Dato che il server OpenVPN e il server CA hanno directory separate, è necessario eseguire il comando *init-pki* anche su server. Essa verrà usata solo come luogo comodo e centralizzato per archiviare richieste di certificati e certificati pubblici.

```
cogotti-giulia@cgt-server:~/easy-rsa$ ./easyrsa init-pki
```

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /home/cogotti-giulia/easy-rsa/pki

Il prossimo passo da fare è quello di generare una chiave privata e una richiesta di firma del certificato (CSR) sul server. Questa richiesta sarà poi inviata alla CA per la firma, creando il certificato richiesto. Una volta ottenuto sarà trasferito al server e installato per l'utilizzo.

Il comando `./easymrsa gen-req server nopass` crea una chiave privata e una richiesta di firma del certificato per il server.

[illegible]

Ora copio la chiave del server nella directory `/etc/openvpn/server`

```
cogotti-giulia@cgt-server:~/easy-rsa$ sudo cp /home/cogotti-giulia/easy-rsa/pki/private/server.key /etc/openvpn/server
```

2.3. Firma del certificato da parte della CA

Il certificato del server viene inviato alla CA tramite comando scp.

ATTENZIONE: il comando SCP (secure copy protocol) consente di copiare in modo sicuro file tra due sistemi. Il protocollo assicura che la trasmissione sia criptata utilizzando la codifica con connessione SSH (secure shell).

```
cogotti-giulia@cgt-server:~$ scp ~/easy-rsa/pki/reqs/server.req cogotti-giulia@192.168.15.59:~/server.req
cogotti-giulia@192.168.15.59's password:
server.req                                100% 887    17.4KB/s   00:00
```

Passando alla macchina della CA, è necessario importare la richiesta tramite il comando `./easyrsa import-req ~/server.req server`.

```
cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa import-req ~/server.req server
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

The request has been successfully imported with a short name of: server
You may now use this name to perform signing operations on this request.
```

La firma avviene eseguendo lo script `./easyrsa` con l'opzione `sign-req` seguita dal tipo di richiesta (client o server, in questo caso sarà server) e nome comune.

```

cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa sign-req server server
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 825 days:

subject=
  commonName          = server

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /home/cogotti-giulia/easy-rsa/pki/easy-rsa-6842.aNCMXu/tmp.9XKvnF
Enter pass phrase for /home/cogotti-giulia/easy-rsa/pki/private/ca.key:
40674FA82F7F0000:error:0700006C:configuration file routines:NCONF_get_string:no value:../crypto/conf
/conf_lib.c:315:group=<NULL> name=unique_subject
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName          :ASN.1 12:'server'
Certificate is to be certified until Aug  2 14:38:32 2025 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /home/cogotti-giulia/easy-rsa/pki/issued/server.crt

```

A questo punto la richiesta di certificato del server OpenVPN è stata firmata utilizzando la chiave privata del server CA. Il file [server.crt](#) risultante contiene la chiave di crittografia pubblica del server OpenVPN e una firma del server CA. Lo scopo di tutto questo è dire a chiunque si fidi del server CA che può fidarsi anche del server OpenVPN quando si connette ad esso.

Infine invio una copia dei file [server.crt](#) e [ca.crt](#) al server OpenVPN.

```

cogotti-giulia@certificate-auth:~/easy-rsa$ scp pki/issued/server.crt cogotti-giulia@192.168.15.147:~/tmp/server.crt
cogotti-giulia@192.168.15.147's password:
server.crt                                100% 4631      3.7MB/s   00:00
cogotti-giulia@certificate-auth:~/easy-rsa$ scp pki/ca.crt cogotti-giulia@192.168.15.147:~/tmp/ca.crt
cogotti-giulia@192.168.15.147's password:
ca.crt                                    100% 1224      12.3KB/s   00:00
cogotti-giulia@certificate-auth:~/easy-rsa$

```

Tornando alla VM del server OpenVPN copio i file ricevuti nella cartella [/etc/openvpn/server](#)

```

cogotti-giulia@cgt-server:~/easy-rsa$ sudo ls /etc/openvpn/server
ca.crt  server.crt  server.key

```


Per aggiungere un ulteriore livello di sicurezza, va aggiunta una chiave segreta condivisa che il server OpenVPN e tutti i client utilizzano con la direttiva `<tls-crypt>` di OpenVPN. Questa opzione viene utilizzata per offuscare il certificato TLS usato quando client e server si connettono all'inizio. Viene utilizzato anche dal server OpenVPN per eseguire controlli rapidi sui pacchetti in arrivo: se un pacchetto viene firmato utilizzando la chiave precondivisa, il server lo elabora; se non è firmato, il server sa che proviene da una fonte non attendibile e può scartarlo senza dover eseguire ulteriori operazioni di decrittazione. Questa opzione aiuterà a garantire che il server OpenVPN sia in grado di far fronte a traffico non autenticato, scansioni delle porte e attacchi Denial of Service, che possono bloccare le risorse del server. Inoltre, rende più difficile identificare il traffico di rete OpenVPN.

Per generare la chiave precondivisa `tls-crypt`, è necessario eseguire il comando `openvpn --genkey secret ta.key`

```
cogotti-giulia@cgt-server:~/easy-rsa$ openvpn --genkey secret ta.key
cogotti-giulia@cgt-server:~/easy-rsa$ cat ta.key
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
5f389364d9b684d330621be963731a73
22ae93cbce17821610e0cfbebb84e962
48f9555818effb7f5d65b5a38a9119ee
c2e8fc37f3d2b0d70c9252728786848c
c322879611f43babaa354d165cf4d60b
0d73a1b19aca5edb46f459dc9376ecdb
68fe0863a1e1803823bde112d255896d
df4525e1fe915d2dc102a6b9c7605201
46d928daab110fd50574018b2e0b551e
834970118595ee194afe7e888ca6476a
30f1b7d3c54092dd969fae136511c456
c5e8d0602f31a23c4e06cd0677b77854
f8e84c5d9832e88dc6a344ef2d1b30b3
99d2e77d72eccc7fe85165fc0cb8b41d
6fac0d51ee53a3652a217663dd0a107d
f93eaf0f64925c48cc0b927b5e237926
-----END OpenVPN Static key V1-----
```

Copio la chiave appena generata nella cartella `/etc/openvpn/server`

```
cogotti-giulia@cgt-server:~/easy-rsa$ sudo cp ta.key /etc/openvpn/server
cogotti-giulia@cgt-server:~/easy-rsa$ sudo ls /etc/openvpn/server
ca.crt  server.crt  server.key  ta.key
```

2.4. Client

Un possibile modo per configurare i client sarebbe quello di generare una chiave privata e una richiesta di certificato su ogni client e inviarli alla CA per la firma. Un modo più efficiente è quello di generare la richiesta sul server OpenVPN. Un vantaggio di questo metodo è la possibilità di creare uno script di configurazione automatica dei file di configurazione (chiavi e certificati) dei client. Questo processo verrà fatto per un unico client, in modo analogo si può ripetere per N di essi.

Innanzitutto sul server OpenVPN creo delle directory per archiviare certificato e chiave dei client.

```
cogotti-giulia@cgt-server:~$ mkdir -p ~/client-configs/keys
cogotti-giulia@cgt-server:~$ chmod -R 700 ~/client-configs
```

Generazione della richiesta del *client1*

[illegible]

Copia della chiave nella cartella precedentemente creata

```
cogotti-giulia@cgt-server:~/easy-rsa$ cp pki/private/client1.key ~/client-configs/keys/
```

A questo punto seguono una serie di passaggi analoghi a quanto avvenuto per la firma della richiesta del server OpenVPN, solo che stavolta viene fatto per il *client1*.

Invio della richiesta al server CA

```
cogotti-giulia@cgt-server:~/easy-rsa$ scp pki/reqs/client1.req cogotti-giulia@192.168.15.59:~/tmp/client1.req
cogotti-giulia@192.168.15.59's password:
client1.req 100% 887 1.2MB/s 00:00
```

Nella macchina della CA viene importata la richiesta e firmata

```
cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa import-req ~/tmp/client1.req client1
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

The request has been successfully imported with a short name of: client1
You may now use this name to perform signing operations on this request.
```

```
cogotti-giulia@certificate-auth:~/easy-rsa$ ./easyrsa sign-req client client1
Using SSL: openssl OpenSSL 3.0.2 15 Mar 2022 (Library: OpenSSL 3.0.2 15 Mar 2022)

You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a client certificate for 825 days:

subject=
  commonName          = client1

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes
Using configuration from /home/cogotti-giulia/easy-rsa/pki/easy-rsa-1903.J39XdP/tmp.aGnZqj
Enter pass phrase for /home/cogotti-giulia/easy-rsa/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName      :ASN.1 12:'client1'
Certificate is to be certified until Aug  2 16:59:08 2025 GMT (825 days)

Write out database with 1 new entries
Data Base Updated

Certificate created at: /home/cogotti-giulia/easy-rsa/pki/issued/client1.crt
```

Trasferimento del certificato [client1.crt](#) appena creato al server OpenVPN

```
cogotti-giulia@certificate-auth:~/easy-rsa$ scp pki/issued/client1.crt cogotti-giulia@192.168.15.147:~/tmp/client1.crt
cogotti-giulia@192.168.15.147's password:
client1.crt 100% 4517 5.5MB/s 00:00
```

Copia del certificato nella cartella [keys](#) precedentemente creata e assegnazione dei permessi adeguati

```
cogotti-giulia@cgt-server:~$ cp ~/tmp/client1.crt ~/client-configs/keys/
cogotti-giulia@cgt-server:~$ cp ~/easy-rsa/ta.key ~/client-configs/keys/
cogotti-giulia@cgt-server:~$ sudo cp /etc/openvpn/server/ca.crt ~/client-configs/keys/
[sudo] password for cogotti-giulia:
cogotti-giulia@cgt-server:~$ sudo chown cogotti-giulia:cogotti-giulia ~/client-configs/keys/*
```

Ora i certificati e le chiavi del server OpenVPN del client sono stati correttamente generati e archiviati nelle directory all'interno del server OpenVPN.

2.5. Configurazione server OpenVPN

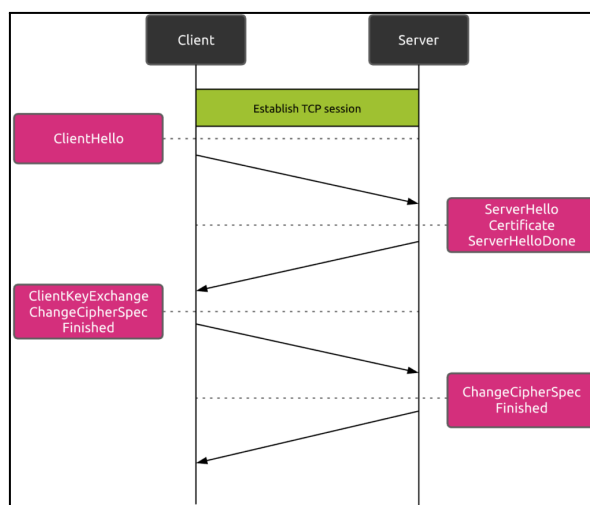
La configurazione va inserita in un file `server.conf` all'interno della cartella `/etc/openvpn/server`. Per semplicità copio la configurazione di esempio situata in `server.conf` e successivamente la modifico per adeguarsi a ciò che mi serve.

```
cogotti-giulia@cgt-server:~$ sudo cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf /etc/openvpn/server/
```

Inizialmente scelgo la porta e il protocollo

```
# Which TCP/UDP port should OpenVPN listen on?  
# If you want to run multiple OpenVPN instances  
# on the same machine, use a different port  
# number for each one. You will need to  
# open up this port on your firewall.  
port 4434  
  
# TCP or UDP server?  
;proto tcp  
proto udp
```

Modifico alcune righe per la sicurezza relative al protocollo TLS. Tramite `tls-crypt` il messaggio iniziale sarà criptato con una chiave condivisa precedentemente. Questo fa in modo che l'inizializzazione dell'handshake TLS sia nascosta, previene gli attacchi DoS (rifiuta la connessione subito invece di aspettare che l'aggressore apra migliaia di connessioni TLS contemporaneamente senza un valido certificato) e inoltre permette di crittografare due volte i dati (da `tls-crypt` e poi dalla sessione TLS).



```
# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
#   openvpn --genkey tls-auth ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0
tls-crypt ta.key
```

Nella sezione sui cifrari cambio il valore in [AES-256-GCM](#) perché offre un miglior livello di crittografia, migliori prestazioni ed è ben supportato dai client. La direttiva auth invece specifica l'algoritmo del messaggio HMAC, uso [SHA256](#).

```
# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the ncp-cipher option in the manpage
;cipher AES-256-CBC
cipher AES-256-GCM
auth SHA256
```

Dato che ho configurato i certificati per usare la crittografia ECC, non è necessario utilizzare un file Diffie-Hellman

```
# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
;dh dh2048.pem
dh none
```

Ora, voglio che openvpn funzioni senza privilegi una volta avviato. Quindi inserisco le direttive user e group che avranno rispettivamente [nobody](#) e [nogroup](#) (il gruppo può variare in base alla distro di linux utilizzata)

```
# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this out on
# non-Windows systems.
user nobody
group nogroup
```

ATTENZIONE: Sebbene sia possibile forzare il reindirizzamento di tutto il traffico della rete sulla vpn, inserendo le seguenti righe nella configurazione del server, al momento non saranno aggiunte [\[vedi sezione 3.3\]](#)

```
push "redirect-gateway [local] def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
```

Queste ultime due righe dicono al client di usare i resolver OpenDNS gratuiti indicati dagli IP.

Ora, per fare in modo di instradare correttamente il traffico attraverso la VPN, è necessario consentire l'ip forwarding modificando il file [/etc/sysctl.conf](#) e attivo nella sessione corrente

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

```
cogotti-giulia@cgt-server:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
```

Questo non basta, è necessario modificare alcune impostazioni del firewall nel file [/etc/ufw/before.rules](#) impostando la politica predefinita per la catena POSTROUTING nella tabella NAT, mascherando tutto il traffico proveniente dalla VPN.

```
# START OPENVPN RULES
# NAT table rules
*nat
:POSTROUTING ACCEPT [0:0]
# Allow traffic from OpenVPN client to enp0s3
-A POSTROUTING -s 10.8.0.0/8 -o enp0s3 -j MASQUERADE
COMMIT
# END OPENVPN RULES
```

Inoltre in [/etc/default/ufw](#) imposto la politica di default per la catena di FORWARD, consentendo tutto.

```
# Set the default forward policy to ACCEPT, DROP or REJECT. Please note that
# if you change this you will most likely want to adjust your rules
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Va abilitato il firewall per consentire il traffico UDP alla porta specificata e abilito anche SSH. Infine carico le modifiche fatte riavviando il firewall.

```
cogotti-giulia@cgt-server:~$ sudo ufw allow 4434/udp
Rule updated
Rule updated (v6)
```

```
cogotti-giulia@cgt-server:~$ sudo ufw allow OpenSSH
Rules updated
Rules updated (v6)
cogotti-giulia@cgt-server:~$ sudo ufw disable
Firewall stopped and disabled on system startup
cogotti-giulia@cgt-server:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Ora, se tutto è stato configurato correttamente, è possibile avviare il servizio

```
cogotti-giulia@cgt-server:/etc/openvpn/server$ sudo systemctl -f enable openvpn-server@server.service
Created symlink /etc/systemd/system/multi-user.target.wants/openvpn-server@server.service → /lib/systemd/system/openvpn-server@.service.
cogotti-giulia@cgt-server:/etc/openvpn/server$ sudo systemctl start openvpn-server@server.service
cogotti-giulia@cgt-server:/etc/openvpn/server$ sudo systemctl status openvpn-server@server.service
● openvpn-server@server.service - OpenVPN service for server
   Loaded: loaded (/lib/systemd/system/openvpn-server@.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2023-04-30 22:48:50 CEST; 14min ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
   Main PID: 10523 (openvpn)
   Status: "Initialization Sequence Completed"
     Tasks: 1 (limit: 2234)
    Memory: 1.8M
       CPU: 39ms
   CGroup: /system.slice/system-openvpn\x2dserver.slice/openvpn-server@server.service
           └─10523 /usr/sbin/openvpn --status /run/openvpn-server/status-server.log --status-verb

Apr 30 22:48:50 cgt-server openvpn[10523]: Could not determine IPv4/IPv6 protocol. Using AF_INET
Apr 30 22:48:50 cgt-server openvpn[10523]: Socket Buffers: R=[212992->212992] S=[212992->212992]
Apr 30 22:48:50 cgt-server openvpn[10523]: UDPv4 link local (bound): [AF_INET][undef]:443
Apr 30 22:48:50 cgt-server openvpn[10523]: UDPv4 link remote: [AF_UNSPEC]
Apr 30 22:48:50 cgt-server openvpn[10523]: GID set to nogroup
Apr 30 22:48:50 cgt-server openvpn[10523]: UID set to nobody
Apr 30 22:48:50 cgt-server openvpn[10523]: MULTI: multi_init called, r=256 v=256
Apr 30 22:48:50 cgt-server openvpn[10523]: IFCONFIG POOL IPv4: base=10.8.0.4 size=62
Apr 30 22:48:50 cgt-server openvpn[10523]: IFCONFIG POOL LIST
Apr 30 22:48:50 cgt-server openvpn[10523]: Initialization Sequence Completed
```

Output di ifconfig, vediamo che è stata correttamente aggiunta l'interfaccia *tun0*

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.8.0.1 netmask 255.255.255.255 destination 10.8.0.2
    inet6 fe80::4865:a228:7ce6:ee3d prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 9 bytes 432 (432.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2.6. Configurazione client OpenVPN

Ogni client deve avere la propria configurazione e ognuno deve essere allineato con le impostazioni presenti nei file di configurazione del server. Per semplificare il processo creerò uno *script* che permetta di effettuare questa operazione su tutti i client partendo da un file di base.

Aggiungo una cartella files all'interno del server OpenVPN per contenere tutti i file di configurazione dei client e copio in *client-configs* la configurazione di esempio

```
cogotti-giulia@cgt-server:/$ mkdir -p ~/client-configs/files
cogotti-giulia@cgt-server:/$ cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf ~/cli
ent-configs/base.conf
```


Imposto lo stesso protocollo del server e nella direttiva remote inserisco *ip/hostname* del server e *porta* del server

```
# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
;proto tcp
proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 192.168.43.147 4434
;remote my-server-2 1194
```

Anche in questo caso user e group saranno *nobody* e *nogroup*

```
# Downgrade privileges after initialization (non-Windows only)
user nobody
group nogroup
```

Inserisco lo stesso cifrario e algoritmo usati dal server!

```
# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
# Note that v2.4 client/server will automatically
# negotiate AES-256-GCM in TLS mode.
# See also the data-ciphers option in the manpage
;cipher AES-256-CBC
cipher AES-256-GCM
auth SHA256
```

Commento le righe relative ai certificati, in quanto saranno aggiunte automaticamente all'esecuzione dello script

```
# SSL/TLS parms.
# See the server config file for more
# description. It's best to use
# a separate .crt/.key file pair
# for each client. A single ca
# file can be used for all clients.
;ca ca.crt
;cert client.crt
;key client.key
```

```
# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1
```

Imposto a 1 il *key-direction* affinché la VPN funzioni correttamente sul client

```
# Set this to 1 for VPN to
# function correctly on the client machine
key-direction 1
```


Infine aggiungo alcune righe commentate per gestire i metodi che i client Linux usano per la risoluzione DNS. Il primo set è relativo ai client che usano *resolvconf*, il secondo per quelli che utilizzano *systemd-resolved*

```
# For client that do NOT use systemd-resolved
# to manage DNS
# These clients rely on the resolvconf utility
# to update DNS information for Linux clients
; script-security 2
; up /etc/ovpn/update-resolv-conf
; down /etc/ovpn/update-resolv-conf

# For clients that use systemd-resolved
# for DNS resolution
; script-security 2
; up /etc/ovpn/update-systemd-resolved
; down /etc/ovpn/update-systemd-resolved
; down-pre
; dhcp-option DOMAIN-ROUTE .
```

A questo punto creo lo script che crea una copia del file *base.conf*, raccoglie tutti i file di certificato e di chiave per il client e ne estrae il contenuto. Aggiunge questi file alla configurazione e la esporta in un nuovo file *nome.ovpn* di configurazione situato nella cartella *files* precedentemente creata. Per aggiungere un client è necessario semplicemente eseguire lo script.

```
GNU nano 6.2 /home/cogotti-giulia/client-configs/make_config.sh *
#!/bin/bash

# First argument: Client identifier

KEY_DIR=~/.client-configs/keys
OUTPUT_DIR=~/.client-configs/files
BASE_CONFIG=~/.client-configs/base.conf

cat ${BASE_CONFIG} \
  <(echo -e '<ca>' \
    ${KEY_DIR}/ca.crt \
  <(echo -e '</ca>\n<cert>' \
    ${KEY_DIR}/${1}.crt \
  <(echo -e '</cert>\n<key>' \
    ${KEY_DIR}/${1}.key \
  <(echo -e '</key>\n<tls-crypt>' \
    ${KEY_DIR}/ta.key \
  <(echo -e '</tls-crypt>' \
  > ${OUTPUT_DIR}/${1}.ovpn
```

Do i permessi di esecuzione

```
cogotti-giulia@cgt-server:~$ nano ~/.client-configs/make_config.sh
cogotti-giulia@cgt-server:~$ chmod 700 ~/.client-configs/make_config.sh
```

Esecuzione dello script per il client, che creerà un file chiamato *client1.ovpn*, nome del file preso dal parametro in input

```
cogotti-giulia@cgt-server:~/.client-configs$ ./make_config.sh client1
cogotti-giulia@cgt-server:~/.client-configs$ ls ~/.client-configs/files
client1.ovpn
```

Trasferisco il file al client (in questo caso un'altra VM) tramite [SFTP](#) (metodo sicuro per il trasferimento che usa la crittografia SSH)

```
cogotti-giulia@cgt-client:~$ sftp cogotti-giulia@192.168.43.147:client-configs/files/client1.ovpn ~/
cogotti-giulia@192.168.43.147's password:
Connected to 192.168.43.147.
Fetching /home/cogotti-giulia/client-configs/files/client1.ovpn to /home/cogotti-giulia/client1.ovpn
client1.ovpn                                100% 12KB 8.0MB/s 00:00
cogotti-giulia@cgt-client:~$ ls -l | grep client1
-rw-rw-r-- 1 cogotti-giulia cogotti-giulia 12313 May  1 11:23 client1.ovpn
```

Verifico quale metodo usa il client per la risoluzione DNS, in questo caso usa resolv-conf

```
cogotti-giulia@cgt-client:~$ ls /etc/openvpn
client  server  update-resolv-conf
```

Quindi nella configurazione [client1.ovpn](#) tolgo i commenti alle seguenti righe

```
# For client that do NOT use systemd-resolved
# to manage DNS
# These clients rely on the resolvconf utility
# to update DNS information for Linux clients
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
```

3. Testing

3.1. Test 1: VM server e client nella stessa LAN

Per effettuare il collegamento del client alla VPN occorre digitare il comando `sudo openvpn --config client1.ovpn`

```
cogotti-giulia@cgt-client:~$ sudo openvpn --config client1.ovpn &
[1] 2660
cogotti-giulia@cgt-client:~$ 2023-05-01 14:11:02 OpenVPN 2.5.5 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EP
OLL] [PKCS11] [MH/PKTINFO] [AEAD] built on Jul 14 2022
2023-05-01 14:11:02 library versions: OpenSSL 3.0.2 15 Mar 2022, LZO 2.10
2023-05-01 14:11:02 NOTE: the current --script-security setting may allow this configuration to call user-defined
scripts
2023-05-01 14:11:02 Outgoing Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
2023-05-01 14:11:02 Outgoing Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authenticati
on
2023-05-01 14:11:02 Incoming Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
2023-05-01 14:11:02 Incoming Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authenticati
on

2023-05-01 14:11:02 TCP/UDP: Preserving recently used remote address: [AF_INET]192.168.43.147:443
2023-05-01 14:11:02 Socket Buffers: R=[212992->212992] S=[212992->212992]
2023-05-01 14:11:02 UDP link local: (not bound)
2023-05-01 14:11:02 UDP link remote: [AF_INET]192.168.43.147:443
2023-05-01 14:11:02 NOTE: UID/GID downgrade will be delayed because of --client, --pull, or --up-delay
2023-05-01 14:11:02 TLS: Initial packet from [AF_INET]192.168.43.147:443, sid=fc67e211 2d6845c4
2023-05-01 14:11:02 VERIFY OK: depth=1, CN=certificate-auth
2023-05-01 14:11:02 VERIFY KU OK
2023-05-01 14:11:02 Validating certificate extended key usage
2023-05-01 14:11:02 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentica
tion
2023-05-01 14:11:02 VERIFY EKU OK
2023-05-01 14:11:02 VERIFY OK: depth=0, CN=server
2023-05-01 14:11:02 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bit RS
A, signature: RSA-SHA256

2023-05-01 14:11:02 [server] Peer Connection Initiated with [AF_INET]192.168.43.147:443
2023-05-01 14:11:02 PUSH: Received control message: 'PUSH_REPLY,route 10.8.0.1,topology net30,ping 10,ping-restart
120,ifconfig 10.8.0.6 10.8.0.5,peer-id 1,cipher AES-256-GCM'
2023-05-01 14:11:02 OPTIONS IMPORT: timers and/or timeouts modified
2023-05-01 14:11:02 OPTIONS IMPORT: --ifconfig/up options modified
2023-05-01 14:11:02 OPTIONS IMPORT: route options modified
2023-05-01 14:11:02 OPTIONS IMPORT: peer-id set
2023-05-01 14:11:02 OPTIONS IMPORT: adjusting link_mtu to 1624
2023-05-01 14:11:02 OPTIONS IMPORT: data channel crypto options modified
2023-05-01 14:11:02 Outgoing Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
2023-05-01 14:11:02 Incoming Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
2023-05-01 14:11:02 net_route_v4_best_gw query: dst 0.0.0.0
2023-05-01 14:11:02 net_route_v4_best_gw result: via 192.168.43.7 dev enp0s3
2023-05-01 14:11:02 ROUTE_GATEWAY 192.168.43.7/255.255.255.0 IFACE=enp0s3 HWADDR=08:00:27:b4:9c:b7
2023-05-01 14:11:02 TUN/TAP device tun0 opened
2023-05-01 14:11:02 net_iface_mtu_set: mtu 1500 for tun0
2023-05-01 14:11:02 net_iface_up: set tun0 up
2023-05-01 14:11:02 net_addr_ptp_v4_add: 10.8.0.6 peer 10.8.0.5 dev tun0
2023-05-01 14:11:02 /etc/openvpn/update-resolv-conf tun0 1500 1624 10.8.0.6 10.8.0.5 init
2023-05-01 14:11:02 net_route_v4_add: 10.8.0.1/32 via 10.8.0.5 dev [NULL] table 0 metric -1
2023-05-01 14:11:02 GID set to nogroup
2023-05-01 14:11:02 UID set to nobody
2023-05-01 14:11:02 WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to pr
event this
2023-05-01 14:11:02 Initialization Sequence Completed
```

Come si può notare, dopo lo scambio di chiavi e validazione dei certificati, viene aperto il tunnel nell'interfaccia `tun0`, assegnato l'ip `10.8.0.6` al client e impostato il

corretto forwarding dei pacchetti. Infine la configurazione termina senza errori. Tramite il comando *ifconfig* verifico la presenza della scheda di rete

```
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.8.0.6 netmask 255.255.255.255 destination 10.8.0.5
    inet6 fe80::de18:9cd2:29cc:d780 prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

In contemporanea il server faceva le seguenti operazioni: (output del comando *systemctl status openvpn-server@server.service*)

```
192.168.43.54:50084 peer info: IV_TCPNL=1
192.168.43.54:50084 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bit RSA, signature: RSA-SHA256
192.168.43.54:50084 [client1] Peer Connection Initiated with [AF_INET]192.168.43.54:50084
MULTI: new connection by client 'client1' will cause previous active sessions by this client to be dropped. Remember to use the --duplicate-cn option if you w
MULTI_sva: pool returned IPv4=10.8.0.6, IPv6=(Not enabled)
MULTI: Learn: 10.8.0.6 -> client1/192.168.43.54:50084
MULTI: primary virtual IP for client1/192.168.43.54:50084: 10.8.0.6
Outgoing Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
Incoming Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
SENT CONTROL [client1]: 'PUSH_REPLY,route 10.8.0.1,topology net30,ping 10,ping-restart 120,ifconfig 10.8.0.6 10.8.0.5,peer-id 1,cipher AES-256-GCM' (status=1)
```

Per verificare che il collegamento funzioni effettuo il ping da server a client e viceversa

```
cogotti-giulia@cgt-server:~$ ping 10.8.0.6
PING 10.8.0.6 (10.8.0.6) 56(84) bytes of data.
64 bytes from 10.8.0.6: icmp_seq=1 ttl=64 time=2.16 ms
64 bytes from 10.8.0.6: icmp_seq=2 ttl=64 time=1.35 ms
64 bytes from 10.8.0.6: icmp_seq=3 ttl=64 time=1.51 ms
64 bytes from 10.8.0.6: icmp_seq=4 ttl=64 time=1.59 ms
64 bytes from 10.8.0.6: icmp_seq=5 ttl=64 time=1.50 ms
^C
--- 10.8.0.6 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 1.350/1.620/2.156/0.279 ms
```

```
cogotti-giulia@cgt-client:~$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=1.65 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=1.53 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=1.53 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=1.62 ms
64 bytes from 10.8.0.1: icmp_seq=5 ttl=64 time=1.52 ms
^C
--- 10.8.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 1.523/1.570/1.649/0.054 ms
```

Sembra funzionare! 🥰



3.2. Test 2: VM server isolata e il mio pc come client

Ho deciso di effettuare un test impostando la VM del server con scheda di rete connessa a **NAT** e impostando di conseguenza il **port forwarding** (in modo che fosse accessibile dall'esterno, quindi dal mio pc).

```
cogotti-giulia@cgt-server:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe76:44aa prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:76:44:aa txqueuelen 1000 (Ethernet)
    RX packets 7457 bytes 8525549 (8.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 3733 bytes 296203 (296.2 KB)
    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 80 bytes 8720 (8.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 80 bytes 8720 (8.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.8.0.1 netmask 255.255.255.255 destination 10.8.0.2
    inet6 fe80::6c63:ef9c:397b:4623 prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 7 bytes 588 (588.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 16 bytes 1020 (1.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Nome	Protocollo	IP dell'host	Porta dell'host	IP del guest	Porta del guest	
OpenVPN	UDP	127.0.0.1	4434	10.0.2.15	4434	
SSH	TCP	127.0.0.1	2522	10.0.2.15	22	

a: NAT

e:

Di conseguenza della configurazione dal lato client ho modificato il file *client1.ovpn* inserendo il nuovo ip della VM

```
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 127.0.0.1 4434
```

Collegamento al server tramite client (mio pc)

```

2023-05-01 17:09:27 [server] Peer Connection Initiated with [AF_INET]127.0.0.1:4434
2023-05-01 17:09:27 TLS: move_session: dest=TM_ACTIVE src=TM_INITIAL reinit_src=1
2023-05-01 17:09:27 TLS: tls_multi_process: initial untrusted session promoted to trusted
2023-05-01 17:09:27 PUSH: Received control message: 'PUSH_REPLY,route 10.8.0.1,topology net30,ping 10,ping-restart 120,ifconfig 10.8.0.6 10.8.0.5,peer-id 1,cipher AES-256-GCM'
2023-05-01 17:09:27 OPTIONS IMPORT: --ifconfig/up options modified
2023-05-01 17:09:27 OPTIONS IMPORT: route options modified
2023-05-01 17:09:27 net_route_v4_best_gw query: dst 0.0.0.0
2023-05-01 17:09:27 net_route_v4_best_gw result: via 192.168.193.144 dev wlan0
2023-05-01 17:09:27 ROUTE_GATEWAY 192.168.193.144/255.255.255.0 IFACE=wlan0 HWADDR=68:07:15:8c:fa:64
2023-05-01 17:09:27 TUN/TAP device tun0 opened
2023-05-01 17:09:27 net_iface_mtu_set: mtu 1500 for tun0
2023-05-01 17:09:27 net_iface_up: set tun0 up
2023-05-01 17:09:27 net_addr_ptp_v4_add: 10.8.0.6 peer 10.8.0.5 dev tun0
2023-05-01 17:09:27 net_route_v4_add: 10.8.0.1/32 via 10.8.0.5 dev [NULL] table 0 metric -1
2023-05-01 17:09:27 UID set to nobody
2023-05-01 17:09:27 GID set to nobody
2023-05-01 17:09:27 Capabilities retained: CAP_NET_ADMIN
2023-05-01 17:09:27 Initialization Sequence Completed
2023-05-01 17:09:27 Data Channel: cipher 'AES-256-GCM', peer-id: 1
2023-05-01 17:09:27 Timers: ping 10, ping-restart 120

```

Su client verifico la presenza della nuova scheda di rete tramite *ifconfig*

```

tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST> mtu 1500
    inet 10.8.0.6 netmask 255.255.255.255 destination 10.8.0.5
    inet6 fe80::f5d1:2860:1f82:683d prefixlen 64 scopeid 0x20<link>
    unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 txqueuelen 500 (UNSPEC)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1 bytes 48 (48.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

Testo il collegamento attraverso il tunnel inviando un *ping* da server a client e viceversa

```

cogotti-giulia@cgt-server:~$ ping 10.8.0.6
PING 10.8.0.6 (10.8.0.6) 56(84) bytes of data.
64 bytes from 10.8.0.6: icmp_seq=1 ttl=64 time=1.28 ms
64 bytes from 10.8.0.6: icmp_seq=2 ttl=64 time=1.40 ms
64 bytes from 10.8.0.6: icmp_seq=3 ttl=64 time=1.36 ms
64 bytes from 10.8.0.6: icmp_seq=4 ttl=64 time=1.60 ms
64 bytes from 10.8.0.6: icmp_seq=5 ttl=64 time=1.36 ms
^C
--- 10.8.0.6 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 1.284/1.401/1.599/0.105 ms

```

```

[cogotti-giulia@wabisabi ~]$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=0.573 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=1.34 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=1.32 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=1.38 ms
64 bytes from 10.8.0.1: icmp_seq=5 ttl=64 time=1.55 ms
^C
--- 10.8.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4013ms
rtt min/avg/max/mdev = 0.573/1.233/1.554/0.340 ms

```

Anche in questo caso sembra che funzionino! 🙄

3.3. Test 3: reindirizzamento di tutto il traffico sulla VPN

Come accennato in precedenza, è possibile forzare il passaggio del traffico del client attraverso la VPN. Questo significa che le richieste devono arrivare al server openvpn e poi andare verso internet.

Inizialmente ho provato avendo server OpenVPN e client sulla stessa LAN.

Prima di modificare la configurazione ho verificato il *gateway del client* durante la connessione al server, tramite il comando `netstat -rn`, si può notare che il default gateway è quello relativo alla rete wifi alla quale il mio pc è collegato

```
[cogotti-giulia@wabisabi ~]$ netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          192.168.36.213  0.0.0.0         UG        0 0          0 wlan0
10.8.0.1         10.8.0.5        255.255.255.255 UGH        0 0          0 tun0
10.8.0.5         0.0.0.0         255.255.255.255 UH        0 0          0 tun0
```

Nel file `server.conf` all'interno del server OpenVPN ho aggiunto le seguenti righe, la prima riga `push "redirect-gateway local def1 bypass-dhcp"` forza i client a reindirizzare il loro traffico di rete attraverso la VPN. Le altre due righe indicano ai client quali DNS usare come riferimento.

- `redirect-gateway`: esegue automaticamente i comandi di routing per reindirizzare tutto il traffico IP in uscita sulla VPN
- `local`: opzione per quando client e server si trovano all'interno della sottorete
- `def1`: ignora il default gateway che esisteva prima di attivare la vpn, fa in modo non venga cancellato ma sovrascritto utilizzando 0.0.0.0/1 e 128.0.0.0/1 invece di 0.0.0.0/0
- `bypass-dhcp`: aggiunge un percorso diretto al server DHCP (se non è locale) che bypassa il tunnel

ATTENZIONE: `redirect-gateway` è un'opzione lato client, infatti il server la invia (fa una push) al client quando esso si collega!


```
# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# or bridge the TUN/TAP interface to the internet
# in order for this to work properly).
push "redirect-gateway local def1 bypass-dhcp"

# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses.  CAVEAT:
# http://openvpn.net/faq.html#dhcpcaveats
# The addresses below refer to the public
# DNS servers provided by opendns.com.
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
```

Nel client ho inserito le seguenti righe nel file *client1.ovpn* per aggiornare i DNS

```
# This updates the resolvconf with dns settings
setenv PATH /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
down-pre
```

Mi sono collegata al server tramite il solito comando *sudo openvpn --config client1.ovpn*

```
2023-05-03 15:49:02 PUSH: Received control message: 'PUSH_REPLY,redirect-gateway local def1,dhcp-option DNS 208.67.222.222,dhcp-option DNS 208.67.220.220,route 10.8.0.1,topology net30,ping 10,ping-restart 120,ifconfig 10.8.0.6 10.8.0.5,peer-id 0,cipher AES-256-GCM'
2023-05-03 15:49:02 OPTIONS IMPORT: --ifconfig/up options modified
2023-05-03 15:49:02 OPTIONS IMPORT: route options modified
2023-05-03 15:49:02 OPTIONS IMPORT: --ip-win32 and/or --dhcp-option options modified
2023-05-03 15:49:02 net_route_v4_best_gw query: dst 0.0.0.0
2023-05-03 15:49:02 net_route_v4_best_gw result: via 192.168.98.163 dev wlan0
2023-05-03 15:49:02 ROUTE_GATEWAY 192.168.98.163/255.255.255.0 IFACE=wlan0 HWADDR=68:07:15:8c:fa:64
2023-05-03 15:49:02 TUN/TAP device tun0 opened
2023-05-03 15:49:02 net_iface_mtu_set: mtu 1500 for tun0
2023-05-03 15:49:02 net_iface_up: set tun0 up
2023-05-03 15:49:02 net_addr_ptp_v4_add: 10.8.0.6 peer 10.8.0.5 dev tun0
2023-05-03 15:49:02 /etc/openvpn/update-resolv-conf tun0 1500 0 10.8.0.6 10.8.0.5 init
dhcp-option DNS 208.67.222.222
dhcp-option DNS 208.67.220.220
IF_DNS_NAMESERVERS= 208.67.222.222 208.67.220.220
IF_DNS_SEARCH=
--set-dns=208.67.222.222 --set-dns=208.67.220.220

2023-05-03 15:49:02 net_route_v4_add: 0.0.0.0/1 via 10.8.0.5 dev [NULL] table 0 metric -1
2023-05-03 15:49:02 net_route_v4_add: 128.0.0.0/1 via 10.8.0.5 dev [NULL] table 0 metric -1
2023-05-03 15:49:02 net_route_v4_add: 10.8.0.1/32 via 10.8.0.5 dev [NULL] table 0 metric -1
```

A questo punto, sul client, ho eseguito di nuovo il comando *netstat -rn* ottenendo ciò che volevo, ovvero che il default gateway è stato sovrascritto da quello usato dalla VPN!







```
[cogotti-giulia@wabisabi ~]$ netstat -rn
Kernel IP routing table
Destination        Gateway            Genmask           Flags     MSS Window  irtt Iface
0.0.0.0            10.8.0.5          128.0.0.0         UG        0 0        0 tun0
0.0.0.0            192.168.36.213    0.0.0.0           UG        0 0        0 wlan0
10.8.0.1           10.8.0.5          255.255.255.255   UGH       0 0        0 tun0
10.8.0.5           0.0.0.0           255.255.255.255   UH        0 0        0 tun0
128.0.0.0          10.8.0.5          128.0.0.0         UG        0 0        0 tun0
```

Il collegamento ad internet funziona, inoltre, sul sito [dns leak test](#), vedo che il traffico passa attraverso i server impostati nella configurazione quindi sembra che tutto sia andato bene.

DNS leak test senza VPN

Test complete			
Query round 1	Progress...	Servers found 3	
IP	Hostname	ISP	Country
83.158.4.18	None	Iliad Italia	Italy 
83.158.4.27	None	Iliad Italia	Italy 
83.158.5.32	Ins-bzn-02-83-158-5-32.adsl.proxad.net.	Iliad Italia	Italy 

DNS leak test con VPN

Test complete			
Query round 1	Progress...	Servers found 5	
IP	Hostname	ISP	Country
146.112.136.70	r7.compute.mil1.edc.strln.net.	Cisco OpenDNS, LLC	Milan, Italy 
146.112.136.72	r9.compute.mil1.edc.strln.net.	Cisco OpenDNS, LLC	Milan, Italy 
83.158.4.18	None	Iliad Italia	Italy 
83.158.4.27	None	Iliad Italia	Italy 
83.158.5.32	Ins-bzn-02-83-158-5-32.adsl.proxad.net.	Iliad Italia	Italy 

4. Riferimenti

[Tutorial configurazione OpenVPN](#)

[Openvpn Man Page](#)

[Errore da non fare configurazione CA](#)

Tantissime altre pagine su internet 📖