Installation et configuration Certificat SSL

Sommaire

1. Cahier des charges – Expression des besoins

1.1 Descriptif de l'existant

Actuellement, le serveur Web interne (**Apache2**) écoute uniquement sur le port 80 (**HTTP**). Le protocole HTTP transmettant les données en clair, toute interception des trames réseau (**via attaque Man-in-the-Middle ou ARP Spoofing sur le LAN**) permet de lire les informations sensibles, telles que les identifiants de connexion aux applications hébergées.

1.2 Besoin(s)

- Chiffrer les communications de bout en bout entre les navigateurs clients et le serveur Web en déployant le protocole HTTPS (**port 443**).
- Générer un bi-clé de chiffrement asymétrique (**clé publique / clé privée**) basé sur l'algorithme RSA (**2048 bits minimum**).
- Forcer la redirection automatique de tout le trafic HTTP non sécurisé vers la version HTTPS.

1.3 Contrainte(s)

- **Infrastructure réseau :** Le serveur n'étant pas exposé sur Internet (**pas de nom de domaine public FQDN, accès par IP locale**), il est impossible d'utiliser une Autorité de Certification publique comme Let's Encrypt.
- **Sécurité :** La clé privée générée ne devra en aucun cas être lisible par un autre utilisateur que ***root***.
- **Continuité de service :** Le redémarrage du service Web pour appliquer la configuration doit se faire sans interruption majeure (**utilisation de *reload* plutôt que *restart***).

2. Ressources

2.1 Ressources mises à disposition

- **Serveur :** Une machine virtuelle Debian 13 hébergeant le service apache2.
- **Accès :** Connexion à distance via SSH depuis un poste client avec privilèges d'administration (***sudo***).

2.2 Ressources nécessaires

- **Paquets systèmes :** L'utilitaire ***openssl*** (pour la génération des certificats X.509) et ***ssl-cert***.
- **Modules Apache :** ***mod_ssl*** (pour le support cryptographique) et ***mod_rewrite*** ou ***mod_alias*** (pour la redirection).

2.3 Gestion des ressources

La génération initiale des clés RSA consomme un cycle CPU très court. En production, le protocole TLS (Transport Layer Security) induit un léger *overhead* (surcoût de calcul lors du "Handshake" SSL entre le client et le serveur), mais négligeable pour un usage interne. L'espace de stockage requis pour les clés est de quelques kilo-octets.

3. Analyse

3.1 Descriptifs des solutions

- **Certificat Let's Encrypt (CA Publique) :** Délivré par un tiers de confiance. Le navigateur valide immédiatement le certificat sans alerter l'utilisateur. Exige un nom de domaine valide et le port 80 ouvert sur Internet pour le challenge de vérification.
- **Certificat Auto-signé (OpenSSL) :** Généré localement. Le serveur est sa propre autorité de certification. Chiffre les données de manière identique, mais déclenche une alerte de sécurité ("Connexion non privée") dans les navigateurs, car l'émetteur n'est pas reconnu mondialement.

3.2 Comparaison des solutions

Critères technologiques	**CA Publique (ex: Let's Encrypt)**	**Certificat Auto-signé (Choisi)**
Méthode de validation	DNS Challenge ou HTTP-01	**Aucune (Génération locale autonome)**
Niveau de Chiffrement	TLS 1.2 / TLS 1.3	**TLS 1.2 / TLS 1.3 (Identique)**
Périmètre d'usage	Serveurs exposés sur le Web	**Réseaux locaux (LAN), Intranets, Tests**
Durée de validité	90 jours (renouvellement auto)	**Définie par l'admin (ex: 365 jours)**

3.3 Choix d'une solution – Argumentation

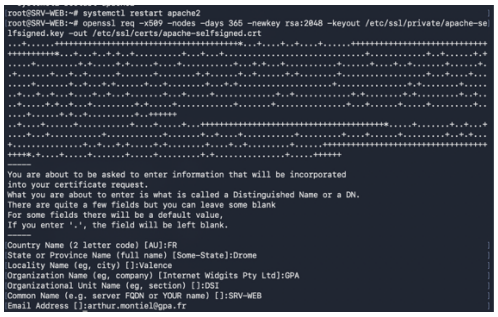
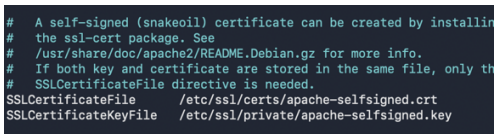
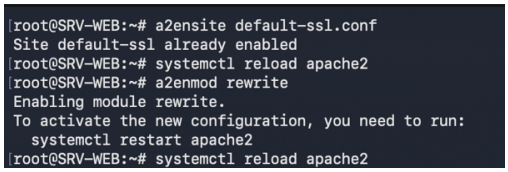
Le **Certificat Auto-signé** est l'unique solution viable et la plus sécurisée pour un environnement de test ou un intranet complètement isolé. Il répond techniquement au besoin de chiffrement sans exposer inutilement le serveur sur Internet.

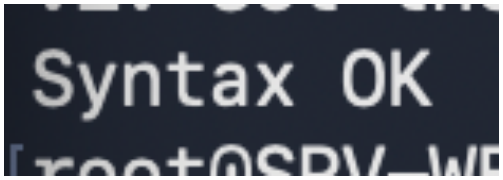
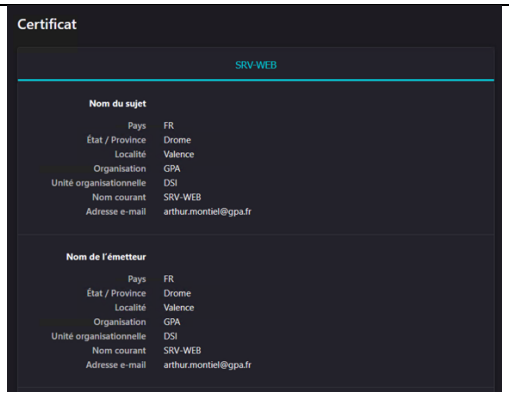
3.4 Étude de l'impact sur le SI existant

- **Réseau :** Nécessité de s'assurer que le pare-feu local (UFW ou iptables) autorise le trafic entrant sur le port TCP 443.
- **Système de fichiers :** Mise en place d'une structure stricte ***/etc/ssl/certs/*** (pour la clé publique/certificat, droits 644) et ***/etc/ssl/private/*** (pour la clé privée, droits 600 exclusifs à root).

4. Mise en place

4.1 Réalisation en suivant le phasage énoncé précédemment

Étape	Description	Images
1	**Activation SSL :** Chargement du module cryptographique d'Apache en mémoire pour lui permettre de comprendre le protocole HTTPS. ***sudo a2enmod ssl***	
2	**Génération de la paire de clés :** Utilisation d'OpenSSL pour créer un certificat X.509 (-x509) sans mot de passe (-nodes) valide 1 an (-days 365) avec une clé RSA 2048 bits. *sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt*	
3	**Configuration du VirtualHost :** Édition du fichier de configuration sécurisé pour lier Apache aux nouvelles clés créées à l'étape 2. *sudo nano /etc/apache2/sites-available/default-ssl.conf* *(Vérifier les lignes : SSLCertificateFile et SSLCertificateKeyFile)* Remplacer ces chemins par ceux de la clé et du certificat de l'étape 2. Ça doit donner exactement ça : SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key	
4	**Activation et Redirection :** Activation du site HTTPS. Configuration d'une redirection permanente (Code 301) pour forcer les utilisateurs tapant HTTP vers HTTPS. sudo a2ensite default-ssl.conf sudo a2enmod rewrite	

5	**Application des règles :** Test de la syntaxe de configuration pour éviter un crash, puis rechargement à chaud d'Apache sans couper les connexions en cours. *sudo apache2ctl configtest* *sudo systemctl reload apache2*	
6	**Test de conformité final :** Accès au serveur via un navigateur Web pour constater la présence du protocole HTTPS et de l'avertissement lié à l'auto-signature.	

4.2 Rapports de tests

Test de conformité	Action effectuée	Résultat attendu	Résultat obtenu
Vérification de l'écoute	Lancer la commande ss -tlpn | grep 443	Le service apache2 écoute bien sur le port 443	**OK**
Droits de la clé privée	ls -l /etc/ssl/private/	Droits restreints (ex: -rw------- root)	**OK**
Test de Redirection	Saisir http://IP_SERVEUR dans le navigateur	Redirection immédiate vers https://IP_SERVEUR	**OK**

5. Bilan

5.1 Conclusion

Le déploiement du protocole TLS/SSL sur le serveur Web est pleinement fonctionnel. La surface d'attaque du serveur a été drastiquement réduite, car les identifiants et les requêtes transitent désormais dans un tunnel chiffré. Le cahier des charges est respecté : la mise en œuvre a été réalisée avec des outils natifs (OpenSSL) sans frais de licence, et la redirection force les utilisateurs à utiliser une connexion sécurisée.

5.2 Auto-critique / Auto-évaluation

- **Points forts :** L'utilisation de la commande apache2ctl configtest avant chaque redémarrage garantit de ne jamais faire tomber le serveur en production à cause d'une faute de frappe dans les fichiers de configuration. Le paramètre -nodeslors de la création de la clé évite à Apache de demander un mot de passe à chaque redémarrage physique de la machine.
- **Points de vigilance :** L'alerte de sécurité "Votre connexion n'est pas privée" sur les navigateurs clients (Safari/Chrome/Edge) est normale mais doit être communiquée aux utilisateurs locaux.
- **Évolutions possibles :** Pour aller plus loin dans la sécurisation, il serait pertinent d'implémenter l'en-tête de sécurité **HSTS** (HTTP Strict Transport Security) dans la configuration d'Apache, afin d'interdire définitivement aux navigateurs de tenter une connexion en clair sur ce serveur.

5.3 Compétence(s) SISR mobilisée(s)

- **Protéger les données à caractère personnel :** Mise en place d'un tunnel chiffré garantissant la confidentialité des échanges sur le réseau.
- **Gérer le patrimoine informatique :** Sécurisation d'un composant de l'infrastructure logicielle (Apache2).