

Activité 3 SISR – Déploiement d'application avec OCSInventory

Document 2 : configuration d'un serveur HTTPS

Le serveur HTTPS sera sur le même serveur physique que le serveur OCS.

Cette fiche ne constitue en aucun cas un TP/cours sur les protocoles HTTPS et SSL. Voir pour plus de précision le TP correspondant de SI5.

Les clés privée, publique et le certificat vont être créés avec l'utilitaire "OpenSSL" ; en conséquence, ce dernier doit être installé sur le serveur OCS.

Première étape : obtenir un certificat pour le serveur WEB

Il est possible :

- d'acheter un certificat SSL 128 bits à un prix raisonnable aux alentours de 29.00\$US/an. Voir par exemple à cette adresse <https://securityserver.org/>.
- d'obtenir un certificat d'essai utilisable en environnement de test : <http://www.verisign.fr/ssl/buy-ssl-certificates/free-trial/>
- de créer son certificat : c'est ce que nous nous proposons de faire... mais toute solution opérationnelle sera acceptée.

La documentation officielle d'OCS propose un script de génération de certificat à utiliser avec Apache reproduit ci-dessous qui fonctionne parfaitement

<http://wiki.ocsinventory-ng.org/index.php/Documentation:Teledeploy/fr> :

#vi apache_generate_cert.sh

```
#!/bin/sh
# En premier, generer le certificat requis
# Generer une clé RSA de 1024 bits, enregistrer la cle privée dans un
# fichier PEM de mot-de-passe non protege server.key, en utilisant
# le fichier de configuration par default d'openssl
#
echo
echo Generation de la cle privée du serveur Apache...
echo
openssl genrsa -out server.key 1024
#
# Maintenant, signez le certificat du serveur Apache avec
# la cle du serveur Apache
# Signez avec le certificat PEM server.crt,
# en utilisant le fichier PEM server.key pour cle privée du server,
# en utilisant le fichier de configuration par default d'openssl.
# Le certificat produit sera valide durant 1825 jours (soit 5 ans).
#
echo
echo Generation des certificats auto-signes du serveur Apache ...
echo
openssl req -outform PEM -new -key server.key -x509 -days 1825 -out server.crt
```

Ce script génère une clé privée RSA dans le fichier "server.key" et un certificat X.509 auto-signé dans le fichier "server.crt".

Il faut maintenant :

- mettre les droits d'exécution au script : `chmod u+x apache_generate_cert.sh`,
- lancer le script en utilisant cette commande : `sh apache_generate_cert.sh`

Celui-ci générera la clé privée, et vous demandera les propriétés du certificat :

- le code pays,
- le nom de la province ou de l'état,
- le nom de la ville,
- le nom de votre organisation,
- le nom de votre section d'organisation,
- **le nom commun (le nom DNS ou l'adresse ip de votre serveur – très important – le "CNAME" du certificat doit correspondre au nom du serveur OCS ou l'@IP telle qu'il (elle) figure dans les fichiers de configuration – ça sera une adresse IP dans notre cas (voir remarque ci-dessous) ,**
- une adresse mél optionnelle.



À priori et pour une raison que je ne m'explique pas, l'utilisation d'un FQDN dans le certificat empêche le bon fonctionnement du télédéploiement (même si ce FQDN figure dans les fichiers de configuration).

Deuxième étape : configurer Apache2 avec mod_ssl

Il est nécessaire de :

- charger le module ssl avec l'utilitaire debian `a2enmod` en lançant la commande `a2enmod ssl` dont le retour doit être du style : "Enabling module ssl. Run '/etc/init.d/apache2 restart' to activate new configuration!" ; ce que l'on pourra faire plus tard car il y a d'autres modifications à apporter.
L'activation du module a notamment pour effet d'activer le port d'écoute 443.
- copier le fichier du certificat du serveur "server.crt" et le fichier de la clé privée "server.key" dans les répertoires appropriés :
#cp /root/server.crt /etc/ssl/certs/
- **#cp /root/server.key /etc/ssl/private/**
- mettre à jour les fichiers de configuration d'Apache2 pour utiliser ces fichiers (/etc/apache2/sites-available/default-ssl ou tout autre fichier de configuration personnel) – Attention, par mesure de précaution faites une copie de ces fichiers avant de les modifier.
Les 2 directives à modifier sont SSLCertificateFile et SSLCertificateKeyFile :
SSLCertificateFile /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
- activer (si ce n'est déjà fait) la nouvelle configuration : `a2ensite default-ssl` dont le retour doit être "Enabling site default-ssl. Run '/etc/init.d/apache2 reload' to activate new configuration! " ;
- redémarrez Apache2 : **`/etc/init.d/apache2 restart`**.

L'URL : `https://@IP|nom_dns|nom_serveur` devrait maintenant vous renvoyer une page d'alerte de sécurité ; ceci est normal car notre certificat n'a pas été signé par une autorité de certification connue. Il suffit d'accepter l'exception. Mais il est bien évident que ce n'est pas une solution acceptable en environnement de production ...

Troisième étape : copier le certificat sur chaque client

L'agent doit avoir un certificat pour valider l'authentification au serveur de déploiement. Il s'agit du fichier `server.crt`. Ce certificat doit être enregistré dans un fichier "**cacert.pem**" dans le répertoire de l'agent OCS Inventory NG sous Windows et dans `"/var/lib/ocsinventory-agent/http/___@IPserveurOCS_ocsinventory/"` sous Linux.

Il vaut mieux créer une copie de ce certificat sur le serveur en le renommant (`mv server.crt cacert.pem`) avant de le copier sur chaque client Windows car si sur ce dernier la visibilité des extensions n'est pas activée le certificat aura pour nom "cacert.pem.crt" et l'authentification ne pourra pas se faire.