M2-RADIS Rezo
TP13 : VPN

Jean Saquet, Davy Gigan
Jean.Saquet@unicaen.fr, Davy.Gigan@unicaen.fr

15/01/2013

1 Introduction

Le but de ce TP est de configurer des clients VPN (avec Openvpn) sur des machines virtuelles, avec une architecture réseau permettant de mettre en évidence l'ajout d'accès à certains services obtenu par l'ouverture du tunnel VPN. La partie serveur est configurée par les administrateurs système mais pourra être expliquée et montrée par le responsable de TP. On pourra aussi dans un second temps faire jouer le rôle du serveur aux étudiants.

2 Architecture utilisée

La figure 1 résume cette architecture

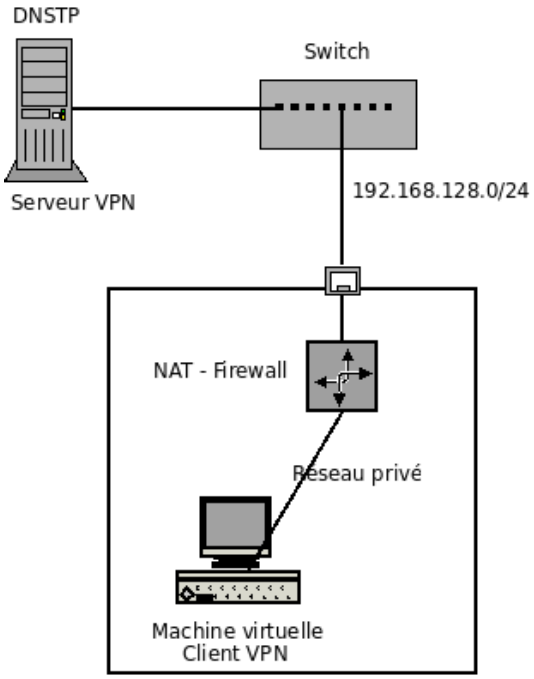

Figure 1 – Notre architecture

Nous utiliserons le réseau de TP de la salle 409, qui comporte :
– un serveur nommé DNSTP en référence à un des services qu'il abrite
– un réseau virtuel sous `marionnet` sur chaque machine réelle de la salle
Rappel : ces machines sont reliées par un switch, les machines de la salle de TP utilisant un pont sur une seconde carte Ethernet eth1, la première étant configurée avec un NAT pour accéder aux services classiques du département et de l'Internet.
Le réseau virtuel de chaque machine (à configurer) utilisé par ce TP devra être constitué de :
– un routeur / NAT / Firewall
– au moins une machine située derrière ce routeur, et qui hébergera le client VPN

L'interface externe du NAT pourra obtenir son adresse (v4) par le DHCP de DNSTP ou manuellement en respectant le préfixe `192.168.128.0/24` et un numéro de machine dans l'intervalle `[2-127]`.
Les autres adresses, toujours IPv4, sont libres, et seront donc masquées par le NAT.

Le firewall sera configuré assez "méchamment" pour ne pas laisser passer grand chose (mais quand même au moins l'accès au serveur VPN situé sur DNSTP !) et au port 8080 servant à échanger les certificats 3.1, plus les pings, le dns ...
Toutefois, on pourra mettre en place ce firewall après les premiers essais ...
Pour configurer le firewall/NAT, il faudra utiliser `iptables` avec à la fois une règle de NAT et des règles de filtrage ne laissant passer que peu de choses. Refuser par défaut les sorties sur l'interface extérieure, mais autoriser le port 1194 en tcp, le port 8080 en tcp (voir plus loin pourquoi), le DNS en udp et tcp, les pings de icmp. Ce sera l'occasion de réviser le TP 5 de filtrage ... Au cas où ce serait parti trop loin, on pourra envisager de recourir à un fichier-type de règles gracieusement fourni par les sysadmins ...

3 Configuration du client VPN

Dans le répertoire `/root/vpn` des machines `marionnet`, il y a des fichiers qui peuvent vous aider à configurer votre client VPN.

3.1 Création du certificat client

Il faut faire une requête qu'on adresse à l'autorité de certification.

Lors de l'exécution de la commande suivante, vous aurez quelques paramètres à spécifier. Pensez à mettre un `CommonName` cohérent (par exemple votre numéro d'étudiant). Votre clef privée ne sera pas protégée par un mot de passe pour faciliter le TP. Cependant, Si quelqu'un vous vole le fichier `NUMETU.key`, il aura accès aux mêmes connexions que vous via le serveur VPN.

```
openssl req -newkey rsa:2048 -keyout NUMETU.key -nodes \
        -out NUMETU.csr -config /root/vpn/openssl.cnf
chmod 600 NUMETU.key
```

Vous pourriez vouloir protéger votre clef privée, dans ce cas vous utilisez la même commande sans `-nodes`. D'autre part, il est possible de changer ou de supprimer la passphrase avec la commande adéquate d'openssl (`man rsa`).

Il faut ensuite fournir votre certificat à l'autorité pour signature, avec `woof` ou `netcat` selon votre goût sur le port 8080 (d'où la nécessité d'avoir laissé passer les messages sur ce port).

3.2 Vérification et signature du certificat

L'autorité de certification (l'enseignant du TP sur le serveur `dnstp`) vérifie le contenu de votre requête et la signe si son contenu lui convient. Il faut particulièrement faire attention à l'unicité du paramètre `CommonName`. Pour afficher le contenu du fichier `req` :

```
openssl req -text -noout -in NUMETU.csr
```

Si le contenu de ce fichier convient à l'enseignant de TP, il peut le certifier avec la commande suivante :

```
sudo vpn-validate NUMETU.csr
```

Vous n'avez plus qu'à récupérer votre certificat ainsi crée par la méthode de votre choix.
Vous devez copier dans le répertoire /etc/openvpn :
– le fichier ça.crt depuis /root/vpn
– votre fichier .key
– le certificat récupéré, renommé en <votre id>.crt

3.3 Paramétrage du client

Recopier le fichier client.conf de la distribution openvpn dans /etc/openvpn.
`cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf .`

Il faut mettre des paramètres en accord avec la configuration du serveur. Les réglages qui *doivent* être communs :
– numéro de port (1194 par défaut)
– protocole TCP
– configuration réseau routé (tun)
– compression lzo activée
Le fichier de configuration du client pourrait ressembler à ceci :

```
client
dev tun
proto tcp
remote 192.168.128.1 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
cert client.crt // remplacer client par votre id
key client.key // idem
ns-cert-type server
comp-lzo
verb 3
```

3.4 Test de connexion et vérification de l'accès à certains services

Pour lancer le client VPN, vous pouvez utiliser la ligne de commande `openvpn --config <file>` ou le script de démarrage situé dans `/etc/init.d`. Pour le second cas, il faut éditer de manière adéquat le fichier `/etc/default/openvpn`. Dans tous les cas, faites attention de ne pas initier plusieurs connexions au serveur en parallèle (ou alors faites le exprès pour en observer les effets).

Vérifier que le tunnel VPN est établi (avec `ifconfig`), re-router le trafic via le tunnel sans perdre l'accès à ce tunnel (sauf si le serveur a modifié lui-même la table de routage - instruction redirect-gateway), puis tester l'accessibilité au serveur web sur le port 80 d'une machine extérieure qui est normalement interdit par le firewall.

4 Configuration du serveur VPN

Ce qui suit a été fait sur la machine `dnstp` pour vous permettre de faire la première ce TP. Lors du TP12, c'est de cette partie dont vous aurez besoin pour la reproduire sur votre machine virtuelle, le routeur de préférence, ça évitera une redirection de port. Pensez à autoriser l'accès au port 1194 ou autre en udp ou tcp selon votre choix.

4.1 Public Key Infrastructure

Le service `OpenVPN` est avantageusement utilisé avec une PKI (Infrastructure à clefs publiques). Pour mettre en place cette infrastructure « simplement », `OpenVPN` met à disposition un outil (`Easy RSA` et une documentation) [1]

4.1.1 Mise en place de l'architecture des certificats

```
# récupération
cp -a /usr/share/doc/openvpn/examples/easy-rsa/2.0 /etc/openvpn/easy-rsa
cd /etc/openvpn/easy-rsa
# configuration
vi vars

export KEY_DIR=/etc/openvpn/keys
export KEY_COUNTRY=fr
export KEY_PROVINCE="bn"
export KEY_CITY="caen"
export KEY_ORG="tpvpn"
export KEY_EMAIL="whatever@example.org"

# chargement des variables dans le shell courant
source vars
# nettoyage (À FAIRE LA PREMIÈRE FOIS SEULEMENT)
./clean-all
# création de l'autorité de certification
./build-ca
# génération des paramètres Diffie-Hellmann
./build-dh
```

À ce point des opérations, une architecture de gestion de clefs existe dans le répertoire qui a été configuré.

```
ca.crt
ca.key
dh1024.pem
index.txt
index.txt.attr
serial
```

1. `http://openvpn.net/index.php/open-source/documentation/howto.html#pki`

4.1.2 Génération du certificats serveur

Le certificat du serveur :

```
./build-key-server <FQDN_du_serveur>
```

Ne mettez pas de challenge password. Lorsqu'on vous le demande, vous acceptez les 2 propositions *sign, commit*. À ce point, on a dans le répertoire `KEY_DIR` tout ce qu'il faut pour mettre en place le serveur VPN :

```
ca.crt
server.key
server.crt
dh1024.pem
```

4.1.3 Génération des certificats client

Il est possible d'utiliser les scripts `build-key` et `build-key-pass` pour créer les clefs et les certificats des clients *simplement*, cependant il est plus sécurisé d'utiliser une procédure non centralisée :
– créer la clef privée sur le client
– créer la requête de signature de certificat sur le client
– transférer la requête au serveur
– signer le certificat sur le serveur
– renvoyer le certificat signé au client
C'est ce que vous avez fait dans la section 3.1. Si vous avez compris l'intérêt de cette procédure, vous saurez choisir entre la version avec `-pass` ou non de la création de certificat client. Puis vous transfèrerez les fichiers `key` et `crt` au client (ssh entre les machines marionnet ?).

4.2 Configuration du serveur

Les chemins des fichiers sont relatifs à `/etc/openvpn` par défaut mais vous pouvez indiquer des chemins complets. Par exemple pour rendre accessible les logs et le statut des connexions à d'autres utilisateurs que `root`. Les commentaires commencent par `#` ou `;`.

```
local IP_SERVER
proto tcp
ca keys/ca.crt
cert keys/server.crt
key keys/server.key
dh keys/dh1024.pem
server 192.168.X.0 255.255.255.0
;client-to-client
;tls-auth ta.key 0
user nobody
group nogroup
status /var/log/openvpn-status.log
log-append  /var/log/openvpn.log
```

Il reste à lancer le serveur (`openvpn --config <file>`)
Les fichiers intéressants crées par le serveur en cours d'execution sont les fichiers `openvpn-status.log` et `openvpn.log`. L'enseignant de TP pourra vous en montrer le contenu avec les commandes :

```
sudo vpn-status
sudo vpn-log
sudo vpn-tcpdump
```

4.3 Configuration du client

Vous devez savoir le faire, c'est dans la partie 3.3.

4.4 Pour aller plus loin (pas forcément testé ...)

4.4.1 tls-auth

Activer le paramètre tls-auth et vérifier que sans la présence de cette clef côté client, le serveur jette très vite la connexion. Attention, la ligne de configuration contient un bit qui doit être différent entre le serveur et ses clients.

Si vous souhaitez tester ce paramètre sur le serveur `dnstp`, il y a un serveur `udp/1194` qui utilise le `ta.key` présent dans le `/root/vpn` des machines marionnet.

4.4.2 Communication inter-client

Tentez d'interdire / d'autoriser (en fonction de ce qui fonctionne de base dans votre config) les communication entre les clients du service VPN.

Un client VPN donné est joignable par le bout de son adresse de tunnel (à voir dans le résultat de la commande ifconfig).

Discutez les avantages et inconvénients de la communication inter-client.

4.4.3 Tenter d'usurper le serveur

Tentez de configurer un serveur VPN avec votre certificat client puis connectez vous sur ce serveur avec un autre client. Comme votre type de certificat est signé, le client pourrait vous faire confiance mais comme vous avez fait attention de faire un certificat de type *server*, le client devrait jeter la connexion. Si ce n'est pas le cas, vous avez un gros souci de sécurité car n'importe quel client peut se faire passer pour un serveur valide.