VPN

- Réseau privé virtuel
- Usages :
  - fournir l'accès à des ressources internes aux clients nomades
  - relier 2 réseaux ”d'entreprise” (sites distants par ex, ou relier 2 labos de maths ;)
  - (”contourner” des sécurités)

Implémentations

- IPSec (NAT ?)
- SSH (depuis peu) (restrictif, compliqué)
- PPTP (sécurité ?)
- Tunnels SSL :
  - Stunnel (point-à-point uniquement) (limité...)
  - OpenVPN

OpenVPN

- Léger (un seul démon ”userspace”)
- Facile (un fichier de configuration)
- Portabilité (Linux, Solaris, OpenBSD, FreeBSD, NetBSD, Mac OS X, Windows 2000/XP)
- Intégration (UDP/TCP/proxys ...)
- Flexibilité (routage IP de l'OS, configuration dynamique, ...)

OpenVPN

Basé sur

- OpenSSL :
  - Support de TLS, de nombreux cipher de crypto, portabilité ...
  - Sécurité, fiabilité
- TUN :
  - Délègue la gestion des interfaces réseaux virtuelles au système d'exploitation
- Plugins d'authentification :
  - Faciles à développer (PAM, LDAP, ...)

Modes de fonctionnement

- Routage
  - Au niveau IP, routage de réseaux
  - Les protocoles broadcastés ne fonctionnent pas (smb entre autres)
- Pont (bridge)
  - Niveau ethernet
  - Véritable intégration au réseau hébergeur
  - Beaucoup de ”bruit”, attention à la bande passante

Réseau

- UDP par défaut (port 1194)
  - ”léger” et donc
  - performant
  - passe mal dans les réseaux fermés (UDP étant souvent tout simplement fermé)
- TCP possible
  - plus lourd
  - plus facile à déployer (on trouve facilement des ports TCP ouverts dans les filtrages ;)
  - supporte les proxys HTTP

Authentification

- 2 modes :
  - certificats X509 (intégrés ou non à un PKI existant, par exemple il serait possible d'utiliser les certificats personnels CNRS)
  - mots de passe (PAM)

=> extensible via plugins simples à développer

Installation

- Linux : distribué par toutes les bonnes distributions en standard
- Windows : package sur http://openvpn.se (un simple exe)
- MacOS : TunnelBlick http://tunnelblick.net
- BSD : ports
- ...

Configuration

- Serveur
  - 1 fichier de configuration par VPN (*.conf, ovpn sous windows)
  - 1 répertoire ”ccd” pour la configuration 'dynamique' des clients (on ne redémarre pas le démon)
  - les certificats pour crypter/authentifier
- Client
  - 1 fichier de configuration .conf (ou .ovpn)
  - Les certificats et clés fournis par l'administrateur du site VPNisé

Certificats

- OpenVPN fournit un package ”clés-en-main” pour générer :
  - 1 autorité de certification (si on en n'a pas une)
  - les certificats/clés du serveur
  - le certificat/clé de chaque client (on peut utiliser le même cert/clé pour tous les clients, mais par précaution, il vaut mieux identifier chaque client)

Exemple de configuration (serveur)

/etc/openvpn/monvpn.conf:

```
local ip_publique_serveur
port port_d'ecoute
proto udp # (ou tcp)
dev tun
ca ca/ca.crt
cert certs/vpn.crt
key keys/vpn.key  # secret
dh dh1024.pem
server 10.20.0.0 255.255.255.0 #le réseau du VPN
client-config-dir ccd #le répertoire de configuration des clients
route 10.20.0.0 255.255.0.0 #un réseau qu'on va router (celui des clients donc)
keepalive 10 120
tls-auth ta.key 0 # secret
comp-lzo #compression ”standard”
persist-key
persist-tun
status openvpn-status.log
verb 4
mute 10
management 127.0.0.1 7505
#plugin /usr/lib/openvpn/openvpn-auth-ldap.so /etc/openvpn/auth-ldap.config
```

Exemple de configuration (serveur - ccd)

- 1 fichier par client :

```
#choix de l'IP du client
ifconfig-push 10.20.1.1 10.20.1.2
#serveurs qui seront 'visibles' par le VPN pour le client
push "route 193.54.241.13 255.255.255.255"
push "route 147.210.16.7 255.255.255.255"
```

- Attention : openvpn ne gère pas la sécurité réseau. Vous devez créer vos filtrages réseaux sur le serveur. Par ex, on ”push” des routes sur les clients, mais le client peut également modifier sa table de routage en shell et accéder à des serveurs/réseaux via le VPN si vous n'avez pas filtrer correctement ces accès.

Exemple de configuration (client)

```
client
dev tun
proto udp
remote serveur_vpn port_vpn
resolv-retry infinite
nobind
persist-key
persist-tun
#http-proxy mon_proxy_web 3128 #optionnel
mute-replay-warnings
ca keys/ca.crt
cert keys/userlogin.crt
key keys/userlogin.key
ns-cert-type server
tls-auth keys/ta.key 1
comp-lzo
verb 3
mute 20
auth-user-pass
```

Options

- Routage de tout le traffic du client dans le tunnel (push "redirect-gateway")
- Reconfiguration dynamique du firewall (”learn-address mon_script.sh”)
- Routage d'un réseau se trouvant derrière un client (”iroute a.b.c.d w.x.y.z”)

Réseaux IPs

- En mode routage, VPN avec IPs privés

=> NAT et MASQUERADING sur la passerelle VPN

La PLM et les nomades

- Cas d'usage :

  Un chercheur dans un colloque qui a une connexion wifi pour surfer sur le web mais où tout le reste est bloqué. Impossible de lire/envoyer ses mails avec thunderbird, impossible de se connecter en ssh, ...

- Solution grace à la PLM (!) :

  Le chercheur lance son client VPN, il est alors connecté aux serveurs de la PLM (mail imap/pop, smtps, jabber, jetons,...) et il peut se logger en ssh sur le SAS de son laboratoire.

- Et ce n'est qu'un début ...

VPN @PLM : Principes (1/2)

- Chaque responsable de laboratoire transmet à l'équipe support les serveurs (de son labo) auxquels il souhaite que ses utilisateurs aient accès via le VPN (SAS SSH, mails internes, ...).
- Chaque usager de la PLM peut ensuite se connecter sur https://webmail.math.cnrs.fr, puis se rendre sur https://webmail.math.cnrs.fr/horde/vpn pour créer son ”package” VPN (un pack contenant tous les fichiers de configuration)

VPN @PLM : Principes (2/2)

- Serveur vpn.math.cnrs.fr, sur le port 443, TCP

=> très grande accessibilité (si le web marche, le VPN marchera)

- On ne route pas ”tout le traffic” par le VPN : accord avec le CRI de Bordeaux, le VPN ne sert qu'à relier les labos de maths entre eux
- 1 labo = 1 subnet (/24) (si besoin plusieurs /24)
- 1 utilisateur = 1 IP fixe (privée)

VPN @PLM : Futur ...

- Tableau virtuel ?
- Vidéoconférence ?
- Partage de fichiers ?
- ...