VPN
Virtual Private Network

Olivier ROUSSEL
olivier.roussel@univ-artois.fr

VPN

- Un réseau virtuel privé (virtual private network) est l'interconnection de réseaux privés par l'intermédiaire de réseaux publics en utilisant des tunnels cryptés.
- On obtient ainsi un réseau qui est globalement **privé** puisqu'un tiers ne peut accéder aux données qui circulent sur le réseau.
- Ce réseau est **virtuel** car concrètement, on utilise des réseau publics.
- Le but est d'assurer la sécurité des informations échangées sur ce réseau en évitant le coût d'une liaison spécialisée sécurisée (si toutefois cela est possible)
- La protection des échanges doit être transparente pour l'utilisateur.

Avantages/inconvénients d'un VPN

Avantages

- renforcement de la sécurité :
  le VPN permet de distinguer les postes situés sur un autre site de l'entreprise (postes mobiles compris) d'une machine quelconque de l'internet. On peut donc affiner la politique de sécurité.
- regroupement de la gestion du réseau
  au lieu d'avoir une équipe par site, la gestion du réseau peut se faire de manière centralisée grâce au VPN.
- transparence vis à vis de l'utilisateur

Inconvénients

- le risque interne augmente
  votre réseau interne s'élargit et contient de plus en plus d'utilisateurs. Pouvez-vous avoir confiance en chacun d'eux (en particulier les postes mobiles) ?
- le VPN devient une ressource critique de l'entreprise

Positionnement du serveur VPN

- sur le pare-feu
  augmente la charge de la machine et affaiblit la sécurité du pare-feu
- en parallèle avec le pare-feu
  le serveur VPN devient pour un pirate un moyen potentiel de contourner le pare-feu
- derrière le pare-feu
  le pare-feu doit laisser passer un protocole crypté ce qui limite les vérifications qu'il peut faire

Construire un VPN

Sur des protocoles standards

- SSH et PPP
- SSL/TLS et PPP
- PPTP
- IPSec

Sur des protocoles non standards

- OpenVPN
- ...

Le protocole PPP (Point to Point Protocol)

- est un protocole de la couche liaison
- permet d'échanger des datagrammes IP entre deux machines (ainsi que des PDU d'autres protocoles)
- contient différents sous-protocoles
  - Link Control Protocol (LCP) : établit, teste, configure et ferme une liaison
  - Network Control Protocol (NCP) : établit, configure et termine l'utilisation de la liaison pour un protocole donné (IP, ...)
  - compression : ne pas transmettre les informations fixes des en-têtes IP (adresses des 2 extrémités) ou transmettre uniquement les différences entre les en-têtes (permet de passer de 40 octets à 3,4 ou 5 octets)

Authentification PPP

- Password Authentication Protocol (PAP) :
  l'identifiant et le mot de passe sont envoyés en clair sur la liaison (une seule fois)
- Challenge Handshake Authentication Protocol (CHAP) :
  - La machine A transmet son identifiant en clair
  - La machine B lui envoie un mot aléatoire
  - La machine A renvoie le hachage du mot aléatoire concaténé avec son mot de passe
  - La machine B vérifie la réponse
  - L'authentification est répété régulièrement au cours de la liaison
- Extensible Authentication Protocol (EAP) :
  le protocole d'authentification est choisi dans une liste commune de protocoles

PPP via SSH

- ssh sert à mettre en place un tunnel crypté, ppp permet le passage de n'importe quel trafic IP par ce tunnel
- il faut créer un utilisateur sur le serveur VPN (et éventuellement utiliser les clefs SSH pour se connecter sans mot de passe : ssh-keygen et ~/.ssh/authorized_keys)
- il faut configurer sudo pour que cet utilisateur puisse lancer pppd sans mot de passe : visudo /etc/sudoers

```
Cmnd_Alias VPN=/usr/sbin/pppd
TheUser    ALL=NOPASSWD: VPN
```

- il faut activer le routage : sysctl -w net.ipv4.ip_forward=1

PPP via SSH (suite)

- il faut lancer pppd de chaque cote :

```
sudo /usr/sbin/pppd updetach noauth
  pty "sudo -u TheUser ssh -t -t TheServer
  sudo pppd noauth
  192.168.254.254:192.168.254.253"
```

- il faut mettre en place les routes depuis chaque machine :

```
route add -net TheRemoteNetwork gw TheLocalPPPAd
```

- On peut éventuellement procéder à une nouvelle authentification avec PPP (est-ce bien utile ?)

PPP via SSH (suite)

- On peut restreindre l'accès à SSH en rajoutant devant la clef correspondante dans ~/.ssh/authorized_keys2

```
from "client.domaine.fr",
command="sudo pppd noauth
  192.168.254.254:192.168.254.253",
no-port-forwarding,
no-X11-forwarding,
no-agent-forwarding
```

- Veiller à laisser le moins de liberté dans la configuration du démon sshd

PPP via SSL

- Le principe général reste le même : stunnel sert à mettre en place un tunnel crypté, ppp permet le passage de n'importe quel trafic IP par ce tunnel
- il faut créer un utilisateur sur le serveur VPN et des certificats pour le client et le serveur.
- il faut configurer sudo pour que cet utilisateur puisse lancer pppd sans mot de passe : visudo /etc/sudoers

```
Cmnd_Alias VPN=/usr/sbin/pppd
TheUser    ALL=NOPASSWD: VPN
```

- il faut activer le routage : sysctl -w net.ipv4.ip_forward=1

PPP via SSL (suite)

- sur le serveur :

```
stunnel -v3 -a TheCertDirectory -v3
  -p TheServer.pem -f -D7
  -d ThePort -L sudo pppd pppd
  debug noauth
  192.168.254.254:192.168.254.253
```


PPP via SSL (suite)

- sur le client :

```
sudo pppd updetach debug noauth
  pty "sudo -u TheUser
  stunnel -v3 -a TheCertDirectory -S0
  -p TheClient.pem -f -D7
  -c -r TheServer:ThePort"
```

- il faut mettre en place les routes depuis chaque machine :

```
route add -net TheRemoteNetwork gw TheLocalPPPAd
```


PPTP (Point to Point Tunneling Protocol)

- développé par Microsoft
- les paquets sont encapsulés par PPP, puis encapsulés dans des paquets GRE (Generic Routing Encapsulation, protocole 47) et acheminé à destination
- un canal de contrôle TCP est ouvert vers le port 1743 du serveur
- natif sous Microsoft Windows
- sous linux : http://pptpclient.sourceforge.net
- attention à la version du protocole utilisée : la v1 comporte de multiples défauts

OpenVPN (openvpn.net)

- solution VPN libre et multi-plateforme (unix, windows,...) basée sur SSL/TLS
- fonctionne en mode routeur (interface tun) ou en mode bridge (interface tap). Le mode bridge permet en particulier de laisser passer le broadcast.
- encapsule le traffic dans des messages UDP (de préférence) ou dans une communication TCP

Exemple de configuration OpenVPN

Sur le routeur du réseau A

```
daemon

proto tcp-server
lport PORT_NUM

dev tun
ifconfig 10.0.0.1 10.0.0.2
route NETWORK_B NETMASK_B

persist-key
persist-tun
persist-local-ip

user nobody
group nobody
```


Exemple de configuration OpenVPN (2)

Sur le routeur du réseau B

```
daemon

proto tcp-client
remote ROUTEUR_A
rport PORT_NUM

dev tun
ifconfig 10.0.0.2 10.0.0.1
route NETWORK_A NETMASK_A

persist-key
persist-tun
persist-local-ip

user nobody
group nobody
```


OpenVPN : protection par chiffrement symétrique

Générer une clé secrète par

```
openvpn --genkey --secret FichierContenantLeSecret.secre
```

La copier par un canal sécurisé (scp) et ajouter dans la configuration des deux routeurs

```
secret FichierContenantLeSecret.secret
```


OpenVPN : protection par chiffrement asymétrique

Sur le routeur du réseau A

```
tls-server

# Diffie-Hellman (tls-server uniquement)
dh dh1024.pem

# certificat CA
ca ca.crt

# certificat de A
cert A.crt

# clé privée de A
key A.key
```


OpenVPN : protection par chiffrement asymétrique (2)

Sur le routeur du réseau B

```
tls-client

# certificat CA
ca ca.crt

# certificat de B
cert B.crt

# clé privée de B
key B.key
```

IPsec

- IPsec est une extension du protocole IP permettant de sécuriser les datagrammes. Il fonctionne au niveau datagramme et est donc adapté aux protocoles non connectés.
- La protection est transparente du point de vue des applications.
- Il propose deux modes de fonctionnement :
  - le mode transport
    le datagramme à sécuriser est modifié en intercalant les en-têtes IPsec entre l'en-tête IP et les données
  - le mode tunnel
    le datagramme à sécuriser est encapsulé dans un nouveau datagramme qui contient un en-tête IPsec

En-têtes IPsec

En-tête d'extension en IPv6 ou options en IPv4

- en-tête AH (Authentication Header)
  il fournit
  - authentification
  - intégrité des messages

  Il correspond au protocole IP de numéro 51. Un datagramme AH contient un en-tête IP, un en-tête AH puis les données. Il assure l'authentification et l'intégrité de tout le datagrame (en-tête IP inclus)

En-têtes IPsec (suite)

- en-tête ESP (Encapsulating Security Payload)
  il fournit
  - les avantages d'AH
  - confidentialité

  Il correspond au protocole IP de numéro 50. Un datagramme ESP contient un en-tête IP, un en-tête ESP, les données et un en-queue ESP. Il assure la confidentialité des données et d'une partie de l'en-queue et l'authentification/intégrité de l'en-tête ESP et de ce qui le suit.

Association de Sécurité (SA)

- une SA est un accord entre deux hôtes sur la manière d'assurer la sécurité (AH ou ESP, avec quel algo et quelles clefs). Il ne faut pas oublier qu'IPsec fonctionne aussi pour les protocoles non connectés, on peut grossièrement considérer que cela remplace la négociation des protocoles cryptographiques dans SSL.
- il doit y avoir une SA pour les deux sens de la communication (ce qui permet d'utiliser des algorithmes différents dans un sens et dans l'autre)
- une SA indique en fait comment protéger les datagrammes
- chaque SA est identifiée par un numéro SPI (Security Parameter Index). Le SPI est transmis dans le datagramme IPsec.
- Les SA sont stockées dans la SAD (Security Association Database)

Politique de sécurité

- La politique de sécurité (SP, Security Policy) détermine si l'on doit protéger les datagrammes ou pas.
- Elle référence en fait indirectement des SA
- La politique de sécurité indique s'il faut protéger ou pas, l'association de sécurité indique comment protéger le cas echéant.
- La politique de sécurité est stockée dans la SPD (Security Policy Database)

Etablissement automatique d'une SA

Mettre en place à la main des SA pose problème

- il faut le faire pour chaque paire de machines et dans les deux sens
- il faut choisir une clef secrète différente à chaque fois et réussir à la transmettre de manière sécurisée sur chacune des machines
- il faut choisir des protocoles communs
- les clefs doivent être changées régulièrement

On utilise donc un service annexe qui se charge de négocier automatiquement le choix des SA. Ce service utilise le protocole IKE (Internet Key Exchange) et permet de se mettre d'accord sur les protocoles cryptographiques à utiliser et sur les clefs symétriques à utiliser (négotiation dans un canal sécurisé par de la cryptographie asymétrique).

Implantation

- Intégrée à Windows 2000
- Grâce a FreeS/WAN pour un noyau linux 2.4
- Intégré au noyau linux 2.6 (package ipsec-tools : http://ipsec-tools.sourceforge.net), voir le howto http://lartc.org/howto/lartc.ipsec.html

Les outils sous Linux

- setkey
  permet de manipuler la SAD et la SPD
  setkey -c permet de saisir les commandes interactivement
  setkey -f lit les commandes depuis un fichier
  les commandes doivent être terminées par un point-virgule
- racoon
  uns fois lancé, ce client/serveur permet de négocier automatiquement des SA

Les commandes de setkey pour les SA

- flush;
- dump;
- add *src dst protoIPsec spi [mode] algorithm*;
  - *src* et *dst* peuvent préciser un numéro de port entre crochets
  - *protoIPsec*: ah, esp ou ipcomp
  - *mode*:
    - -m tunnel
    - -m transport
    - -m any
  - *algorithm*:
    - -E *encryptionAlgo key*
    - -A *authenticationAlgo key*
    - -C *compressionAlgo*

Les algorithmes disponibles (man setkey)

Le nombre de bits de la clef est entre parenthèses

- pour AH : hmac-md5 (128), hmac-sha1 (160), keyed-md5 (128), keyed-sha1 (160), hmac-sha256 (256), hmac-sha384 (384), hmac-sha512 (512), hmac-ripemd160 (160), aes-xcbc-mac (128), tcp-md5 (8 à 640)
- pour ESP : des-cbc (64), 3des-cbc (192), blowfish-cbc (40 a 448), cast128-cbc (40 à 128), des-deriv (64), 3des-deriv (192), rijndael-cbc (128/192/256), twofish-cbc (0 à 256), aes-ctr (160/224/288)
- pour IPComp : deflate

Les commandes de setkey pour les SP

- spdflush;
- spddump;
- spdadd *src dst upperProto policy*
- *upperProto* est un protocole de /etc/protocols ou bien any
- *policy* peut être
  - -P *direction* discard
    ignorer le datagramme
  - -P *direction* none
    transmettre en clair
  - -P *direction* ipsec *protocol/mode/src-dst/level*
    protection IPSec
  - *direction* est in,out ou fwd ; *protocol* est ah,esp ou ipcomp ; *mode* est transport ou tunnel ; *src-dst* sont les extrémités du tunnel (vide pour le mode transport) ; *level* est default, use, require ou unique.

IPsec et le pare-feu

Le pare-feu doit laisser passer les protocoles 50 (ESP) et 51 (AH).

Exemple de configuration

Contexte

On veut dans cet exemple protéger la confidentialité des échanges de A vers le service 'echo' de B (port 7). Pour bien illustrer le mécanisme, on ne protège pas les échanges de B vers A, et on ne protège pas les autres services.
Les requêtes au serveur seront donc chiffrées, mais pas les réponses.
On peut utiliser un sniffer réseau pour vérifier que les communications sont protégées dans un sens, mais pas dans l'autre (tcpdump -n -p -i eth0 -s 100 -X)

Mise en place de SA à la main

Sur A

```
#!/sbin/setkey -f
flush;

add hostA hostB esp 10000 -m transport
    -E blowfish-cbc "leMotDePasse";
```

Sur B

```
#!/sbin/setkey -f
flush;

add hostA hostB esp 10000 -m transport
    -E blowfish-cbc "leMotDePasse";
```


Mise en place des SP

Sur A

```
#!/sbin/setkey -f
spdflush;

spdadd hostA hostB[7] any
       -P out ipsec esp/transport//require ;
```

Sur B

```
#!/sbin/setkey -f
spdflush;

spdadd hostA hostB[7] any
       -P in ipsec esp/transport//require ;
```


Gestion automatique des SA

- on définit toujours la SP, mais on laisse un serveur (racoon) négocier les SA quand le besoin s'en fait sentir
- un intérêt majeur est que les SA (et donc les clefs) sont renégociées régulièrement
- on peut authentifier les serveurs en utilisant un secret partagé mais l'idéal est d'utiliser de la cryptographie asymétrique (donc utilisation de certificat)
- le fichier de configuration est /etc/racoon/racoon.conf

Configuration de racoon avec un secret partagé

```
path include "/etc/racoon";
path pre_shared_key "/etc/racoon/psk.txt";

remote anonymous
{
  exchange_mode main;       doi ipsec_doi;
  situation identity_only;  my_identifier address;
  lifetime time 1 hour;     initial_contact on;
  proposal_check obey;
  proposal
  {
    encryption_algorithm 3des;
    hash_algorithm sha1;
    authentication_method pre_shared_key;
    dh_group 2;
  }
}
```


Configuration de racoon (secret partagé, suite)

```
sainfo anonymous
{
  pfs_group 2;
  lifetime time 1 hour ;
  encryption_algorithm 3des, blowfish 448, rijndael ;
  authentication_algorithm hmac_sha1, hmac_md5 ;
  compression_algorithm deflate ;
}
```


Pre-shared secret (/etc/racoon/psk.txt)

Sur A

```
hostB   LeSecretPartage
```

Sur B

```
hostA   LeSecretPartage
```

Configuration de racoon avec certificats

```
path certificate "/etc/racoon/certs";
remote hostB
{
  exchange_mode main;
  my_identifier asn1dn;
  peers_identifier asn1dn;
  certificate_type x509 "hostA.public" "hostA.private";
  peers_certfile "hostB.public";
  proposal {
    encryption_algorithm 3des;
    hash_algorithm sha1;
    authentication_method rsasig;
    dh_group 2 ;
  }
}
```