Openvpn User Acess Manager

OUAM Project

Specifications :

L'outil doit permettre de gérer divers aspects quant à l'authentification des utilisateurs du service VPN. Le dit service sera du type OpenVPN, une implémentation libre d'un protocole permettant la connexion entre deux points de manière sécurisée.

Voici les différents critères de succès :

- Connexion : Chaque **utilisateur doit avoir ses propres credentials**. Un credential est assimilé à une méthode d'authentification. Il peut être représenté par un **certificat unique** et des **login/password**, ou un **ensemble de certificats**.
- Authentification : Les credentials doivent avoir une **durée de vie** raisonnable (parametrable) afin de limiter au maximum les failles du système en cas de perte de ceux-ci. Dans le même axe, l'outil doit pouvoir **générer les nouveaux crédentials** et les **fournir aux différents clients** en temps raisonnable afin de maintenir le service.
- Profil : Chaque utilisateur doit être représenté par un **profil** contenant des données permettant de l'identifier et l'ensemble de ses **préférences**.
- Sécurité :
    - le système doit être maître des utilisations du service VPN, chaque **connexion doit être vérifiée** en accord avec la situation courante du système
    - il doit garder une **trace de chaque connexion** via un enregistrement systématique
    - garder des **traces de l'ensemble des opérations** qu'il réalise (génération de credentials, expiration de credentials, envoie de credentials à un utilisateur)
    - intercepter les **tentatives d'intrusions** et générer des **alertes**
- Assistance : le système doit contenir un module d'**assistance à l'utilisateur**. Ce module peut être sous la forme :
    - d'une aide (**manuel**)

    - d'un **détecteur d'erreur de configuration** (certificat client) qui peut envoyer des notifications à l'utilisateur contenant des astuces pour régler son problème
- API : L'application doit posséder un point d'entrée logiciel (API) qui garantit l'aspect modulaire en autorisant d'autres applicatifs à faire des requêtes sur le statut du système.

Contraintes

Le système doit être modulaire au possible car il doit pouvoir s'interfacer avec d'autres applications au sein d'un SI plus vaste.

Le stockage des données doit être fait dans une base de type SQL mise à disposition.

Le programme doit être constitué d'un executable simple et doit être écrit dans un langage évolué facilitant les utilisations de commandes shell.

Le programme doit pouvoir tourner dans un environnement restreint type CHROOT avec des droits limités.

Premier incrément :

1. Moteur du programme (programme de base + définition de l'aspect modulaire)
2. Définition des profils et gestionnaire de profils
3. Moteur de génération de credentials
4. Lien Profils ⇔ génération de credentials