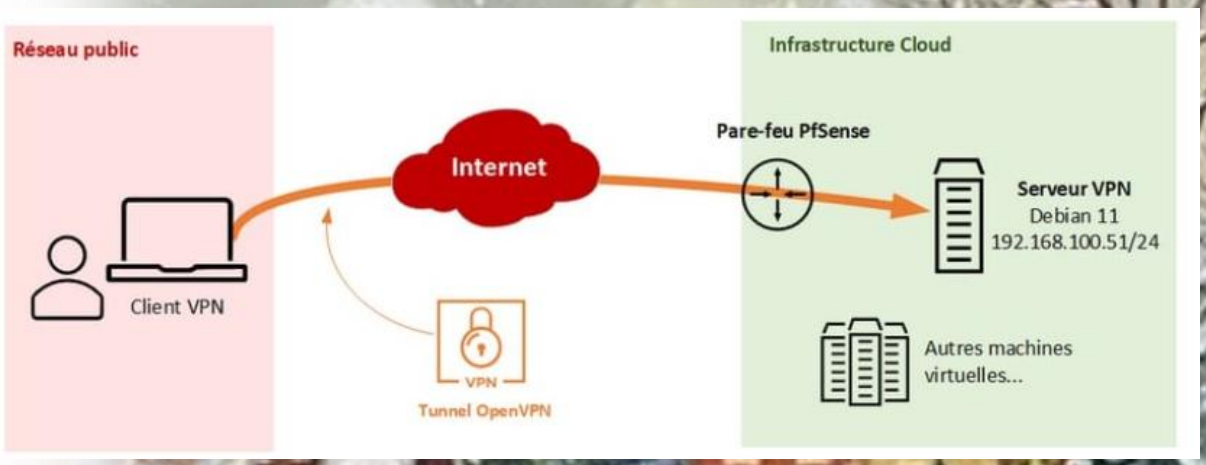
VPN – 1ere méthode qui génère un seul fichier avec des scripts facilitants :

VM *Admin Gr* server anne – anne

Client VPN sciandra mdp : sciandra



Le script GITHUB suivant GitHub - Script d'installation OpenVPN Server va permettre de :

- Installer les paquets suivants : openvpn, iptables, openssl, wget, ca-certificates, curl, unbound
- Configurer OpenVPN via le fichier de configuration : /etc/openvpn/server.conf
- Configurer IPtables sur le serveur pour autoriser les flux
- Activer le routage sur le serveur VPN ("*sysctl net.ipv4.ip_forward = 1" dans /etc/sysctl.d/99-openvpn.conf*)

```
apt-get update

apt-get install curl
```

```
curl -O https://raw.githubusercontent.com/Angristan/openvpn-install/master/openvpn-install.sh
```

```
chmod +x openvpn-install.sh
```

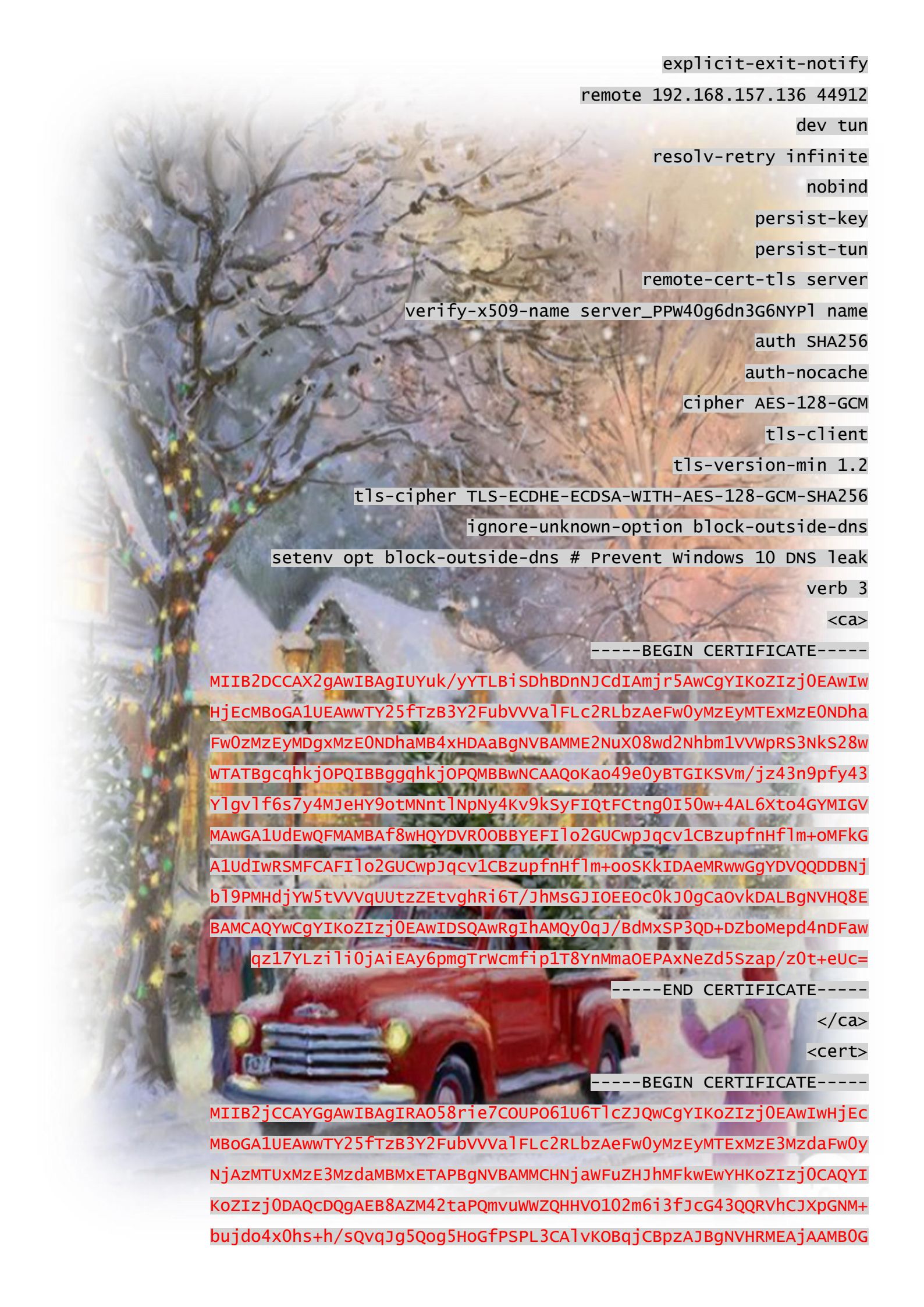
```
./openvpn-install.sh
```

Il est très facile, ensuite, de le paramétrer en fonction des critères propres aux machines utilisées

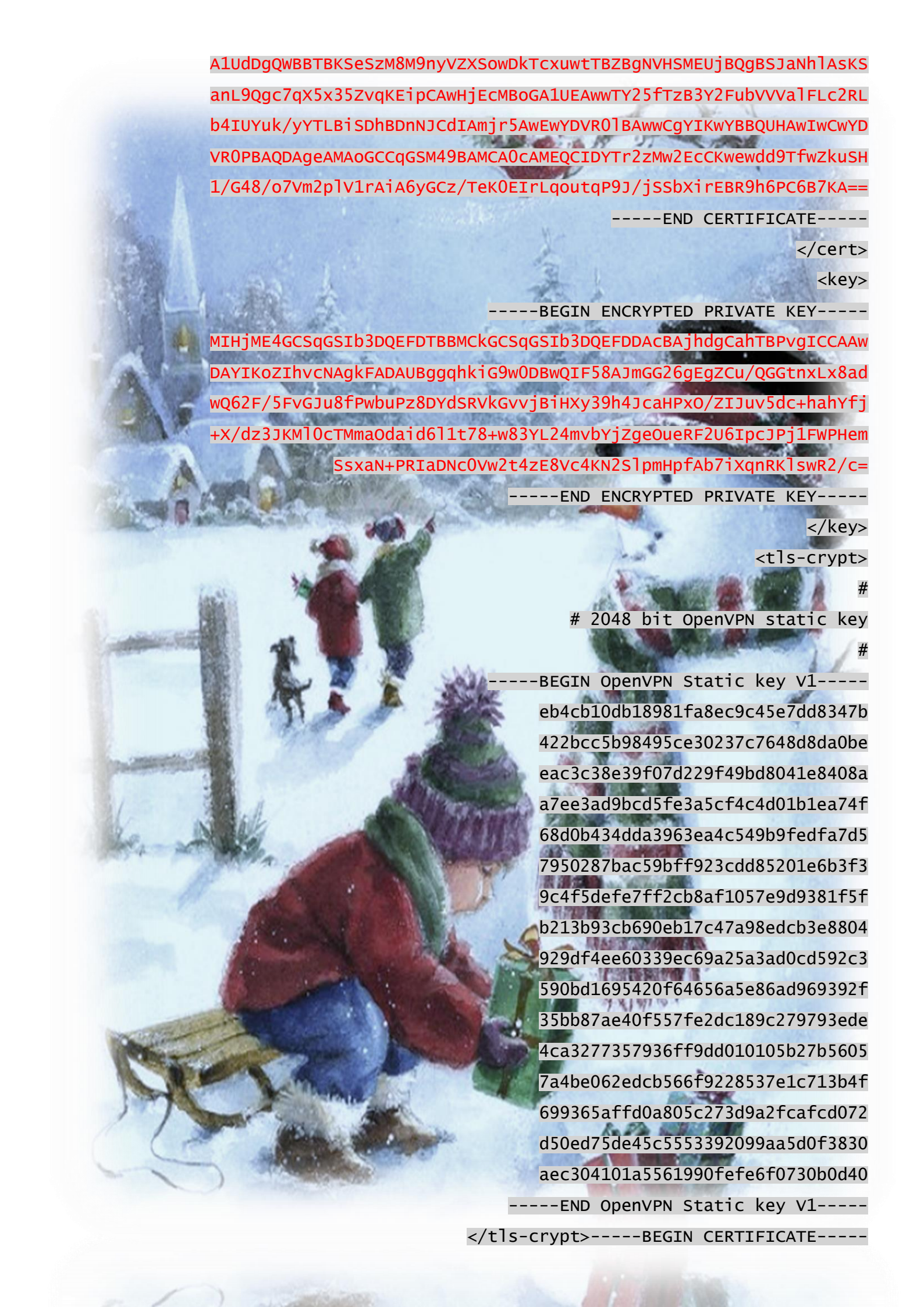
Il suffit de lui indiquer les éléments tels que le type de connexion (Ipv4 ou ipv6), le protocole désiré, le type de port, les options de chiffrement, etc… ce script facilite grandement l'installation, car il va générer le fichier que OpenVPN va utiliser lors de l'installation sur le PC qui se connectera à la VM

Il s'agit de générer ce fichier de préconfig, après avoir téléchargé et installé OpenVPM :, qui porte le **nomduclient.ovpn :**

```
client
proto udp
```



```
explicit-exit-notify
remote 192.168.157.136 44912
dev tun
resolv-retry infinite
nobind
persist-key
persist-tun
remote-cert-tls server
verify-x509-name server_PPW4Og6dn3G6NYPl name
auth SHA256
auth-nocache
cipher AES-128-GCM
tls-client
tls-version-min 1.2
tls-cipher TLS-ECDHE-ECDSA-WITH-AES-128-GCM-SHA256
ignore-unknown-option block-outside-dns
setenv opt block-outside-dns # Prevent Windows 10 DNS leak
verb 3
<ca>
-----BEGIN CERTIFICATE-----
MIIB2DCCAX2gAwIBAgIUYuk/yYTLBiSDhBDnNJCdIAmjr5AwCgYIKoZIzj0EAwIw
HjEcMBoGA1UEAwwTY25fTzB3Y2FubVVValFLc2RLbzAeFw0yMzEyMTExMzE0NDha
Fw0zMzEyMDgxMzE0NDhaMB4xHDAaBgNVBAMME2NuX08wd2Nhbm1VVWpRS3NkS28w
WTATBgcqhkjOPQIBBggqhkjOPQMBBwNCAAQoKao49e0yBTGIKSVm/jz43n9pfy43
Ylgvlf6s7y4MJeHY9otMNntlNpNy4Kv9kSyFIQtFCtng0I50w+4AL6Xto4GYMIGV
MAwGA1UdEwQFMAMBAf8wHQYDVR0OBBYEFIlo2GUCwpJqcv1CBzupfnHflm+oMFkG
A1UdIwRSMFCAFIlo2GUCwpJqcv1CBzupfnHflm+ooSKkIDAeMRwwGgYDVQQDDBNj
bl9PMHdjYw5tVVVqUUtzZEtvghRi6T/JhMsGJIOEEOc0kJ0gCaOvkDALBgNVHQ8E
BAMCAQYwCgYIKoZIzj0EAwIDSQAwRgIhAMQy0qJ/BdMxSP3QD+DZboMepd4nDFaw
qz17YLziliOjAiEAy6pmgTrWcmfip1T8YnMmaOEPAxNeZd5Szap/z0t+eUc=
-----END CERTIFICATE-----
</ca>
<cert>
-----BEGIN CERTIFICATE-----
MIIB2jCCAYGgAwIBAgIRAO58rie7COUPO61U6TlcZJQwCgYIKoZIzj0EAwIwHjEc
MBoGA1UEAwwTY25fTzB3Y2FubVVValFLc2RLbzAeFw0yMzEyMTExMzE3MzdaFw0y
NjAzMTUxMzE3MzdaMBMxETAPBgNVBAMMCHNjaWFuZHJhMFkwEwYHKoZIzj0CAQYI
KoZIzj0DAQcDQgAEB8AZM42taPQmvuWWZQHHVO1O2m6i3fJcG43QQRVhCJXpGNM+
bujdo4x0hs+h/sQvqJg5Qog5HoGfPSPL3CAlvKOBqjCBpzAJBgNVHRMEAjAAMB0G
```



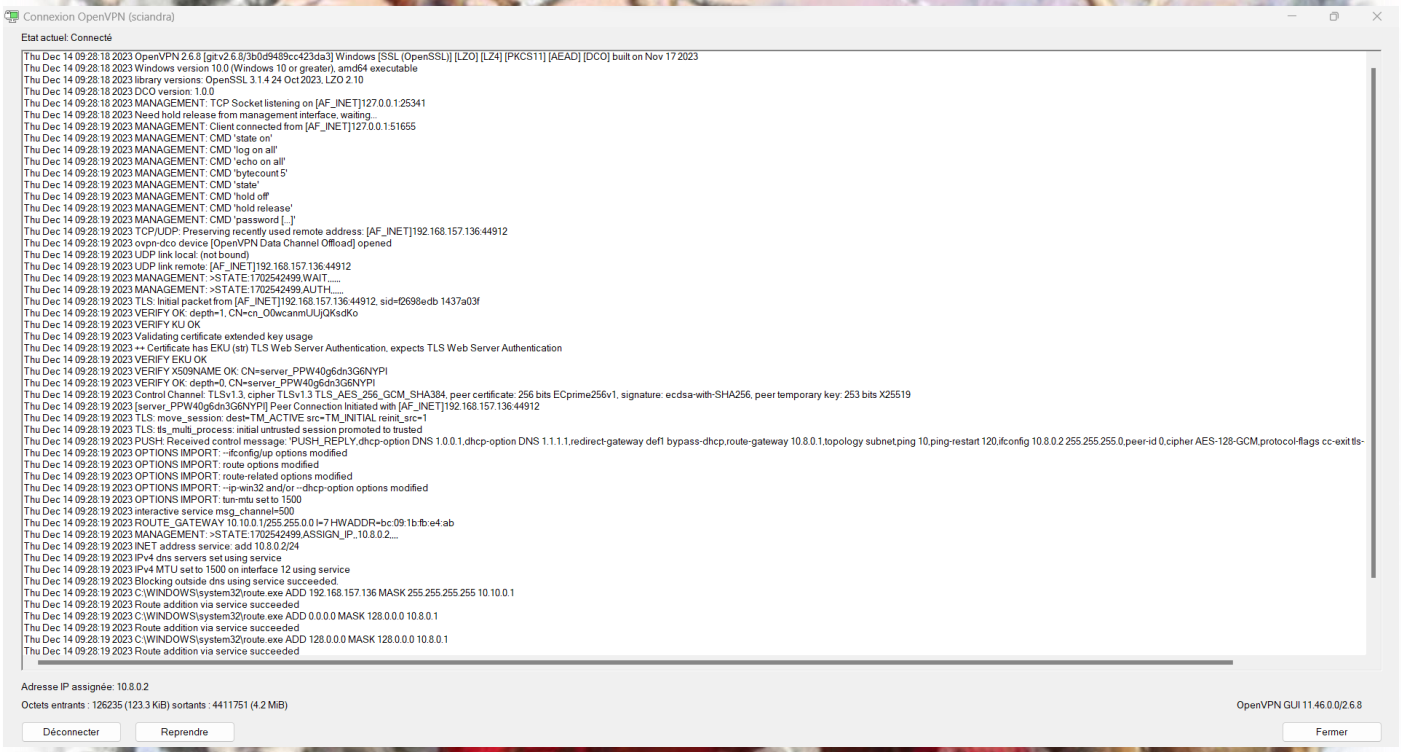
```
A1UdDgQWBBTBKSeSzM8M9nyVZXSowDkTcxuwtTBZBgNVHSMEUjBQgBSJaNhlAsKS
anL9Qgc7qX5x35ZvqKEipCAwHjEcMBoGA1UEAwwTY25fTzB3Y2FubVVValFLc2RL
b4IUYuk/yYTLBiSDhBDnNJCdIAmjr5AwEwYDVR0lBAwwCgYIKwYBBQUHAwIwCwYD
VR0PBAQDAgeAMAoGCCqGSM49BAMCA0cAMEQCIDYTr2zMw2EcCKwewdd9TfwZkuSH
1/G48/o7Vm2plV1rAiA6yGCz/TeK0EIrLqoutqP9J/jSSbXirEBR9h6PC6B7KA==
-----END CERTIFICATE-----
</cert>
<key>
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIHjME4GCSqGSIb3DQEFDTBBMCkGCSqGSIb3DQEFDDAcBAjhdgCahTBPvgICCAAw
DAYIKoZIhvcNAgkFADAUBggqhkiG9w0DBwQIF58AJmGG26gEgZCu/QGGtnxLx8ad
wQ62F/5FvGJu8fPwbuPz8DYdSRVkGvvjBiHXy39h4JcaHPxO/ZIJuv5dc+hahYfj
+X/dz3JKMlOcTMmaOdaid6l1t78+w83YL24mvbYjZgeOueRF2U6IpcJPj1FWPHem
SsxaN+PRIaDNcOVw2t4zE8Vc4KN2SlpmHpfAb7iXqnRKlswR2/c=
-----END ENCRYPTED PRIVATE KEY-----
</key>
<tls-crypt>
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
eb4cb10db18981fa8ec9c45e7dd8347b
422bcc5b98495ce30237c7648d8da0be
eac3c38e39f07d229f49bd8041e8408a
a7ee3ad9bcd5fe3a5cf4c4d01b1ea74f
68d0b434dda3963ea4c549b9fedfa7d5
7950287bac59bff923cdd85201e6b3f3
9c4f5defe7ff2cb8af1057e9d9381f5f
b213b93cb690eb17c47a98edcb3e8804
929df4ee60339ec69a25a3ad0cd592c3
590bd1695420f64656a5e86ad969392f
35bb87ae40f557fe2dc189c279793ede
4ca3277357936ff9dd010105b27b5605
7a4be062edcb566f9228537e1c713b4f
699365affd0a805c273d9a2fcafcd072
d50ed75de45c5553392099aa5d0f3830
aec304101a5561990fefe6f0730b0d40
-----END OpenVPN Static key V1-----
</tls-crypt>-----BEGIN CERTIFICATE-----
```

Sur le PC qui va se connecter : télécharger OPENVPN,

Ouvrir WinSCP pour aller récupérer le fichier,

Importer le fichier dans OpenVPN va leur interface « importer »

Ouvrir OpenVPN :



Sur la VM s'assurer de la présence de TUN0 :

```
root@ldap:/etc/openvpn# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:57:3d:1e brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.157.136/24 brd 192.168.157.255 scope global dynamic noprefixroute ens33
       valid_lft 1731sec preferred_lft 1731sec
    inet6 fe80::20c:29ff:fe57:3d1e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 500
    link/none
    inet 10.8.0.1/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::ae57:e8d3:b32b:28e/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
root@ldap:/etc/openvpn#
```

VPN – 2eme méthode qui génère quatre fichiers avec paramètres à faire manuellement :

VM *Zoneminder*

anne – anne

en root : installer ces paquets :

```
apt -y install openvpn easy-rsa iptables
```

il s'agit de paramétrer les fichiers de config :

lancer :

```
./easyrsa init-pki
```

Qui est dans `/usr/share/easy-rsa`

Puis :

```
./easyrsa build-ca
```

Ceci va demander les paramètres : password, DN, nom du serveur, le fichier créé est :

```
/usr/share/easy-rsa/pki/ca.crt
```

Il faut ensuite créer les certificats des serveurs pour qu'ils puissent se connecter. Lancer :

```
./easyrsa build-server-full server1 nopass
```

Pour générer ces fichiers :

```
req: /usr/share/easy-rsa/pki/reqs/server1.req
key: /usr/share/easy-rsa/pki/private/server1.key
```

pour signer le certificat faire yes

```
Using configuration from /usr/share/easy-rsa/pki/496e5420/temp.00b726f7
Enter pass phrase for /usr/share/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 Mar 17 13:55:17 2026 GMT (825 days)

Write out database with 1 new entries
Database updated

* Notice:
```

Le fichier :

```
/usr/share/easy-rsa/pki/issued/server1.crt
```

est généré

```
# 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 1194

# TCP or UDP server?
#proto tcp
proto udp

# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel.
# Use "dev tap0" if you are ethernet bridging
# and have precreated a tap0 virtual interface
# and bridged it with your ethernet interface.
# If you want to control access policies
# over the VPN, you must create firewall
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
#dev tap
                                              [ ligne  52/316 (16%), col.  1/ 9 ( 11%), car.  1998/10856 (18%) ]
cp /usr/share/doc/openvpn/examples/sample-config-files/server.conf /etc/openvpn/server/
```

il faut ensuite modifier le fichier de conf de la manière suivante :

nano – **/etc/openvpn/server/server.conf**

```
                                                    anne@Zoneminder: ~
Fichier  Édition  Affichage  Recherche  Terminal  Aide
 GNU nano 7.2                                        /etc/openvpn/server/server.conf
#############################################
# Sample OpenVPN 2.0 config file for        #
# multi-client server.                      #
#                                           #
# This file is for the server side          #
# of a many-clients <-> one-server          #
# OpenVPN configuration.                    #
#                                           #
# OpenVPN also supports                     #
# single-machine <-> single-machine         #
# configurations (See the Examples page     #
# on the web site for more info).           #
#                                           #
# This config should work on Windows        #
# or Linux/BSD systems.  Remember on        #
# Windows to quote pathnames and use        #
# double backslashes, e.g.:                 #
# "C:\\Program Files\\OpenVPN\\config\\foo.key" #
#                                           #
# Comments are preceded with '#' or ';'     #
#############################################

# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

                                              [ ligne   1/316 (  0%), col.  1/50 (  2%), car.     0/10856 (  0%) ]
```

```
dev tun

# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one.  On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap

# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key).  Each client
# and the server must have their own cert and
# key file.  The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys.  Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca ca.crt
                                              [ ligne  78/316 (24%), col.  1/10 ( 10%), car.  2845/10856
```

```
cert issued/server1.crt
key private/server1.key  # This file should be kept secret

# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048
dh dh.pem

# Network topology
# Should be subnet (addressing via IP)
# unless Windows clients v2.0.9 and lower have to
# be supported (then net30, i.e. a /30 per client)
# Defaults to net30 (not recommended)
;topology subnet

# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 192.168.100.0 255.255.255.0

# Maintain a record of client <-> virtual IP address
# associations in this file.  If OpenVPN goes down or
                                              [ ligne 104/316 (32%), col.  1/54 (  1%), car.  3681/10856 (33%) ]
```

```
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist /var/log/openvpn/ipp.txt

# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface.  Then you must manually set the
# IP/netmask on the bridge interface, here we
# assume 10.8.0.4/255.255.255.0.  Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients.  Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100

# Configure server mode for ethernet bridging
# using a DHCP-proxy, where clients talk
# to the OpenVPN server-side DHCP server
# to receive their IP address allocation
# and DNS server addresses.  You must first use
# your OS's bridging capability to bridge the TAP
# interface with the ethernet NIC interface.
# Note: this mode only works on clients (such as
# Windows), where the client-side TAP adapter is
                                              [ ligne 130/316 (41%), col.  1/49 (  2%), car.  4802/10856 (44%) ]
```

```
# bound to a DHCP client.
;server-bridge

# Push routes to the client to allow it
# to reach other private subnets behind
# the server.  Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.

push "route 10.0.0.0 255.255.255.0"

# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).

# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
                                              [ ligne 156/316 (49%), col.  1/38 (  2%), car.  5676/10856 (52%) ]
```

```
# Then create a file ccd/Thelonious with this line:
#   iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN.  This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.

# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
#   ifconfig-push 10.9.0.1 10.9.0.2

# Suppose that you want to enable different
# firewall access policies for different groups
# of clients.  There are two methods:
# (1) Run multiple OpenVPN daemons, one for each
#     group, and firewall the TUN/TAP interface
#     for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
#     modify the firewall in response to access
#     from different clients.  See man
#     page for more info on learn-address script.
;learn-address ./script
                                                [ ligne 182/316 (57%), col.  1/24 (
```

```
# 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 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"

# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
;client-to-client
                                                [ ligne 209/316 (66%), col.  1/18 (
```

```
# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names.  This is recommended
# only for testing purposes.  For production use,
# each client should have its own certificate/key
# pair.
#
# IF YOU HAVE NOT GENERATED INDIVIDUAL
# CERTIFICATE/KEY PAIRS FOR EACH CLIENT,
# EACH HAVING ITS OWN UNIQUE "COMMON NAME",
# UNCOMMENT THIS LINE OUT.
;duplicate-cn

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120

# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
                                              [ ligne 236/316 (74%), col.  1/ 2 ( 50%),
```

```
# 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 # This file is secret

# 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

# Enable compression on the VPN link and push the
# option to the client (v2.4+ only, for earlier
# versions see below)
;compress lz4-v2
;push "compress lz4-v2"

# For compression compatible with older clients use comp-lzo
# If you enable it here, you must also
# enable it in the client config file.
                                              [ ligne 262/316 (82%), col.  1/39 (  2%
```

```
;comp-lzo

# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100

# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
# You can uncomment this on non-Windows
# systems after creating a dedicated user.
;user openvpn
;group openvpn

# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun

# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status /var/log/openvpn/openvpn-status.log

                                                [ ligne 263/316 (83%), col
```

```
  GNU nano 7.2                                                  /etc/o
# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it.  Use one
# or the other (but not both).
;log         /var/log/openvpn/openvpn.log
;log-append  /var/log/openvpn/openvpn.log

# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 3

# Silence repeating messages.  At most 20
# sequential messages of the same message
# category will be output to the log.
;mute 20

# Notify the client that when the server restarts so it
# can automatically reconnect.
explicit-exit-notify 1

                                                [ ligne 316/316 (100%), col.  1/ 1 (100%
```

Paramétrage des lignes suivantes :

32, 35, 53, 78, 85, 101, 142, 231, 244, 281, 306 Avec les critères de notre installation

Créer ce fichier : nano **/etc/openvpn/server/add-bridge.sh**

```
Fichier  Édition  Affichage  Recherche  Terminal  Aide
  GNU nano 7.2                                        /etc/openvpn/server/add-bridge.sh *
#!/bin/bash

# network interface which can connect to local network
IF=ens33
# interface VPN tunnel uses
# for the case of this example like specifying [tun] on the config, generally this param is [tun0]
VPNIF=tun0

echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -A FORWARD -i tun0 -j ACCEPT
iptables -t nat -A POSTROUTING -o ens33 -j MASQUERADE
```

Et

Nano **/etc/openvpn/server/remove-bridge.sh**

```
Fichier  Édition  Affichage  Recherche  Terminal  Aide
  GNU nano 7.2                                        /etc/openvpn/server/remove-bridge.sh
#!/bin/bash

# network interface which can connect to local network
IF=ens33
# interface VPN tunnel uses
# for the case of this example like specifying [tun] on the config, generally this param is [tun0]
VPNIF=tun0

echo 0 > /proc/sys/net/ipv4/ip_forward
iptables -D FORWARD -i  ${VPNIF} -j ACCEPT
iptables -t nat -D POSTROUTING -o ${IF} -j MASQUERADE
```

On change les droits pour pouvoir les utiliser :

chmod 700 /etc/openvpn/server/{add-bridge.sh,remove-bridge.sh}

dans la ligne [service] : ajouter les add et remove bridge

```
[Service]
Type=notify
PrivateTmp=true
WorkingDirectory=/etc/openvpn/server
ExecStart=/usr/sbin/openvpn --status %t/openvpn-server/s
CapabilityBoundingSet=CAP_IPC_LOCK CAP_NET_ADMIN CAP_NET
LimitNPROC=10
DeviceAllow=/dev/null rw
DeviceAllow=/dev/net/tun rw
ProtectSystem=true
ProtectHome=true
KillMode=process
RestartSec=5s
Restart=on-failure

ExecStartPost=/etc/openvpn/server/add-bridge.sh
ExecStopPost=/etc/openvpn/server/remove-bridge.sh
```

systemctl daemon-reload

systemctl enable --now openvpn-server@server

le serveur VPN est prêt

il faut ensuite aller configurer le VPN dans le PC qui va se connecter à la VM (ici : mon PC Hôte°)

avec filezilla, les mettre dans le PC et les lier au VPN du PC

- /etc/openvpn/server/ca.crt
- /etc/openvpn/server/ta.key
- /etc/openvpn/server/issued/client1.crt
- /etc/openvpn/server/private/client1.key

Il faut modifier le fichier .ovpn pour lui dire d'utiliser ces fichiers.

Après, on connecte la VM avec le PC et c'est magnifique . En plus là, ca sert à rien 😊

J'espère qu'avec ça je vais avoir plein d'étoiles car en plus c'est Noël