Responsable du cours (ARIR81/M1-APP-RS) : Yousfi Souheib

TP 5 : OpenVPN : De la Configuration de Base à la Sécurité Avancée avec PKI

Introduction

Ce TP vous guide à travers l'exploration des réseaux privés virtuels (VPN) via OpenVPN, un outil open-source conçu pour la sécurisation des communications réseau. Le TP se décompose en deux parties principales :

La première partie vous immerge dans l'environnement existant d'OpenVPN, vous permettant de vous familiariser avec le PKI (Public Key Infrastructure) déjà configuré sur votre système Linux.

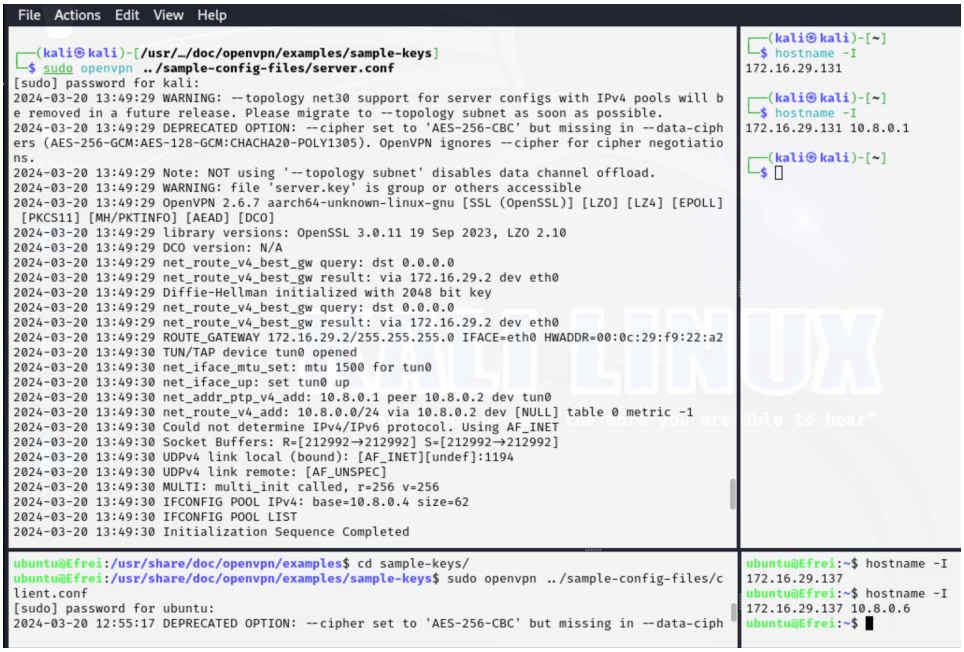
Dans la seconde partie, vous progresserez à travers trois étapes distinctes : vous commencerez par examiner une configuration non sécurisée afin de saisir les principes de base. Puis, vous évoluerez vers une configuration sécurisée, en implémentant le partage de clés entre le serveur et le client pour améliorer la sûreté des communications. Enfin, vous serez confronté à l'exigence de développer votre propre PKI, un challenge stimulant qui garantira une communication sécurisée entre le serveur et le client de manière autonome.

Configuration existante :

Dans cette section du TP, nous explorerons la configuration existante d'OpenVPN, en utilisant la PKI déjà en place sur votre système Linux. Cela inclut l'analyse des configurations serveur (server.conf) et client (client.conf) situées dans /usr/share/doc/openvpn/examples/sample-config-files, ainsi que les clés et les certificats dans /usr/share/doc/openvpn/examples/sample-keys. Notre but est de comprendre comment OpenVPN gère les clés, les certificats et les configurations réseau, et de montrer comment utiliser la PKI existante pour sécuriser une connexion VPN.

Commencez par insérer l'adresse de votre serveur dans client.conf pour une première connexion opérationnelle. Cette étape pose les bases pour les configurations avancées à venir.

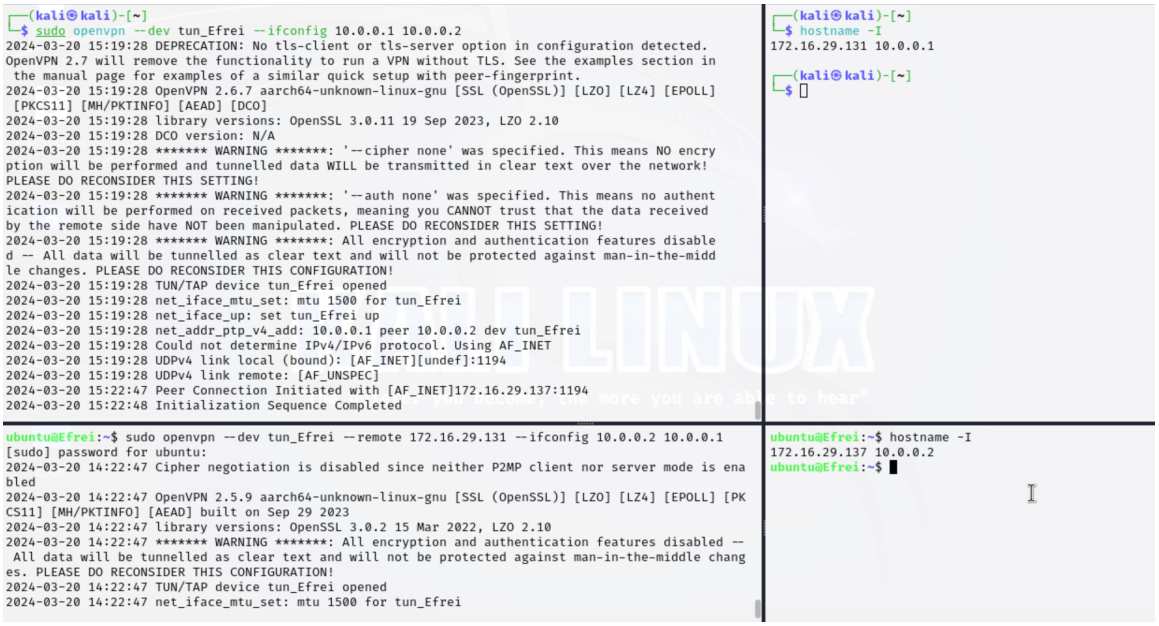
Par la suite, nous procédons à l'établissement du tunnel VPN entre le serveur et le client, tout en vérifiant les adresses IP nouvellement attribuées à notre interface réseau pour assurer une connexion réussie.



Configuration personnelle :

Dans cette partie, vous débuterez avec une configuration basique d'OpenVPN pour comprendre les fondements. Progressivement, vous passerez à une configuration sécurisée par le partage de clés entre serveur et client, renforçant la protection des échanges. Enfin, vous relèverez le défi de créer votre propre PKI, assurant une communication sécurisée et autonome entre le serveur et le client.

Tunnel non sécurisé :

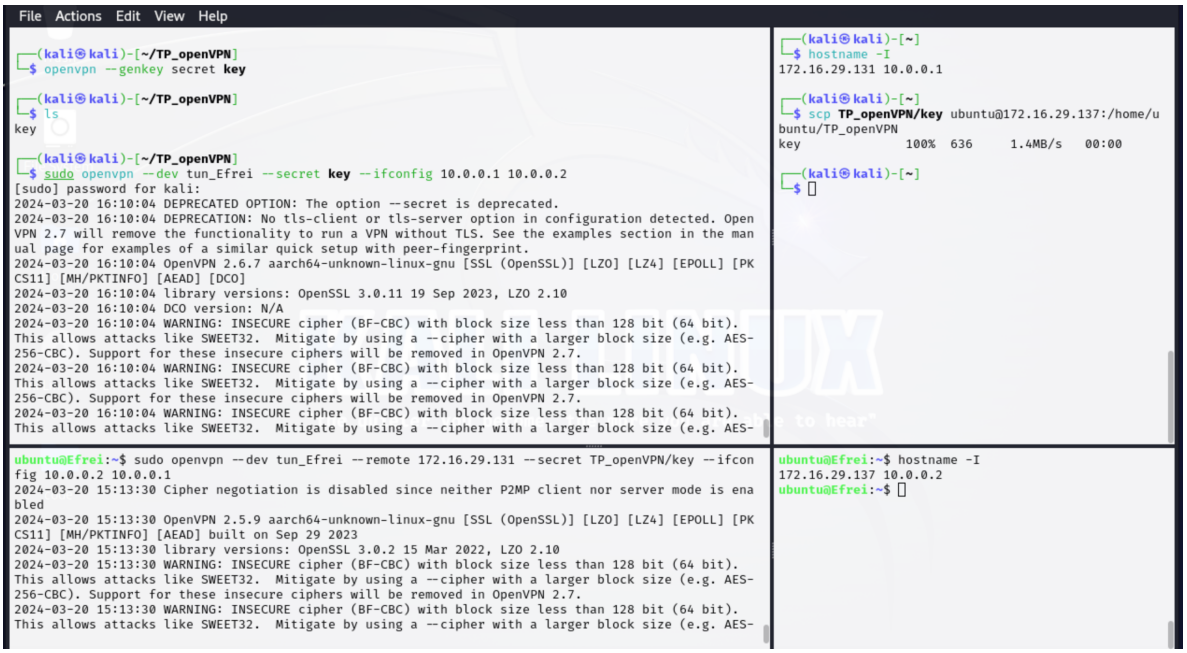


Dans votre rapport, décrivez les failles de sécurité identifiées.

Tunnel sécurisé avec partage de clé :

Dans cette sous section, nous allons établir un tunnel VPN sécurisé en utilisant une clé partagée entre le serveur et le client. Cette méthode renforce la sécurité de notre communication

en s'assurant que seules les entités possédant la clé partagée peuvent participer à l'échange de données.
Nous débuterons par la génération et la distribution sécurisée de la clé partagée, avant de lancer le tunnel coté serveur et coté client en s'appuyant sur cette cette clé partagée.



Optimisez la configuration en sélectionnant le protocole AES-256-CBC au lieu de BF-CBC, et documentez les vulnérabilités de sécurité observées dans votre rapport.

Tunnel sécurisé avec apport de PKI :

Dans cette partie recommandée du TP, nous allons nous concentrer sur la création de notre propre infrastructure à clés publiques (PKI), ainsi que sur la génération des clés et des certificats nécessaires, en utilisant un outil dénommé easy-rsa.

Après avoir préparé votre environnement de travail, ouvrez le fichier easyrsa et prenez le temps d'en explorer le contenu afin de saisir les principes fondamentaux qu'il contient.

```
USAGE: easyrsa [global-options] COMMAND [command-options]

To get detailed usage and help for a command, use:
  ./easyrsa help COMMAND

For a list of global-options, use:
  ./easyrsa help options

For a list of extra test commands, use:
  ./easyrsa help more

A list of commands is shown below:
  init-pki [ cmd-opts ]
  build-ca [ cmd-opts ]
  gen-dh
  gen-req <file_name_base> [ cmd-opts ]
  sign-req <type> <file_name_base> [ cmd-opts ]
  build-client-full <file_name_base> [ cmd-opts ]
  build-server-full <file_name_base> [ cmd-opts ]
  build-serverClient-full <file_name_base> [ cmd-opts ]
  inline <file_name_base>
  revoke <file_name_base> [ cmd-opts ]
  renew <file_name_base>
  revoke-renewed <file_name_base> [ cmd-opts ]
  rewind-renew <certificate_serial_number>
  rebuild <file_name_base> [ cmd-opts ]
  gen-crl
  update-db
  show-req <file_name_base> [ cmd-opts ]
  show-cert <file_name_base> [ cmd-opts ]
  show-ca [ cmd-opts ]
  show-crl
```

Nous débutons par la création de notre infrastructure à clés publiques (PKI), lequel comprendra les clés, les demandes de certificats et les certificats eux-mêmes.

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa init-pki
[sudo] password for kali:

Notice
------
'init-pki' complete; you may now create a CA or requests.

Your newly created PKI dir is:
* /etc/openvpn/server/pki

Using Easy-RSA configuration:
* undefined

┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ tree
.
├── easy-rsa
│   ├── easyrsa
│   ├── openssl-easyrsa.cnf
│   ├── vars.example
│   └── x509-types
│       ├── COMMON
│       ├── ca
│       ├── client
│       ├── code-signing
│       ├── email
│       ├── kdc
│       ├── server
│       └── serverClient
└── pki  [error opening dir]

4 directories, 11 files
```

Par la suite, nous procéderons à la création de notre autorité de certification, en générant sa clé privée (ca.key) dans le dossier ”private” et son certificat (ca.crt), ainsi que d'autres éléments nécessaires, dans le répertoire approprié.

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa build-ca
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)

Enter New CA Key Passphrase:

Confirm 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]:autorite

Notice
------
CA creation complete. Your new CA certificate is at:
* /etc/openvpn/server/pki/ca.crt
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 2 pki/
pki/
├── ca.crt
├── certs_by_serial
├── index.txt
├── index.txt.attr
├── inline
├── issued
├── openssl-easyrsa.cnf
├── private
│   └── ca.key
├── reqs
├── revoked
│   ├── certs_by_serial
│   ├── private_by_serial
│   └── reqs_by_serial
└── serial

10 directories, 6 files
```

Après avoir établi l'autorité de certification, nous passerons à la création de la clé et de la demande de certificat pour le serveur Efrei ainsi que pour le client. Il est essentiel que notre autorité de certification signe ces demandes. À chaque étape, il est recommandé de vérifier les nouveaux fichiers créés à l'aide de la commande tree pour une meilleure compréhension du processus.

Pour le serveur Efrei :

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa gen-req Efrei nopass
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)
.....+.+.....+......+...+.++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++*.....+..+...+......+.......+.....................................+........+.......+.....
....+....+.....................+.....+....+..+..............+....+..+....+......+..
............+...+.+..+.......+..+.............+...+..+...+...+....+..+....+......+..
.....................................+.............+.+........+.+.............+......+..
.+.+.....+......+..........+.....+..............+.+..............+..+.+...........
+++
...........+....+..++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++*......+..+....+...+...................+..+...+...+.+...+...+....
...........+..............+.+......+.....................+.+...+............+.....+.
++++++++++++++++++++++++++++++++
-----
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) [Efrei]:

Notice
------
Private-Key and Public-Certificate-Request files created.
Your files are:
* req: /etc/openvpn/server/pki/reqs/Efrei.req
* key: /etc/openvpn/server/pki/private/Efrei.key
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 3 pki/
pki/
├── ca.crt
├── certs_by_serial
├── index.txt
├── index.txt.attr
├── inline
├── issued
├── openssl-easyrsa.cnf
├── private
│   ├── Efrei.key
│   └── ca.key
├── reqs
│   └── Efrei.req
├── revoked
│   ├── certs_by_serial
│   ├── private_by_serial
│   └── reqs_by_serial
└── serial

10 directories, 8 files
```

Puis, l'autorité signe la demande de certificat de Efrei :

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa sign-req server Efrei
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)
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                = Efrei

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

Using configuration from /etc/openvpn/server/pki/openssl-easyrsa.cnf
Enter pass phrase for /etc/openvpn/server/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:'Efrei'
Certificate is to be certified until Jun 23 18:43:35 2026 GMT (825 days)

Write out database with 1 new entries
Database updated

Notice
------
Certificate created at:
* /etc/openvpn/server/pki/issued/Efrei.crt
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 3 pki/
pki/
├── ca.crt
├── certs_by_serial
│   └── E0250235D0CD920ACD097BB8B761D623.pem
├── index.txt
├── index.txt.attr
├── index.txt.attr.old
├── index.txt.old
├── inline
├── issued
│   └── Efrei.crt
├── openssl-easyrsa.cnf
├── private
│   ├── Efrei.key
│   └── ca.key
├── reqs
│   └── Efrei.req
├── revoked
│   ├── certs_by_serial
│   ├── private_by_serial
│   └── reqs_by_serial
├── serial
└── serial.old

10 directories, 13 files
```

Pour le client :

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa gen-req Client_VPN nopass
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)
..+......+..+............+.+.........+................+...+.+..++++++++++++
.....++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.....
+......+......+.......+...+..+...+....+...+..+...+.....................+....+
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
..+.....+....+.....+...+............+.+......+.........+......++++++++++++++++
...++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+.....
+...+......+...........+......+....+.....+.+..+.........+..+.................
++++++++
-----
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) [Client_VPN]:

Notice
------
Private-Key and Public-Certificate-Request files created.
Your files are:
* req: /etc/openvpn/server/pki/reqs/Client_VPN.req
* key: /etc/openvpn/server/pki/private/Client_VPN.key
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 3 pki/
pki/
├── ca.crt
├── certs_by_serial
│   └── E0250235D0CD920ACD097BB8B761D623.pem
├── index.txt
├── index.txt.attr
├── index.txt.attr.old
├── index.txt.old
├── inline
├── issued
│   └── Efrei.crt
├── openssl-easyrsa.cnf
├── private
│   ├── Client_VPN.key
│   ├── Efrei.key
│   └── ca.key
├── reqs
│   ├── Client_VPN.req
│   └── Efrei.req
├── revoked
│   ├── certs_by_serial
│   ├── private_by_serial
│   └── reqs_by_serial
├── serial
└── serial.old

10 directories, 15 files
```

Puis, l'autorité signe la demande de certificat du client :

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa sign-req client Client_VPN
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)
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                = Client_VPN

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

Using configuration from /etc/openvpn/server/pki/openssl-easyrsa.cnf
Enter pass phrase for /etc/openvpn/server/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:'Client_VPN'
Certificate is to be certified until Jun 23 18:51:03 2026 GMT (825 days)

Write out database with 1 new entries
Database updated

Notice
------
Certificate created at:
* /etc/openvpn/server/pki/issued/Client_VPN.crt
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 3 pki/
pki/
├── ca.crt
├── certs_by_serial
│   ├── 73BDDA937CCABDCB043D3F24073BDCD8.pem
│   └── E0250235D0CD920ACD097BB8B761D623.pem
├── index.txt
├── index.txt.attr
├── index.txt.attr.old
├── index.txt.old
├── inline
├── issued
│   ├── Client_VPN.crt
│   └── Efrei.crt
├── openssl-easyrsa.cnf
├── private
│   ├── Client_VPN.key
│   ├── Efrei.key
│   └── ca.key
├── reqs
│   ├── Client_VPN.req
│   └── Efrei.req
├── revoked
│   ├── certs_by_serial
│   ├── private_by_serial
│   └── reqs_by_serial
├── serial
└── serial.old

10 directories, 17 files
```

Création de la clé Diffie Hellman et la clé d'authentification TLS TA.key :

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo ./easy-rsa/easyrsa gen-dh
No Easy-RSA 'vars' configuration file exists!

Using SSL:
* openssl OpenSSL 3.0.11 19 Sep 2023 (Library: OpenSSL 3.0.11 19 Sep 2023)
Generating DH parameters, 2048 bit long safe prime
.....................................................................+.....
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo openvpn --genkey secret ta.key
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ sudo tree -L 2
.
├── easy-rsa
│   ├── easyrsa
│   ├── openssl-easyrsa.cnf
│   ├── vars.example
│   └── x509-types
├── pki
│   ├── ca.crt
│   ├── certs_by_serial
│   ├── dh.pem
│   ├── index.txt
│   ├── index.txt.attr
│   ├── index.txt.attr.old
│   ├── index.txt.old
│   ├── inline
│   ├── issued
│   ├── openssl-easyrsa.cnf
│   ├── private
│   ├── reqs
│   ├── revoked
│   ├── serial
│   └── serial.old
└── ta.key

10 directories, 13 files
```

On envoie les fichiers concernés au client :

```
File  Actions  Edit  View  Help

┌──(root㉿kali)-[/etc/openvpn/server]
└─# scp pki/ca.crt pki/issued/Client_VPN.crt \
pki/private/Client_VPN.key ta.key ubuntu@172.16.29.137:/home/ubuntu/TP2
The authenticity of host '172.16.29.137 (172.16.29.137)' can't be established.
ED25519 key fingerprint is SHA256:xSu5OLNAUb5vcr6T3efP1Xpex+VjN3Y0TFDfObp2/bU.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.16.29.137' (ED25519) to the list of known hosts.
ubuntu@172.16.29.137's password:
ca.crt                                        100% 1192     2.0MB/s   00:00
Client_VPN.crt                                100% 4487     6.5MB/s   00:00
Client_VPN.key                                100% 1704     3.7MB/s   00:00
ta.key                                        100%  636     1.4MB/s   00:00

┌──(root㉿kali)-[/etc/openvpn/server]
└─#
```

```
ubuntu@Efrei:~$ ls TP2/
ca.crt  Client_VPN.crt  Client_VPN.key  ta.key
ubuntu@Efrei:~$
```

Nous procédons aux ajustements requis dans le fichier server.conf, en spécifiant les chemins appropriés pour le certificat de l'autorité, ainsi que pour la clé et le certificat du serveur Efrei. Il est également nécessaire d'indiquer le chemin et le nom du fichier de la clé Diffie-Hellman et de la clé d'authentification ta.key.

```
┌──(root㉿kali)-[/etc/openvpn/server]
└─# cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf .

┌──(root㉿kali)-[/etc/openvpn/server]
└─# nano server.conf

┌──(root㉿kali)-[/etc/openvpn/server]
└─# grep . server.conf | grep -v '#'
;local a.b.c.d
port 1194
;proto tcp
proto udp
;dev tap
dev tun
;dev-node MyTap
ca pki/ca.crt
cert pki/issued/Efrei.crt
dh pki/dh.pem
```

Ensuite, nous activons le transfert de paquets IP avec net.ipv4.ip_forward pour permettre au serveur VPN de router les paquets entre les clients VPN et Internet.

```
┌──(root㉿kali)-[/etc/openvpn/server]
└─# sudo nano /etc/sysctl.conf

┌──(root㉿kali)-[/etc/openvpn/server]
└─# sudo cat  /etc/sysctl.conf | grep ip_forward
net.ipv4.ip_forward=1
```

Pour simplifier le processus sur la machine cliente, nous copions dans le répertoire /etc/openvpn : le client.conf, le certificat du CA, la clé et le certificat du client et le ta.key.

```
ubuntu@Efrei:~$ sudo cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf /etc/openvpn/
[sudo] password for ubuntu:
ubuntu@Efrei:~$ sudo cp TP2/* /etc/openvpn/
ubuntu@Efrei:~$ ls /etc/openvpn/
ca.crt  client  client.conf  Client_VPN.crt  Client_VPN.key  server  ta.key  update-resolv-conf
```

Nous procédons aux ajustements requis dans le fichier client.conf, en spécifiant les chemins appropriés pour le certificat de l'autorité, ainsi que pour la clé et le certificat du client. Il est également nécessaire d'indiquer le chemin de la clé d'authentification ta.key.

```
ubuntu@Efrei:~$ sudo nano /etc/openvpn/client.conf
ubuntu@Efrei:~$ grep . /etc/openvpn/client.conf | grep -v '#'
client
;dev tap
dev tun
;dev-node MyTap
;proto tcp
proto udp
remote 172.16.29.131 1194
;remote my-server-2 1194
;remote-random
resolv-retry infinite
nobind
;user nobody
;group nobody
persist-key
persist-tun
;mute-replay-warnings
ca ca.crt
cert Client_VPN.crt
key Client_VPN.key
remote-cert-tls server
tls-auth ta.key 1
cipher AES-256-CBC
```

Nous parvenons au terme de notre projet en testant le tunnel entre le serveur et le client, en affichant les nouvelles adresses attribuées à chacun, conformément à la topologie définie dans les fichiers de configuration.

```
└─$ sudo openvpn server.conf
2024-03-21 08:43:05 WARNING: --topology net30 support for server configs with IPv4 pools will be removed in a future release. Please migrate to --topology subnet as soon as possible.
2024-03-21 08:43:05 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM:AES-128-GCM:CHACHA20-POLY1305). OpenVPN ignores --cipher for cipher negotiations.
2024-03-21 08:43:05 Note: NOT using '--topology subnet' disables data channel offload.
2024-03-21 08:43:05 OpenVPN 2.6.7 aarch64-unknown-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] [DCO]
```

```
┌──(kali㉿kali)-[/etc/openvpn/server]
└─$ hostname -I
172.16.29.131 10.8.0.1

┌──(kali㉿kali)-[/etc/openvpn/server]
└─$
```

```
ubuntu@Efrei:/etc/openvpn$ sudo openvpn client.conf
2024-03-21 07:48:20 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM:AES-128-GCM). Future OpenVPN version will ignore --cipher for cipher negotiations. Add 'AES-256-CBC' to --data-ciphers or change --cipher 'AES-256-CBC' to --data-ciphers-fallback 'AES-256-CBC' to silence this warning.
2024-03-21 07:48:20 OpenVPN 2.5.9 aarch64-unknown-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built on Sep 29 2023
2024-03-21 07:48:20 library versions: OpenSSL 3.0.2 15 Mar 2022, LZO 2.10
2024-03-21 07:48:20 Outgoing Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
2024-03-21 07:48:20 Incoming Control Channel Authentication: Using 160 bit message hash 'SHA1' for HMAC authentication
2024-03-21 07:48:20 TCP/UDP: Preserving recently used remote address: [AF_INET]172.16.29.131:1194
2024-03-21 07:48:20 Socket Buffers: R=[212992→212992] S=[212992→212992]
2024-03-21 07:48:20 UDP link local: (not bound)
2024-03-21 07:48:20 UDP link remote: [AF_INET]172.16.29.131:1194
2024-03-21 07:48:20 TLS: Initial packet from [AF_INET]172.16.29.131:1194, sid=7264a75e 1fdd124f
2024-03-21 07:48:20 VERIFY OK: depth=1, CN=autorite
2024-03-21 07:48:20 VERIFY KU OK
2024-03-21 07:48:20 Validating certificate extended key usage
2024-03-21 07:48:20 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2024-03-21 07:48:20 VERIFY EKU OK
```

```
ubuntu@Efrei:~$ hostname -I
172.16.29.137 10.8.0.10
ubuntu@Efrei:~$
```

Dans votre rapport, analysez les échanges de données entre le serveur et le client, soulignant l'importance de la clé Diffie-Hellman et des certificats pour sécuriser la communication. Affichez la table de routage : route -n et discutez de son impact sur le routage des paquets dans le réseau.

♣ S.Y. ♣

Bon travail