

Atelier sur OpenVPN

Auteur : Alice de Bignicourt (UREC/CNRS)

Relecteurs : Françoise Berthoud – Dominique Fournier

8 juin 2006

Nous allons voir comment installer OpenVPN, puis renforcer le niveau de sécurité de cette installation.

Table des matières

1 Généralités.....	3
2 Certificats.....	4
2.1 Certificats côté serveur.....	4
Certificat de la machine.....	4
Autorité(s) de certification.....	4
2.2 Certificat côté client.....	5
2.3 Comment faire pour obtenir un certificat CNRS-Standard ?.....	5
3 Télécharger et Installer (TP).....	5
3.1 Télécharger.....	5
Côté serveur.....	5
Côté client.....	5
3.2 Installer.....	6
4 Configuration simple (TP).....	6
4.1 Choix du protocole (serveur & client).....	6
4.2 Méthode d'authentification (serveur & client).....	7
4.3 Définir le mode routeur plutôt que le mode pont (serveur & client).....	7
4.4 Générer le paramètre Diffie-Hellman (serveur).....	7
4.5 Définir le serveur OpenVPN comme gateway (serveur).....	7
4.6 Fichier server.conf.....	7
4.7 Fichier client.conf.....	9
4.8 Vérifier que tout se passe bien.....	10
Lors du lancement du serveur.....	10
Lors du lancement du client.....	10
5 Sécuriser l'installation.....	11
5.1 Aux niveaux serveur et client.....	11
5.2 Au niveau du serveur.....	11
Abaisser les privilèges :.....	11
Emprisonner le serveur, redéfinir la racine :.....	12
Utilisation d'un script.....	12
6 Annexes.....	16
6.1 Bibliographie.....	16
6.2 A Guide to basic RSA Key Management.....	16
Build your own root certificate authority (ca) certificate/key.....	16

Build an intermediate certificate authority certificate/key (optional).....	16
Build diffie-hellman parameters (necessary for the server end of a ssl/tls connection).....	17
Build a certificate signing request	17
Sign a certificate signing request.....	17
Build and sign a certificate signing request using a locally installed root certificate/key...	17
6.3 Convertir un certificat PEM ↔ p12.....	18
Convertir PEM → p12.....	18
Convertir p12 → PEM.....	19

1 Généralités

OpenVPN est disponible sur plusieurs plateformes : Linux, Windows 2000/XP et plus récent, OpenBSD, FreeBSD, NetBSD, Mac OS X, et Solaris.

Un utilisateur sur un réseau distant veut accéder à un serveur dans la zone interne du réseau de son laboratoire. Dans cet exemple, l'architecture réseau du laboratoire est composée de deux zones :

- la « zone semi-ouverte », accessible de l'Internet
- la « zone interne » hébergeant les serveurs internes du laboratoire

Ces deux premières zones sont séparées par un routeur assurant la sécurité entre elles. Une troisième zone nommée « zone VPN » est créée pour les machines distantes. Elle regroupe une plage d'adresses qui sera utilisée par le serveur OpenVPN pour celles-ci.

Les zones se partagent les adresses IP de la manière suivante :

- Zone semi-ouverte : 195.195.195.32/27
- Zone interne : 195.195.195.64/27
- Zone VPN : 195.195.195.224/27

Voici un schéma représentant ce que nous cherchons à mettre en place :

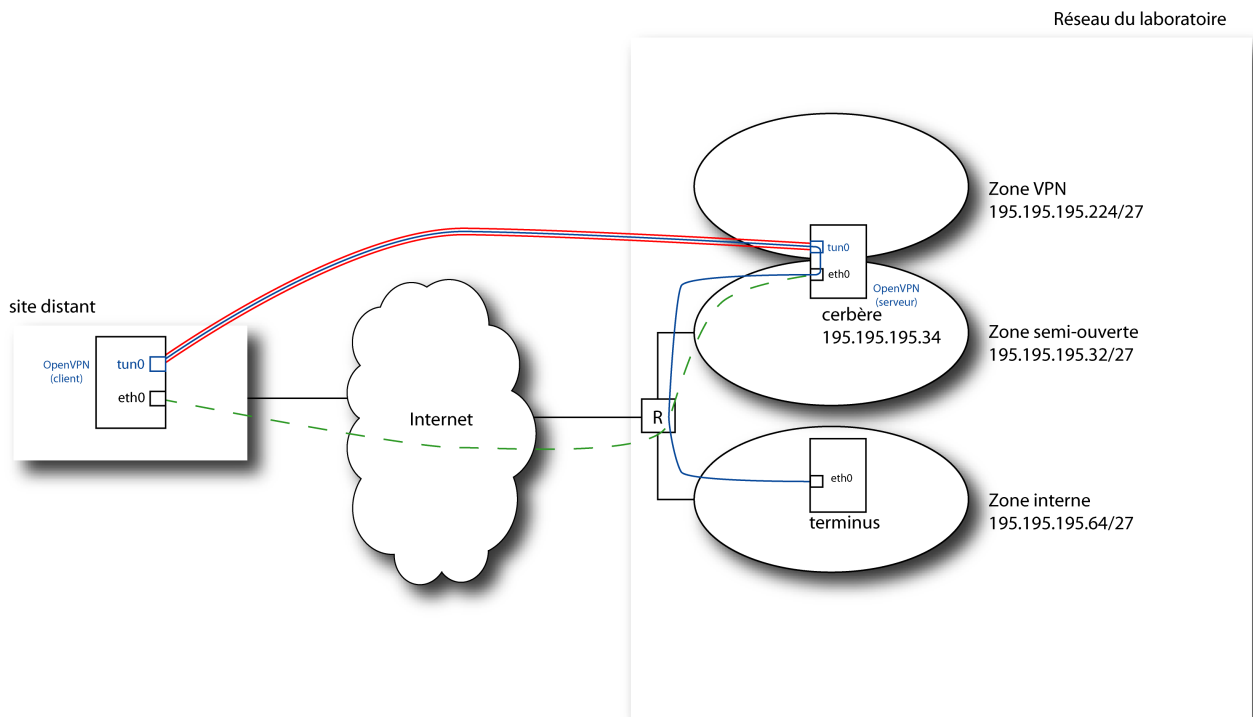


Figure 1 : Vue d'ensemble

Le processus est le suivant :

1. L'utilisateur démarre le client OpenVPN et se connecte au serveur OpenVPN sur la machine *Cerbère* (se trouvant dans la zone semi-ouverte) via Internet (traits pointillés vert).
2. A l'ouverture du tunnel, le serveur OpenVPN :
 1. affecte des adresses IP aux extrémités du tunnel entre le serveur et la machine cliente dans la zone VPN du réseau du laboratoire.
 2. envoie des commandes qui seront exécutées sur la machine cliente en particulier pour créer une interface réseau et modifier sa table de routage.
3. Une fois le tunnel établi, toutes les connexions entre le poste client et la zone interne du réseau du laboratoire passeront par le tunnel. Les datagrammes provenant de la machine cliente à destination du réseau du laboratoire auront une adresse source contenue dans la plage d'adresses de la « zone VPN ». La mise en place de la politique de filtrage en sera facilitée.

2 Certificats

Dans le cadre de cet atelier, les fichiers contenant certificats et la clé privée vous seront fournis.

2.1 Certificats côté serveur

Certificat de la machine

Le serveur doit avoir son certificat qui doit être de type « serveur » défini dans les extensions. Le serveur doit aussi avoir sa clé privée (fichier critique dans le sens où il doit être gardé secret).

Autorité(s) de certification

Le fichier CAs.crt doit contenir le(s) certificat(s) de la chaîne de certification de l'Autorité de Certification (AC) du certificat du serveur et des Autorités de Certification autorisées à se connecter au serveur OpenVPN pour établir un tunnel.

Ces certificats sont au format PEM. Par exemple :

Un certificat au format PEM a cette

```
-----BEGIN CERTIFICATE-----
MIIDZDCCakygAwIBAgIBADANBgkqhkiG9w0BAQQFADArMQswCQYDVQQGEwJGUjEN
MAsGA1UEChMEQ05SUzENMAsGA1UEAxMEQ05SUzAeFw0wMjA0MjcwNTQ0MzZaFw0y
MTA0MjIwNTQ0MzZaMAsCZAJBgNVBAYTAkZSMQ0wCwYDVQQKEwRDTlJTMQ0wCwYD
VQQDEwRDTlJTMiIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAA3Xer8er8
eLUUodx3YlaXjC+3VMJMKbUfSjHe3Sp9+OtfFWyFPBIXziPAR2SEbaIT8QV9W9Y
W/eJtSfqr6D8oI6Iho+fJLaQtiTcZ9BPj35WLRsoB3KxF2egDttCTsN8tCWi+IwE
samCXYyP1IN77qf19Lb9ltuwoEQ1pqrXYgcNsoFZGhLi57AUJQj/bYota9dpN2m
xdPYVys++LW6xNj/EiJfJGkHYuQ0Sgh3yjC77NptdZBooxxciFeFSYsJSGEKp5X
GIFyI71mHw/jt70X2hK6GVT0HC2PcVIszbYotnpoy5pNUjj6SIzBS4mWj8YXW8u5
Dg6BxBRHNd1XvQIDAQABo4GSMIGPMawGA1UdEwQFMAMBAf8wHQYDVR0OBBYEFFbr
aLnSXH6YtaVTw5FvY1jE+Wu3MFMGA1UdIwRMMEqAFFbraLnSXH6YtaVTw5FvY1jE
+Wu3oS+kLTAzMQswCQYDVQQGEwJGUjENMAsGA1UEChMEQ05SUzENMAsGA1UEAxME
```

```
Q05SU4IBADALBgNVHQ8EBAMCAQYwDQYJKoZIhvcNAQEEBQADggEBADjXwym8enei
XhZHSYZe0Bk4at6BCAK5p6ACpoi4DeJJNc7mqgI00vmjhDeaFelZK7e9zBGuKSGP
j5E5+p134ug56uwulspIRyJMzBHTtm9YGzQuihCdEoR0pH1SVwBTfMiY3oFuxQt1
p1rn0zUIRYj1hFCY8Ac+xYY+LgLaoty2oGt7N6ufA3Bu3fWcWA4F7LWEWY00iSEQ
iK37LQjkABMcVbOPd74grNwBHH18ZwxaX0+51Im+q5osErGoY8ZigAP9THCVvcho
BdrLvgmmH+1t0oUuQ9MfG1x2/hN2YWD5ZNRyeL9/7+Vzo0PaKn932zR5ctm05aXt
UtDERkxfG6o=
-----END CERTIFICATE-----
```

Pour vérifier le certificat (un seul à la fois) on peut utiliser la commande suivante :

```
openssl x509 -in CAs.crt -noout -text
```

Cette commande affiche le certificat dans un format que l'on peut lire.

2.2 Certificat côté client

Un seul fichier permet d'identifier la machine cliente. Il contient le certificat de l'AC émettrice, le certificat du client et sa clé privée. Il est au format pkcs12.

2.3 Comment faire pour obtenir un certificat CNRS-Standard ?

En allant sur l'interface web de l'IGC du CNRS :

<http://igc.services.cnrs.fr/CNRS-Standard/>

cliquer sur l'onglet 'Certificat' et choisir :

- certificat personnel pour identifier une personne
- certificat serveur pour identifier une machine ou un service

3 Télécharger et Installer (TP)

3.1 Télécharger

Côté serveur

La librairie LZO n'est pas nécessaire à l'exécution du serveur OpenVPN, mais elle permet de compresser les échanges entre le client et le serveur. Télécharger_la en allant sur le lien :

<http://www.oberhumer.com/opensource/lzo/download/>

Télécharger la version 1.08-2_1 (par rpmfind) et taper la commande :

```
root yum install lzo-1.08-2_1.rhfc3.at.i386.rpm
```

Les sources OpenVPN sur le serveur,

<http://openvpn.net/download.html>

```
root wget http://openvpn.net/release/openvpn-2.0.7.tar.gz
```

Côté client

Client graphique :

<http://openvpn.se/download.html>

Télécharger la dernière version stable et l'installer sur le poste.

3.2 Installer

Pour installer LZO, exécuter la commande :

```
root# yum install lzo
```

Exécuter pour OpenVPN :

```
root# gunzip openvpn-2.0.7.tar.gz
root# tar xvf openvpn-2.0.7.tar
root# cd openvpn-2.0.7
root# ./configure \
--with-ssl-headers=/usr/local/openssl/include \
--with-ssl-lib=/usr/local/openssl/lib
root# make
root# make install
root# mkdir /etc/openvpn
root# cd /etc/openvpn
root# cp /usr/local/src/openvpn-2.0.7/sample-config-files/server.conf .
root# touch log/openvpn.log
root# touch log/openvpn-status.log
```

Ou vous pouvez également exécuter la commande

```
root# yum install openvpn
```

lancer le serveur

```
root# /etc/rc.d/init.d/openvpn start
```

ou encore

```
root# /usr/local/sbin/openvpn --writepid /var/run/openvpn/server.pid --config server.conf
```

en mode « daemon »

```
root# /usr/local/sbin/openvpn --daemon --writepid /var/run/openvpn/server.pid --config server.conf
```

4 Configuration simple (TP)

Préambule sur les fichiers de configuration :

Les fichiers de configuration serveur et client(s) sont très similaires. Certaines directives se retrouvent dans les deux fichiers.

Sur le serveur :

Copier le fichier */sources/sample-config-files/server.conf* dans */etc/openvpn*

Sur le poste client :

Copier le fichier « *C:\Program Files\OpenVPN\sample-config\client.ovpn* » dans « *C:\Program Files\OpenVPN\config* »

Editer ce fichier pour pouvoir effectuer les modifications suivantes :

4.1 Choix du protocole (serveur & client)

UDP ? TCP ?

Le choix du protocole dépend de la politique de sécurité du réseau. Dans le fichier de configuration du serveur et dans celui du/des client(s), spécifier la directive **proto** :

```
proto tcp
```

4.2 Méthode d'authentification (serveur & client)

OpenVPN propose plusieurs méthodes d'authentification comme par exemple, login/mot de passe, certificat. Lors de cet atelier, la méthode d'authentification se fait par certificat, Préparer des certificats pour la configuration de votre réseau sécurisé ou les récupérer.

Sur le poste serveur :

```
ca /etc/openvpn/CAs.crt
cert /etc/openvpn/server.pem
key /etc/openvpn/server.key # This file should be kept secret
```

Sur le poste client :

```
pkcs12 "C:\\Program Files\\OpenVPN\\config\\user.p12"
```

4.3 Définir le mode routeur plutôt que le mode pont (serveur & client)

Dans les fichiers de configuration serveur et client(s), spécifier la directive :

```
dev tun
```

Activer le mode routeur pour Linux sur le serveur

```
root# echo 1 > /proc/sys/net/ipv4/ip_forward
```

A titre d'information qui ne sera pas fait dans le cadre de cet atelier : Si on veut mettre en place le NAT, il faut mettre dans le fichier de configuration d'Iptables :

```
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4
```

où **1.2.3.4** est l'adresse de sortie du routeur et **eth0** l'interface de sortie

4.4 Générer le paramètre Diffie-Hellman (serveur)

Exécuter la commande :

```
root# openssl dhparam -out dh1024.pem 1024
```

4.5 Définir le serveur OpenVPN comme gateway (serveur)

Le principe est de faire passer toutes les routes par le serveur OpenVPN. Spécifier la directive dans le fichier de configuration du serveur comme suit :

```
push "redirect-gateway"
```

4.6 Fichier server.conf

```
#Fichier de configuration de OpenVPN partie SERVEUR
#####

port 1194
```

```

proto tcp
dev tun
#certificates on the server
ca /etc/openvpn/CAs.crt
cert /etc/openvpn/server.pem
key /etc/openvpn/server.key # This file should be kept secret

# Diffie hellman parameters, generate your own with:
# $ openssl dhparam -out dh1024.pem 1024
# Substitute 2048 for 1024 if you are using 2048 bit keys.
dh /etc/openvpn/dh1024.pem

# Configure server mode and supply a VPN subnet
server 10.8.0.0 255.255.255.0
# Cette directive revient a définir
# ifconfig 10.8.0.1 10.8.0.2
# ifconfig-pool 10.8.0.4 10.8.0.251
# route 10.8.0.0 255.255.255.0
# push "route 10.8.0.1"

# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120
# 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 /etc/openvpn/log/openvpn-status.log

# 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-append /etc/openvpn/log/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 5
```

4.7 Fichier *client.conf*

```
#Fichier de configuration de OpenVPN partie CLIENT
#####

client
dev tun
proto tcp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 195.220.197.110 1194

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server. Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific local port number.
nobind

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

# Try to preserve some state across restarts.
persist-key
persist-tun

# 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.
pkcs12 "C:\\Program Files\\OpenVPN\\config\\user.p12"
```



```
with 195.220.197.65:1527
Thu Apr 13 14:47:47 2006 us=7167 Alice_De_Bignicourt/195.220.197.65:1527 MULTI: Learn: 10.8.0.6 ->
Alice_De_Bignicourt/195.220.197.65:1527
Thu Apr 13 14:47:47 2006 us=7201 Alice_De_Bignicourt/195.220.197.65:1527 MULTI: primary virtual IP
for Alice_De_Bignicourt/195.220.197.65:1527: 10.8.0.6
RThu Apr 13 14:47:48 2006 us=126716 Alice_De_Bignicourt/195.220.197.65:1527 PUSH: Received control
message: 'PUSH_REQUEST'
Thu Apr 13 14:47:48 2006 us=126831 Alice_De_Bignicourt/195.220.197.65:1527 SENT CONTROL
[Alice_De_Bignicourt]: 'PUSH_REPLY,route 10.8.0.1,ping 10,ping-restart 120,ifconfig 10.8.0.6
10.8.0.5' (status=1)
WWRRR
```

Côté client :

On doit avoir une bulle de confirmation et voir quelle adresse de la zone VPN a été allouée au client par le serveur. Pour rappel, la zone VPN est définie par la directive **ifconfig-pool** . Par contre, on peut regarder quelles sont les routes définies sur le client après établissement du tunnel :

- ouvrir une fenêtre 'Invite de commandes'
- exécuter la commande :

```
netstat -r
```

5 Sécuriser l'installation

5.1 Aux niveaux serveur et client

Utilisation d'une clé statique partagée. Pour la créer, utiliser la commande **openvpn** suivante :

```
root# cd /etc/openvpn
root# openvpn --genkey --secret ta.key
```

Rajouter dans le fichier de configuration du serveur :

```
tls-auth ta.key 0 # --- 0 pour le serveur
```

Récupérer la clé pour la mettre sur la machine client et rajouter dans le fichier de configuration du client :

```
tls-auth "C:\Program Files\OpenVPN\config\ta.key" 1 # --- 1 pour les clients
```

5.2 Au niveau du serveur

Abaisser les privilèges :

Tout d'abord créer le groupe et le user **openvpn** en exécutant les commandes :

```
root# groupadd openvpn
root# adduser openvpn -g openvpn
```

Changer les droits sur les fichiers pour permettre à l'utilisateur de lire les fichiers de configuration :

```
root# chown -R openvpn:openvpn /etc/openvpn
root# ls -l
total 60
-rw-r--r-- 1 openvpn openvpn 4720 Apr 13 13:52 CAs.crt
-rw-r--r-- 1 openvpn openvpn 245 Apr 12 17:44 dh1024.pem
-rw----- 1 openvpn openvpn 29 Apr 13 15:13 ipp.txt
drwxr-xr-x 2 openvpn openvpn 4096 Apr 13 11:27 log
-rw-r--r-- 1 openvpn openvpn 419 Apr 13 15:02 server.conf
-rwxr--r-- 1 openvpn openvpn 886 Apr 12 17:34 server.key
-rwxr--r-- 1 openvpn openvpn 1329 Apr 12 17:34 server.pem
-rw----- 1 openvpn openvpn 636 Apr 13 15:06 ta.key
```

Modifier le fichier de configuration du serveur :

```
user openvpn
group openvpn
```

Emprisonner le serveur, redéfinir la racine :

Permet de redéfinir la racine du serveur et évite ainsi toute lecture ou écriture dans des fichiers externes au serveur OpenVPN. Dans le fichier de configuration du serveur, rajouter :

```
chroot /etc/openvpn
```

NB : pour l'utiliser, il faut copier le bash static, la commande sh ainsi que tous les exécutables et toutes les bibliothèques nécessaires au serveur OpenVPN et aux scripts externes appelés après son initialisation dans le répertoire spécifié par la directive **chroot**.

```
root# ls /etc/openvpn/bin
bash  sh
```

Pour retrouver les bibliothèques associées aux exécutables, vous pouvez utiliser la commande :

```
root# ldd /bin/bash
        libtermcap.so.2 => /lib/libtermcap.so.2 (0x00936000)
        libdl.so.2 => /lib/libdl.so.2 (0x00fdd000)
        libc.so.6 => /lib/tls/libc.so.6 (0x006e4000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00195000)
```

Puis copier ces bibliothèques dans `/etc/openvpn/lib`

Utilisation d'un script

Lorsque le client se connecte au serveur OpenVPN, une première vérification est faite sur la validité du certificat. Cette vérification permet d'authentifier le client, mais ne permet pas de vérifier les droits d'accès. Nous allons utiliser un script simple dans le cadre de cet atelier. Le script suivant récupère le champ OU du DN du certificat présenté par le client et vérifie s'il fait parti d'une liste d'organismes autorisés par le serveur OpenVPN pour ensuite établir un tunnel.

Voici le script :

```
#!/usr/bin/perl
# Script verify-dn
# -----
# envoie
# 0 si le DN du client correspond a un utilisateur du laboratoire, renvoie
# 1 dans le cas contraire
#
# Basé sur le script /usr/share/openvpn/sample-scripts/verify-cn
#
die "usage: verify-dn certificate_depth X509_NAME_oneline" if (@ARGV != 2);

# Parse out arguments:
# depth -- The current certificate chain depth.  In a typical
#          bi-level chain, the root certificate will be at level
#          1 and the client certificate will be at level 0.
#          This script will be called separately for each level.
# x509   -- the X509 subject string as extracted by OpenVPN from
#          the client's provided certificate.
($depth, $x509) = @ARGV;

if ($depth == 0) {
    # If depth is zero, we know that this is the final
    # certificate in the chain (i.e. the client certificate),
    # and the one we are interested in examining.
    # If so, parse out the common name substring in
    # the X509 subject string.
```

```

#
# Accepte la connection pour :
# MONLABO :
if ($x509 =~ m|^/c=fr/o=cnrs/ou=MONLABO|oi) { exit 0; }
# UPS836 :
if ($x509 =~ m|^/c=fr/o=cnrs/ou=UPS836|oi) { exit 0; }

# Authentication failed -- Either we could not parse
# the X509 subject string, or the common dn in the
# subject string didn't match ours
exit 1;
}
exit 0;

```

Vérifier ce script et le mettre sous le chemin `/etc/openvpn/script`

Puis spécifier la directive dans le fichier de configuration du serveur :

```
tls-verify /script/verify-dn
```

Pour utiliser cet exemple et la redéfinition de la racine du serveur (par la directive `chroot`), vous devez impérativement copier la commande PERL ainsi que toutes ses librairies (cf. [emprisonner le serveur](#)) dans le chemin 'chrooté'.

Fichier `server.conf`

```

#Fichier de configuration de OpenVPN partie SERVEUR

#-----Securite-----
## Affecter l'utilisateur et le groupe openvpn aux droits reduits
user openvpn
group openvpn
## Enfermer le processus openvpn dans le repertoire /etc/openvpn
chroot /etc/openvpn
## Protéger le serveur contre les denis de service
tls-auth /etc/ta.key 0 #fichier a distribuer a tous les clients!
## application d'un script de vérification des droits d'accès
tls-verify "/script/verify-dn"
; client-connect "/script/connect"
#-----Reseau-----
## Mode tunnel niveau 3
dev tun0
## serveur (une seul instance pour plusieurs clients)
mode server
## OpenVPN va assigner la 1ere adresse (.1) a l'extremite locale du tunnel et
## va fournir aux clients une adresse dans ce reseau en incrementant
server 147,173,102,0 255.255.255.0

# client-config-dir /ccd
## MTU de l'interface TUN (On peut trouver la valeur ideal avec mtu-test)
;mtu-test
tun-mtu 1500
## Les sessions TCP etablies dans le tunnel doivent limiter la taille max
## de leur paquet afin qu'apres encapsulation, le paquet UDP resultant ne

```

```

## depasse pas la taille specifiee:
mssfix 1400
## Apres la creation du tunnel, ajouter la route suivante a la table de routage
locale
;route reseau masque passerelle metric # non utilise dans notre exemple
## ifconfig: la 1ere adresse est l'adresse local du tunnel
##          la 2nde est celle distante
persist-key
persist-tun
keepalive 10 60

## A part dans certains cas, il est deconseille d'encapsuler du tcp dans du
## tcp, ce tunnel utilisera udp
proto tcp
port 1194
# Adresse IP de l'interface connectee au reseau publique
;local 147,173,102.163
## Augmente ou decroit le nombre d information donne par OpenVPN sur son
## fonctionnement, en cas de probleme ou pour tester une valeur a 5
## peut etre utile
verb 5
## La commande push est tres pratique puisqu'elle permet entre autre de modifier
## la table de routage des clients:
## Dans notre configuration le serveur Samba est situe sur le reseau 192.168.0.0
## Il faut donc rajouter une route pour le rendre accessible au client
push "route 147,173,102.96 255.255.255.224"
push "route 147,173,102.0 255.255.255.224"

;push "dhcp-option DNS 147,173,102.1"
push "dhcp-option DOMAIN grenoble.urec.cnrs.fr"

# -----Certificats-----
## Cette instance d'openVPN jouera le role de serveur dans l'echange TLS
tls-server
## Parametres Diffie-Hellman
ca      /etc/openvpn/etc/CAs.crt
cert    /etc/openvpn/etc/server.pem
key     /etc/openvpn/etc/server.key # This file should be kept secret
dh      /etc/openvpn/etc/dh1024.pem
## Controle de la liste de revocation
;curl-verify /root/pk/crl/crl.pem #non utilise dans cet environnement de test

# logs
log-append    log/openvpn.log
status       log/openvpn-status.log

```

```
comp-lzo
```

Fichier client.conf

```
client
dev tun
proto tcp
remote serverOpenVPN.cnrs.fr 1194
resolv-retry infinite
nobind
persist-key
persist-tun
pkcs12 "C:\\Program Files\\OpenVPN\\config\\plus.p12"
#ca ca.crt
#cert client.crt
#key client.key
ns-cert-type server
;tls-client
# Protéger le serveur contre les denis de service
tls-auth "C:\\Program Files\\OpenVPN\\config\\ta.key" 1
comp-lzo
pull
verb 11
```

6 Annexes

6.1 Bibliographie

Site officiel d'OpenVPN : <http://openvpn.net/>

Graphical User Interface for Windows (Mathias Sundman version) : <http://openvpn.se/>

Article « Installation Sécurisée d'OpenVPN » (A. de Bignicourt) :
<http://www.urec.cnrs.fr/IMG/pdf/articles.06.openvpn.pdf>

6.2 A Guide to basic RSA Key Management

A titre d'information, ce mini « how-to » permet de créer votre autorité de certification et de délivrer des certificats si vous ne pouvez demander un certificat. Vous pouvez le consulter sur Internet :

<http://openvpn.net/easyrsa.html>

1. Edit vars.
2. Set KEY_CONFIG to point to the openssl.cnf file included in this distribution.
3. Set KEY_DIR to point to a directory which will contain all keys, certificates, etc. This directory need not exist, and if it does, it will be deleted with `rm -rf`, so BE CAREFUL how you set KEY_DIR.
4. (Optional) Edit other fields in vars per your site data. You may want to increase KEY_SIZE to 2048 if you are paranoid and don't mind slower key processing, but certainly 1024 is fine for testing purposes. KEY_SIZE must be compatible across both peers participating in a secure SSL/TLS connection.
- 5 . vars
6. ./clean-all
7. As you create certificates, keys, and certificate signing requests, understand that only .key files should be kept confidential. .crt and .csr files can be sent over insecure channels such as plaintext email.
8. You should never need to copy a .key file between computers. Normally each computer will have its own certificate/key pair.

Build your own root certificate authority (ca) certificate/key

1. ./build-ca
2. ca.crt and ca.key will be built in your KEY_DIR directory

Build an intermediate certificate authority certificate/key (optional)

1. ./build-inter inter

```
2. inter.crt and inter.key will be built in your KEY_DIR
   directory and signed with your root certificate.
```

Build diffie-hellman parameters (necessary for the server end of a ssl/tls connection).

```
1. ./build-dh
```

Build a certificate signing request

(If you want to sign your certificate with a root certificate controlled by another individual or organization, or residing on a different machine).

1. Get ca.crt (the root certificate) from your certificate authority. Though this transfer can be over an insecure channel, to prevent man-in-the-middle attacks you must confirm that ca.crt was not tampered with. Large CAs solve this problem by hardwiring their root certificates into popular web browsers. A simple way to verify a root CA is to call the issuer on the telephone and confirm that the md5sum or shasum signatures on the ca.crt files match (such as with the command: "md5sum ca.crt").
2. Choose a name for your certificate such as your computer name. In our example we will use "mycert".
3. ./build-req mycert
4. You can ignore most of the fields, but set "Common Name" to something unique such as your computer's host name. Leave all password fields blank, unless you want your private key to be protected by password. Using a password is not required -- it will make your key more secure but also more inconvenient to use, because you will need to supply your password anytime the key is used. NOTE: if you are using a password, use ./build-req-pass instead of ./build-req
5. Your key will be written to \$KEY_DIR/mycert.key
6. Your certificate signing request will be written to \$KEY_DIR/mycert.csr
7. Email mycert.csr to the individual or organization which controls the root certificate. This can be done over an insecure channel.
8. After the .csr file is signed by the root certificate authority, you will receive a file mycert.crt (your certificate). Place mycert.crt in your KEY_DIR directory.
9. The combined files of mycert.crt, mycert.key, and ca.crt can now be used to secure one end of an SSL/TLS connection.

Sign a certificate signing request

```
1. ./sign-req mycert
2. mycert.crt will be built in your KEY_DIR
   directory using mycert.csr and your root CA
   file as input.
```

Build and sign a certificate signing request using a locally installed root certificate/key

This script generates and signs a certificate in one step, but it requires that the generated certificate and private key files be copied to the destination host over a secure channel.

1. ./build-key mycert (no password protection)
2. OR ./build-key-pass mycert (with password protection)
3. OR ./build-key-pkcs12 mycert (PKCS #12 format)
4. OR ./build-key-server mycert (with nsCertType=server)
5. mycert.crt and mycert.key will be built in your

KEY_DIR directory, and mycert.crt will be signed by your root CA. If ./build-key-pkcs12 was used a mycert.p12 file will also be created including the private key, certificate and the ca certificate.

IMPORTANT

To avoid a possible Man-in-the-Middle attack where an authorized client tries to connect to another client by impersonating the server, make sure to enforce some kind of server certificate verification by clients. There are currently four different ways of accomplishing this, listed in the order of preference:

- (1) Build your server certificates with the build-key-server script. This will designate the certificate as a server-only certificate by setting **nsCertType=server**. Now add the following line to your client configuration:

ns-cert-type server

This will block clients from connecting to any server which lacks the **nsCertType=server** designation in its certificate, even if the certificate has been signed by the CA which is cited in the OpenVPN configuration file (**--ca** directive).

- (2) Use the **--tls-remote** directive on the client to accept/reject the server connection based on the common name of the server certificate.
- (3) Use a **--tls-verify** script or plugin to accept/reject the server connection based on a custom test of the server certificate's embedded X509 subject details.
- (4) Sign server certificates with one CA and client certificates with a different CA. The client config "ca" directive should reference the server-signing CA while the server config "ca" directive should reference the client-signing CA.

NOTES

Show certificate fields:
openssl x509 -in cert.crt -text

Exporter un certificat au format pkcs12:
openssl pkcs12 -export -inkey \$dir/tmp/OPERATOR.key -in \$dir/tmp/OPERATOR.pem -out \$dir/tmp/OPERATOR.p12 -passin pass:\$ret{PASS} -name "CNRS-PKI manager" -certfile \$dir/conf/ssl.crt/ca-bundle.crt

6.3 Convertir un certificat PEM ↔ p12

cert.p12 : le nom de fichier comportant le certificat au format pkcs12

userkey.pem : le nom de fichier comportant la clé publique

privatekey.pem : le nom de fichier comportant la clé privée

Convertif PEM → p12

Utiliser la commande openssl

```
openssl pkcs12 \  
-export \  
-out cert.p12 \  
-name "My certificate" \  
-inkey ~user/userkey.pem \  
-in ~user/usercert.pem
```

Convertir p12 → PEM

Pour obtenir la clé privée :

```
openssl pkcs12 \  
-nocerts \  
-in cert.p12 \  
-out ~user/userkey.pem
```

Pour obtenir la clé publique :

```
openssl pkcs12 \  
-clcerts \  
-nokeys \  
-in cert.p12 \  
-out ~user/usercert.pem
```

Changer la passphrase de la clé privée :

```
openssl rsa -in ~user/userkey.pem -des3
```

openssl vous demandera :

1. votre ancien mot de passe
2. votre nouveau mot de passe
3. confirmation du nouveau mot de passe