**TP 2 :**

Creation de tunnel VPN avec openvpn

Paquetages à installer : openssl, openvpn

I – Preparatifs

1. Situation matérielle

Le but du TP est de créer un lien entre un client et la société Booktic

Utilisez deux machines virtuelles :

Sous Linux : installez les paquetages habituels openvpn(pour le client y compris), openssl

Une fois installé, on placera la machine cliente sur le réseau SIO (VLAN 60 :172.16.19.137)

Installez les paquetages nécessaires : openssl, openvpn, dnsutils, firefox-esr, procps, psmisc,openssh-server

1. La commande openvpn

Expliquez le role des options suivantres (man openvpn)

**Open vpn –remote :**

Connexion à distante via :

host name or IP address, port and protocol

Syntaxe valide :

* Remote host
* Remote host port
* Remote host port proto

Si le client veut se connecter au serveur avec le nom et le port, l’argument proto indique le protocole à utilisé quand on veut se connecter à distance soit : tcp soit udp

Like udp4/udp5/udp6

**--dev :** indique le peripherique du réseau (virtuel)

Tun(correspond aux ipv4)

Tap(802.3)

**--port :**

Numero du port TCP/UDP pour les deux hotes

--proto :

**--verb n**

Régler le niveau de traces à **n** (par défaut=1). Chaque niveau rajoute de l'information au niveau précédent. Le niveau 3 est recommandé si vous souhaitez un bon résumé de ce qu'il se passe, sans être inondé de traces.

**0 --** Pas de trace, sauf erreur fatale.
**1 --** Informations de démarrage + d'initialisation de la connexion + erreurs non-fatales de chiffrement et de réseau.
**2 --** Affiche les dialogues SSL/TLS
**3 --** Eléments SSL/TLS complémentaires + alertes réseau **--gremlin** + changement d'état de la compression dynamique (arrêtée/démarrée).
**4 --** Affiche la valeur de tous les paramètres de réglage.
**5 --** Imprime les lettres **R** et **W** sur la console pour chaque paquet lu *(R)* ou écrit *(W)*. Les majuscules sont utilisées pour les paquets TCP/UPD, les minuscules pour les paquets TUN/TAP.
**6 à 11 --** Affiche des traces de déboguage de plus en plus nombreuses (voyez errlevel.h pour plus d'information sur les niveaux de trace de déboguage).

**--ifconfig :**

Pour les périphériques TUN qui permettent de réaliser des connexions virtuelles de type point à point, **--ifconfig** s'utilise en définissant deux adresses IP privées qui n'appartiennent à aucun sous-réseau en cours d'utilisation.

Pour les périphériques TAP qui permettent de créer des segments ethernet virtuels, **--ifconfig** est utilisé pour définir une adresse IP et un masque de sous réseau, exactement comme une interface ethernet classique

--genkey : Permet de générer une clé avec une direction défini

**--secret fichier [direction]**

Active le mode de chiffrement statique (sans TLS) par secret partagé. Le secret partagé **fichier** doit avoir été généré par l'option **--genkey.**

**--dh fichier**

Fichier au format .pem contenant les paramètres Diffie Hellman (requis pour **--tls-server** seulement). Lancez

**openssl dhparam -out dh1024.pem 1024**

pour générer vos propres paramètres, ou utilisez le fichier dh1024.pem existant dans la distribution d'OpenVPN. Les paramètres Diffie Hellman peuvent être considérés comme des données publiques.

**-ca fichier**

Fichier de l'Autorité de Certification (CA) au format .pem, couramment dénommé certificat *racine*. Le fichier peut contenir plusieurs certificats au format .pem concaténés. Pour créer votre propre certificat racine et sa clé privée, vous pouvez utiliser une commande OpenSSL comme :

**openssl req -nodes -new -x509 -keyout tmp-ca.key -out tmp-ca.crt**

Ensuite, éditez votre fichier de configuration openssl.cnf et faites pointer la variable **certificate** vers votre nouveau certficiat racine **tmp-ca.crt.**

A des fins exclusives de test, la distribution OpenVPN inclut un exemple de certificat racine d'Autorité de Certification (tmp-ca.crt). Vous ne devez absolument jamais utiliser les clés et les certificats distribués en standard avec OpenVPN dans un environnement de production : étant à la libre disposition de quiconque, ces clés ne procurent aucune sécurité.

**--cert**

Pour activer le mode TLS, chaque hôte en vis-à-vis doit posséder en local sa propre paire clé/certificat ( **--cert** et **--key** ), signés par le certificat racine défini par l'option **--ca.**

Lorsque deux hôtes OpenVPN se connectent, chacun présente son certificat à l'autre. Chacun vérifie alors que ce certificat est bien signé par le certificat racine spécifié par **--ca.**

Si c'est bien le cas des deux côtés, la négociation TLS est réussie et les hôtes OpenVPN vont s'échanger des clés temporaires qui sont utilisées pour le transfert des données dans le tunnel.

**-key fichier**

Clé privée au format .pem de l'hôte local. Vous devez utiliser la clé privée qui a été générée lorsque vous avez obtenu le certificat de cet hôte (voir **-cert fichier** ci-dessus).

Commencez par construire un tunnel minimal avec chiffrement symétrique, en utilsiant la commande openvpn en ligne :

D’abord, on commence par un tunnel non-crypté

**2.1 Sur le serveur :**

Openvpn –dev tun0 –verb5 –ifconfig 10.255.0.1 10.255.0.2

**2.2 Sur le client :**

Openvpn –dev tun0 –verb5 –ifconfig 10.255.0.1 10.255.0.2 –remote@172.17.1.70



Depuis le client, pingue l’interface du serveur et observer l’interface eth0 depuis Wireshark ;



Depuis le client, pingue l’interface du serveur et observer l’interface tun0 depuis Wireshark ;



**On va generer une clef symetrique**

Avec la commande :

Openvpn –genkey –secret clefsym

**Transférer la clé symétrique vers le client :**



Refaire les étapes 2.1 et 2.2 pour retrouver l’apparition du chiffrement



**Creation de certificat avec openssl**

Créer des repertoires de travail : /apps/openvpn/keys

/apps/openvpn/conf

/apps/pki-booktic

Nous créerons 3 couples certificats/clés privées :

* L’autorité de certification
* Le serveur VPN
* Le client VPN

Modifiez les variables suivantes dans le fichier /apps/pki-booktic/easy-rsa/vars : configurez les aux valeurs de la société Booktic



Installez notre infrastructure de certification dans /apps/pki-booktic/easy-rsa : **/easyrsa init-pki**

**Dans /apps/pki-booktic/easy-rsa/ :**

Utilisez le script easyrsa pour construire le certificat de l’autorité, répondez aux questions :

/easyrsa build-ca nopass **#afin de ne pas gérer de mot de passe à chaque connexion**

BookticSrv

**De la même façon, creez le certificat serveur et de vos clients :**

**Pour le serveur :**

**./easyrsa gen-req BookticSrv no pass**

**Pour le client :**

**./easyrsa gen-req BookticSrv no pass**

**BookticCltVPN**

**Ensuite il faut ensuite faire signer ces certificats par l’autorité de certification :**

**./easyrsa sign-req server BookticSrv #pour le serveur**

**./easyrsa sign-req client BookticCltVPN #pour le client**

**CltVPN**

**Creez enfin une clé de session de Diffle**

**./easyrsa gen-dh**

**Creez enfin une signature electronique pour authentifier la connexion client-server, creez la directement dans le dossier /apps/openvpn/keys**

**Openvpn –genkey –secret /apps/openvpn/keys/bookticsign.key**

**L’ensemble des clefs,certificats dans le repertoire /apps/pki-booktic/pki et ses sous repertoires issued , priate ainsi que la clé DH vers /apps/openvpn/keys**

**Le repertoire issued, contient tout les certificats qu’il a crée(.crt) (clt.crt)**

**Le repertoire private,contient toutes les clés privés(.key)(clt.key) -> transférer le client**

**Et le ca.crt (et normalement un fichier booktic sign pour faire la validation de la connexion)**

**Le gen req contient une demande (csr) et c’est ici qui est crée le .key**

**Le sign req valide le certificat (et c’est lui qui vient crée le certificat)**

**II Mise en place du tunnel openvpn**

**I Configuration du serveur**

**Nous allons créer un lien pour le fichier conf afin qu’il aille le chercher dans repertoires**

**Créer un fichier de conf dans /apps/openvpn/conffiles/bookticVPN.conf**

**Créer le lien symbolique depuis /etc/openvpn vers ce fichier, le service openvpn cherche les fichiers conf dans /etc/openvpn/ dans notre lien permettra d’aller chercher le notre à son emplacement réel :**

**Ln –s /apps/openvpn/conf/bookticVPN.conf /etc/openvpn/bookticVPN.conf**

****

**Exemple de fichier de conf bookticVPN.conf**

****

****

**Verifiez le processus si il est lancé : ps –axf**

 **Stoppez le service : systemctl stop openvpn**

**Verifiez le fonctionnement dans les logs :**

**Tail –f /apps/openvpn/log/openvpn.log**

****

**2.Configuration client**

**Transférez les certificats et clés privés, clé de connexion (bookticisign.key) et le certificat d’autorité**

**Créer le lien symbolique comme pour le serveur**

****

****

**Verifiez aussi le lien du tunnel via Wireshark :**

****

**Desormais, le client et el Servuer communique via un tunnel mais le client ne peut pas aller sur tout les reseaux Booktic Pour cela, votre serveur VPN devra devenir routeur et faire du NAT**

**Deplus, le PF sense devra rediriger tous les paquets VPN entrant vers votre serveur VPN**

**III Mise en route du routage sur le serveur et redirection sur le PareFeu**

**Preparation**

****

****

**Routage du serveur VPN**

**Dans le fichier /etc/sysctl.conf , rendez le serveur VPN routeur :**

**Net.ipv4.forward = 1**

**Redemarrez le service systemctl restart procps**

**NAT DYNAMIQUE :**

**Masquez toutes les IPs sortant du tunnel vers le réseau**

**Up iptables –t nat –F**

**Up iptables –t nat –A POSTROUTING –o eth2 –j MASQUERADE**

****

**On change le remote parce que on passe par le PF la connexion avec le port adéquat**

****

**On ajoute le réseau « DATA » sur le serveurVPN**

****

**Test(ping) du client avec une adresse ip en dehors du réseau afin de pingue mon serveur proxy :**

****

1. **Nating et Filtrage**

**Votre PF n’est pas serveur VPN, donc redirigez les requêtes VPN vers les serveus VPN avec les règles de NAT de PFSENSE :**

**Le client envoie une requête au PF avec la redirection de port :1194, donc comme le PF ne connait pas d’où provient cette requête il va rediriger la requete vers le serveur avec le port 1194**

**CORRECTION DES PROBLEMES :**

**Recreer le tunnel pour le client**