2024

Configuration TLS de Keycloak

CAS D'USAGES PRATIQUES

MIKAËL POLLET

Table des matières

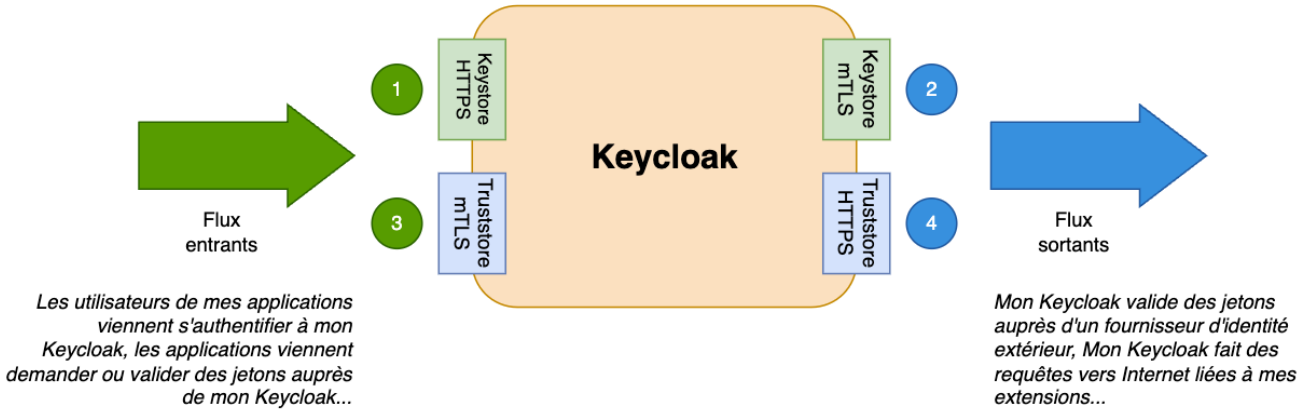
Notions importantes concernant les Keystores / Truststores Keycloak

La solution Keycloak utilise des mécanismes de Keystores et Truststores Java pour assurer le chiffrement TLS des flux entrant et sortant, assurer l'authentification client par certificats, et assurer une communication de confiance avec les serveurs distants dont elle accepte l'autorité de certification.
Une séparation métier claire permet de contrôler proprement les éléments techniques utilisés (certificats) pour le contexte métier adéquate.

Cette séparation se fait de la forme suivante :

ATTENTION : Un changement majeur de l'utilisation des Truststores étant implémenté à partir de la version 24.0.0 de Keycloak sortie le 04/03/2024, il est important de se référer à la partie de la documentation vous concernant. Les deux configurations seront présentées.

Pour Keycloak < 24.0.0



Les Keystores (1 et 2)

1 : Keystore utilisé pour chiffrer la communication TLS avec les clients. Le fameux HTTPS présenté par votre Keycloak. Il est généralement composé d'un certificat commercial (Certigna, Certinomis…) acheté pour présenter un certificat provenant d'une autorité de certification publique de confiance.

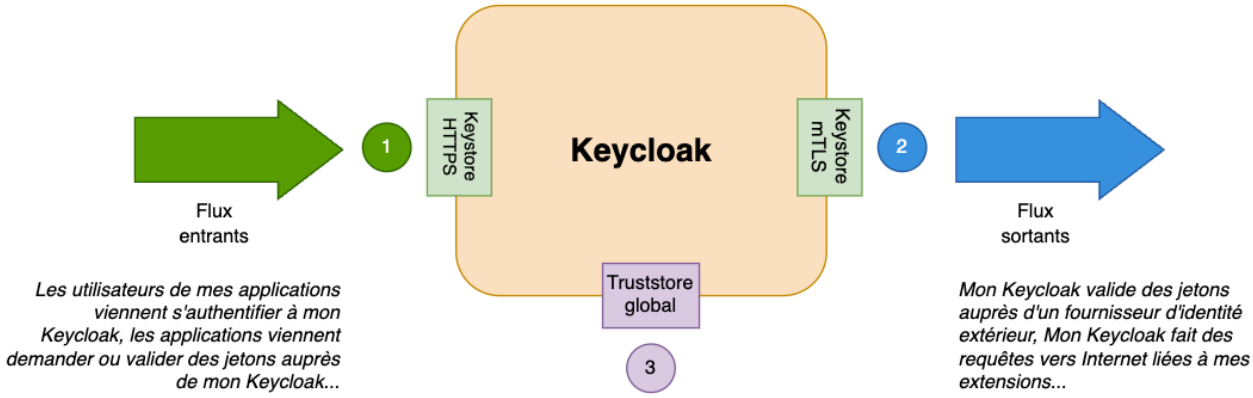
2 : Keystore utilisé pour présenter un certificat d'authentification client à un serveur distant. Il s'agit alors d'une authentification mTLS (Mutual TLS). Les certificats intégrés sont généralement issus d'autorités de certification privées, comme l'IGC Santé de l'ANS pour assurer une authentification cliente maîtrisée par le fournisseur du service consommé.

Les Truststores (3 et 4)

3 : Truststore utilisé pour identifier les certificats ou les autorités des certificats pour lesquels j'accepte l'authentification de mes clients. Il s'agit alors d'authentification mTLS que j'impose à mes utilisateurs/applications. Généralement, il s'agit d'une autorité de certification privée, que je maîtrise et avec laquelle je fournirai des certificats à mes utilisateurs/applications.
Ce Truststore ne doit contenir que des autorités de certification limitées au périmètre pertinent pour l'accès à mes applications, et généralement pas d'autorités publiques, afin d'éviter qu'un utilisateur puisse utiliser un certificat acheté auprès d'une autorité publique et puisse s'authentifier à mes applications.

4 : Truststore utilisé pour identifier les autorités de certification auprès desquelles j'accepte de me connecter. Quand mon Keycloak vérifiera un jeton auprès d'un fournisseur d'identité extérieur par exemple, je dois pouvoir accepter le certificat TLS qui me sera présenté par le serveur distant.
Généralement, ce Truststore contient l'ensemble des autorités publiques de confiance, ainsi que les autorités privées que j'estime de confiance.

Pour Keycloak >= 24.0.0



Les Keystores (1 et 2)

1 : Keystore utilisé pour chiffrer la communication TLS avec les clients. Le fameux HTTPS présenté par votre Keycloak. Il est généralement composé d'un certificat commercial (Certigna, Certinomis…) acheté pour présenter un certificat provenant d'une autorité de certification publique de confiance.

2 : Keystore utilisé pour présenter un certificat d'authentification client à un serveur distant. Il s'agit alors d'une authentification mTLS (Mutual TLS). Les certificats intégrés sont généralement issus d'autorités de certification privées, comme l'IGC Santé de l'ANS pour assurer une authentification cliente maîtrisée par le fournisseur du service consommé.

Truststore global (3)

3 : Désormais, Keycloak utilise un Truststore commun pour les flux entrants et sortants. Il contient donc les certificats et autorités de certification pour lesquels je choisi de faire confiance.
Par défaut, Keycloak accepte désormais les autorités publiques présentes dans le Truststore par défaut de Java (CACERT), et le Truststore commun de Keycloak permet d'inclure des certificats/autorités de certification privés.

ATTENTION : Keycloak acceptant désormais par défaut les autorités de certification publiques, si votre Keycloak est utilisé pour faire de l'authentification cliente mTLS, il est important de verrouiller la vérification des certificats dans la configuration du flux d'authentification pour correspondre à l'autorité de certification souhaitée. Si ce verrouillage n'est pas réalisé, n'importe quel certificat issu d'une autorité publique pourrait être utilisé pour réaliser une authentification cliente valide. Ces contrôles n'étaient alors pas nécessaires vu que seuls les certificats ou les autorités de confiance, implémentées dans le Truststore mTLS permettaient une authentification valide. Ce n'est plus le cas désormais.

Création et configuration des Keystores / Truststores dans Keycloak

Dans le monde Java, les Keystores et les Truststores sont techniquement identiques. Il s'agit d'un conteneur appelé « Porte-clés » ou « Keystore » en anglais, qui contiendra les éléments nécessaires au rôle pour lequel il est attribué.

La différence majeure sera donc son contenu.

Les Keystores contiennent un ou plusieurs certificats (clé publique) ainsi que leur clé privée, accompagnés de leur autorité de certification.

Les Truststores contiennent uniquement les certificats (clé publique) et autorités de certification pour lesquels je fais confiance.
Aucune clé privée ne doit être intégrée dans un Truststore. En effet, cet élément pourrait être affiché à un client lors de la communication avec votre Keycloak.

Les éléments créés ici seront systématiquement créés au format PKCS12 qui est le format cible des Keystores Java. Les formats JKS et JCEKS ne seront pas abordés dans la suite de ce document.

Prérequis

Installation du JDK Java

La création d'un Keystore nécessite l'installation de Java (JDK) sur le poste ou le serveur qui initie la création, afin d'utiliser l'outil Keytool.

Création du répertoire de destination

Sur le serveur Keycloak :

Serveur Linux :

sudo mkdir /opt/keycloak/conf/tls/

Serveur Windows :

- Se positionner dans le répertoire de Keycloak > conf
- Créer un répertoire « tls »

Keystore HTTPS (1)

Création du Keystore

Préparer un mot de passe : (ex : afCqwJiABQP-P2OqVGhqWBHHo_9PWtGI)

Sur Linux :

sudo keytool -genkeypair -storetype pkcs12 -alias keycloak-keystore -keyalg RSA -validity 72000 -keysize 4096 -keystore /opt/keycloak/conf/tls/keycloak-keystore.jks -dname "CN=keycloak-keystore,OU=mon_service,OU=mon_entreprise,C=FR"

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -genkeypair -storetype pkcs12 -alias keycloak-keystore -keyalg RSA -validity 72000 -keysize 4096 -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-keystore.jks -dname "CN=keycloak-keystore,OU=mon_service,OU=mon_entreprise,C=FR"

Import du certificat TLS propriétaire ou commercial au Keystore

Sur Linux :

sudo keytool -importkeystore -srckeystore <CHEMIN_CERTIFICAT>/mon_certificat.pfx -srcstoretype pkcs12 -destkeystore /opt/keycloak/conf/tls/keycloak-keystore.jks -deststoretype pkcs12

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -importkeystore -srckeystore <CHEMIN_CERTIFICAT>/mon_certificat.pfx -srcstoretype pkcs12 -destkeystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-keystore.jks -deststoretype pkcs12

Configuration de Keycloak

Éditer le fichier de configuration de Keycloak et ajouter ou modifier les lignes suivantes :

Sur Linux :

Ficher de configuration : /opt/keycloak/conf/keycloak.conf

https-key-store-file=/opt/keycloak/conf/tls/keycloak-keystore.jks
https-key-store-password=afCqwJiABQP-P2OqVGhqWBHHo_9PWtGI
https-key-store-type=pkcs12

Sur Windows :

Ficher de configuration : <CHEMIN_KEYCLOAK>\conf\keycloak.conf

https-key-store-file=<CHEMIN_KEYCLOAK>\conf\tls\keycloak-keystore.jks
https-key-store-password=afCqwJiABQP-P2OqVGhqWBHHo_9PWtGI
https-key-store-type=pkcs12

Faites un rebuild de la configuration Keycloak et redémarrez le service

Keystore mTLS (2)

Création du Keystore

Préparer un mot de passe pour le Keystore : (ex : aqkl_j3hVTLafWuFS-w34lnBObxCdB5)
Préparer un mot de passe pour les clés privées des certificats d'authentification client : (ex : 3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS)

Sur Linux :

sudo keytool -genkeypair -storetype pkcs12 -alias keycloak-mtls-keystore -keyalg RSA -validity 72000 -keysize 4096 -keystore /opt/keycloak/conf/tls/keycloak-mtls-keystore.jks -dname "CN=keycloak-mtls-keystore,OU=mon_service,OU=mon_entreprise,C=FR"

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -genkeypair -storetype pkcs12 -alias keycloak-mtls-keystore -keyalg RSA -validity 72000 -keysize 4096 -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-keystore.jks -dname "CN=keycloak-mtls-keystore,OU=mon_service,OU=mon_entreprise,C=FR"

Import d’un certificat d’authentification client au Keystore

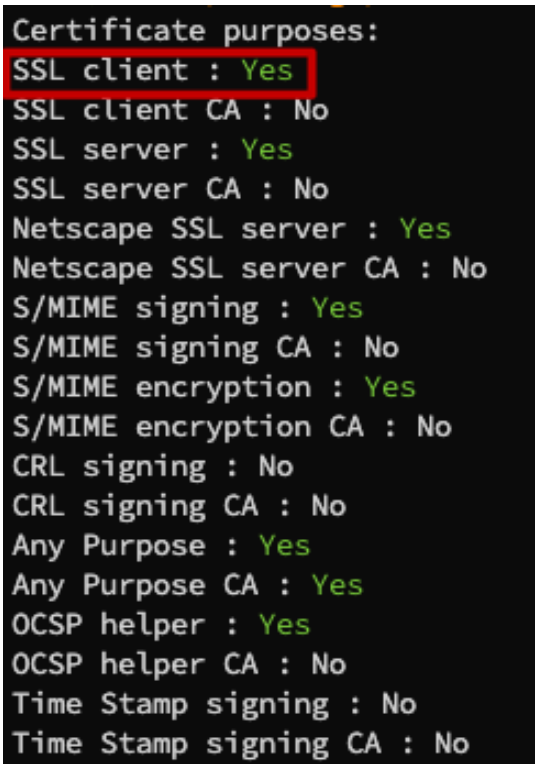
Vérification du/des rôle(s) du certificat

IMPORTANT : Seul un certificat qui possède un rôle d’authentification client pourra être utilisé pour l’authentification mTLS auprès d’un serveur distant.

Il est possible de vérifier si le certificat concerné possède bien ce rôle :

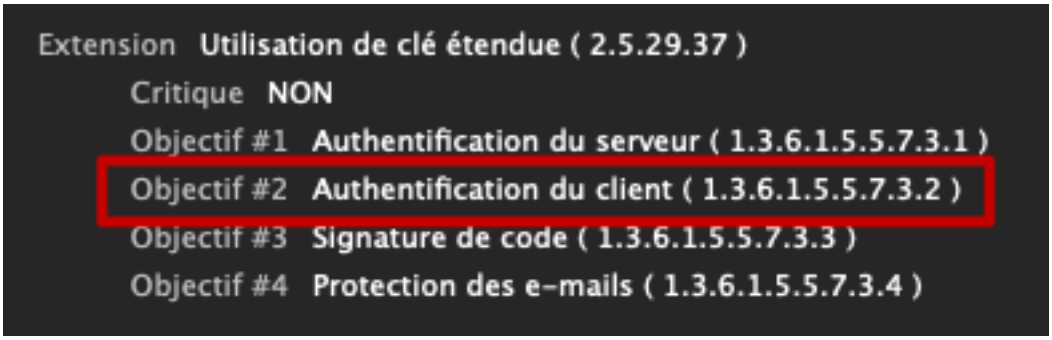
- Grâce à la commande OpenSSL : (Le certificat doit être extrait au format PEM)

openssl x509 -purpose -in mon_certificat.crt

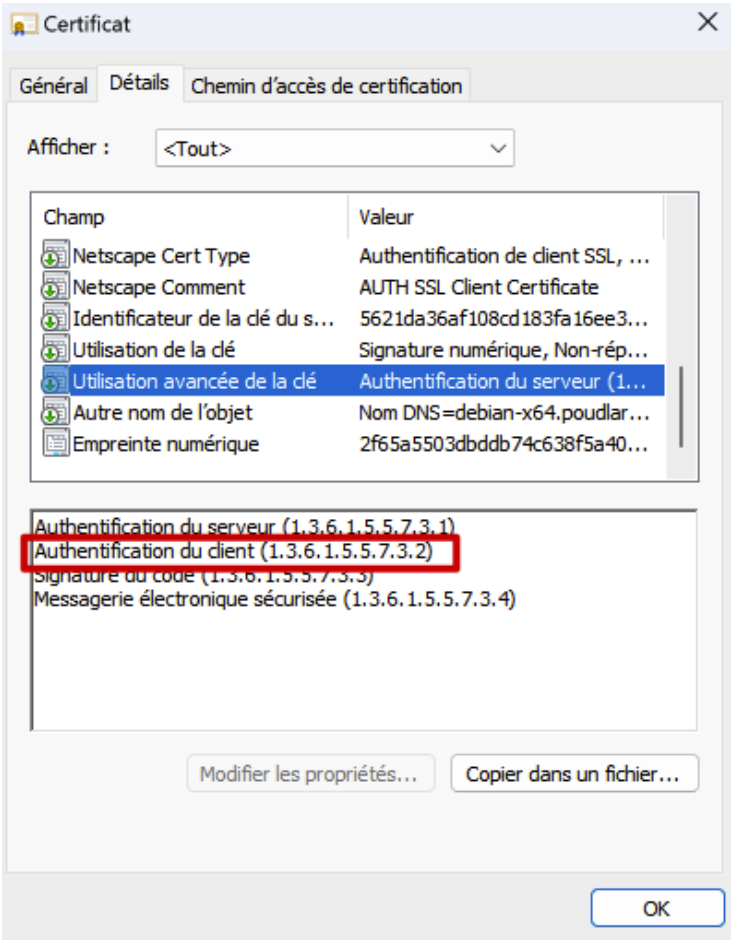


- À l’affichage du certificat sur le système d’exploitation :

Exemple sur MacOS



Exemple sur Windows



Modification du mot de passe de la clé privée du certificat qui sera importé

Il est important de sécuriser l'accès à la clé privée des certificats qui seront utilisés pour l'authentification client vers un serveur distant par un mot de passe.

La solution Keycloak permet de prendre en charge la sécurisation de ces clés privées grâce à un mot de passe différent de celui d'accès au Keystore. Il est cependant nécessaire que ce mot de passe soit identique pour tous les certificats importés.

Il est donc nécessaire de changer le mot de passe initial de la clé privée d'un certificat avant de l'importer dans le Keystore.

Exemple de changement de mot de passe depuis un certificat PFX

Ce changement doit être réalisé grâce à l'outil OpenSSL

openssl pkcs12 -in <CHEMIN_CERTIFICAT>/mon_certificat.pfx -out <CHEMIN_CERTIFICAT>/cle_privee.pem -nodes

- ⇨ Indiquer le mot de passe original de la clé privée

openssl pkcs12 -export -out <CHEMIN_CERTIFICAT>/mon_certificat_protege.pfx -in <CHEMIN_CERTIFICAT>/cle_privee.pem

- ⇨ Indiquer le nouveau mot de passe prévu à cet effet précédemment 3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS

IMPORTANT : N'oubliez pas de supprimer <CHEMIN_CERTIFICAT>/cle_privee.pem une fois l'opération terminée.

Exemple de changement de mot de passe depuis un Keystore Java

Ce changement doit être réalisé grâce à l'outil Keytool

Sur Linux :

sudo keytool -storepasswd -keystore <CHEMIN_CERTIFICAT>/mon_keystore.jks

- ⇨ Indiquer le nouveau mot de passe prévu à cet effet précédemment 3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -storepasswd -keystore <CHEMIN_CERTIFICAT>/mon_keystore.jks

- ⇨ Indiquer le nouveau mot de passe prévu à cet effet précédemment 3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS

Import sur Linux :

sudo keytool -importkeystore -srckeystore <CHEMIN_CERTIFICAT>/mon_certificat.pfx -srcstoretype pkcs12 -destkeystore /opt/keycloak/conf/tls/keycloak-mtls-keystore.jks -deststoretype pkcs12

Import sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -importkeystore -srckeystore <CHEMIN_CERTIFICAT>/mon_certificat.pfx -srcstoretype pkcs12 -destkeystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-keystore.jks -deststoretype pkcs12

Configuration de Keycloak

Éditer le fichier de configuration de Keycloak et ajouter ou modifier les lignes suivantes :

Sur Linux :

Ficher de configuration : /opt/keycloak/conf/keycloak.conf

spi-connections-http-client-default-client-keystore=/opt/keycloak/conf/tls/keycloak-mtls-keystore.jks
spi-connections-http-client-default-client-keystore-password=aqkl_j3hVTLafWuFS-w34lnBObxCdB5
spi-connections-http-client-default-client-key-password=3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS

Sur Windows :

Ficher de configuration : <CHEMIN_KEYCLOAK>\conf\keycloak.conf

spi-connections-http-client-default-client-keystore=<CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-keystore.jks
spi-connections-http-client-default-client-keystore-password=aqkl_j3hVTLafWuFS-w34lnBObxCdB5
spi-connections-http-client-default-client-key-password=3YTa-uGpd2GlG-ayY5bO_vJwyV9wxWfS

Faites un rebuild de la configuration Keycloak et redémarrez le service

Truststores

Il est très important de **ne jamais importer de certificat contenant une clé privée**. Seul le certificat et sa clé publique doivent être importés.

Pour vous assurer de ne pas faire l'erreur, l'import depuis un certificat au format PEM est une bonne solution. Au format PEM, le certificat est décorrélé de sa clé privée. Il est en général stocké avec l'extension .crt, mais seul le contenu est important.

Vous pouvez reconnaître un certificat au format PEM en l'éditant avec un éditeur de texte.

Il doit commencer par la balise ***-----BEGIN CERTIFICATE-----*** et se terminer par la balise ***----- END CERTIFICATE-----***

Ex :

Seuls ces éléments doivent être présents.

Keycloak < 24.0.0

Truststore mTLS (3)

Création du Truststore

Préparer un mot de passe : (ex : ZNcF1_HU_H0-boHsYmkrUaSGJcaKNNmG)

Sur Linux :

sudo keytool -genkeypair -storetype pkcs12 -alias keycloak-mtls-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore /opt/keycloak/conf/tls/keycloak-mtls-truststore.jks -dname "CN=keycloak-mtls-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -genkeypair -storetype pkcs12 -alias keycloak-mtls-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-truststore.jks -dname "CN=keycloak-mtls-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Import d’un certificat de confiance dans le Truststore

Sur Linux :

sudo keytool -import -alias certificat_client -file <CHEMIN_CERTIFICAT>/certificat_client.crt -keystore /opt/keycloak/conf/tls/keycloak-mtls-truststore.jks

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -import -alias certificat_client -file <CHEMIN_CERTIFICAT>/certificat_client.crt -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-truststore.jks

Configuration de Keycloak

Éditer le fichier de configuration de Keycloak et ajouter ou modifier les lignes suivantes :

Sur Linux :

Ficher de configuration : /opt/keycloak/conf/keycloak.conf

https-trust-store-file=/opt/keycloak/conf/tls/keycloak-mtls-truststore.jks
https-trust-store-password=ZNcF1_HU_H0-boHsYmkrUaSGJcaKNNmG
https-trust-store-type=pkcs12

Sur Windows :

Ficher de configuration : <CHEMIN_KEYCLOAK>\conf\keycloak.conf

https-trust-store-file=<CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-truststore.jks
https-trust-store-password=ZNcF1_HU_H0-boHsYmkrUaSGJcaKNNmG
https-trust-store-type=pkcs12

Faites un rebuild de la configuration Keycloak et redémarrez le service

Truststore HTTPS (4)

Création du Truststore

Préparer un mot de passe : (ex : 7HLlulOA3_2BnZ2AI03Z30-ytI-ldHGr)

Sur Linux :

sudo keytool -genkeypair -storetype pkcs12 -alias keycloak-https-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore /opt/keycloak/conf/tls/keycloak-https-truststore.jks -dname "CN=keycloak-https-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -genkeypair -storetype pkcs12 -alias keycloak-https-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-https-truststore.jks -dname "CN=keycloak-https-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Import d'un certificat de confiance dans le Truststore

Sur Linux :

sudo keytool -import -alias autorite_de_confiance -file <CHEMIN_CERTIFICAT>/autorite_de_confiance.crt -keystore /opt/keycloak/conf/tls/keycloak-https-truststore.jks

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -import -alias autorite_de_confiance -file <CHEMIN_CERTIFICAT>/autorite_de_confiance.crt -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-truststore.jks

Configuration de Keycloak

Éditer le fichier de configuration de Keycloak et ajouter ou modifier les lignes suivantes :

Sur Linux :

Ficher de configuration : /opt/keycloak/conf/keycloak.conf

spi-truststore-file-file=/opt/keycloak/conf/tls/keycloak-https-truststore.jks
spi-truststore-file-password=7HLlulOA3_2BnZ2AI03Z30-ytI-ldHGr
spi-truststore-file-hostname-verification-policy=STRICT
spi-truststore-file-type=pkcs12

Sur Windows :

Ficher de configuration : <CHEMIN_KEYCLOAK>\conf\keycloak.conf

spi-truststore-file-file=<CHEMIN_KEYCLOAK>\conf\tls\keycloak-https-truststore.jks
spi-truststore-file-password=7HLlulOA3_2BnZ2AI03Z30-ytI-ldHGr
spi-truststore-file-hostname-verification-policy=STRICT
spi-truststore-file-type=pkcs12

Niveau de vérification du certificat présenté

Il est possible d'élever ou de réduire le niveau de vérification du certificat présenté.
Pour ce faire, il est nécessaire de modifier le paramètre spi-truststore-file-hostname-verification-policy.

Liste des niveaux disponibles :

- STRICT : Le DN du certificat présenté doit être exactement celui du hostname présenté
- WILDCARD : Le DN présenté peut être un sous-domaine du hostname présenté (ex : *.mon_domaine.fr)
- ANY : Pas de vérification avec le hostname présenté. Ce paramètre **ne doit pas** être utilisé en production.

Faites un rebuild de la configuration Keycloak et redémarrez le service

Keycloak >= 24.0.0

Truststore Global (3)

Création du Truststore

Depuis Keycloak 24.0.0, les certificats de confiance sont récupérés à la volée par Keycloak depuis un emplacement générique ou un emplacement défini.
A des fins de sécurité, il est plus pertinent de forcer l'utilisation d'un Truststore déclaré et de lui importer les certificats nécessaires.
Pour la récupération des certificats de confiance par Keycloak il est nécessaire de **ne pas mettre de mot de passe** sur le/les Truststore(s) concernés.

Sur Linux :

sudo keytool -genkeypair -storetype pkcs12 -alias keycloak-global-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore /opt/keycloak/conf/tls/keycloak-global-truststore.jks -dname "CN=keycloak-global-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -genkeypair -storetype pkcs12 -alias keycloak-global-truststore -keyalg RSA -validity 72000 -keysize 4096 -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-global-truststore.jks -dname "CN=keycloak-global-truststore,OU=mon_service,OU=mon_entreprise,C=FR"

Import d'un certificat de confiance dans le Truststore

Sur Linux :

sudo keytool -import -alias certificat_de_confiance -file <CHEMIN_CERTIFICAT>/ certificat_de_confiance.crt -keystore /opt/keycloak/conf/tls/keycloak-global-truststore.jks

Sur Windows :

"<CHEMIN_JAVA_HOME>\bin\keytool.exe" -import -alias certificat_de_confiance -file <CHEMIN_CERTIFICAT>/certificat_de_confiance.crt -keystore <CHEMIN_KEYCLOAK>\conf\tls\keycloak-mtls-truststore.jks

Configuration de Keycloak

Éditer le fichier de configuration de Keycloak et ajouter ou modifier les lignes suivantes :

Sur Linux :

Ficher de configuration : /opt/keycloak/conf/keycloak.conf

truststore-paths=/opt/keycloak/conf/tls/keycloak-global-truststore.jks
tls-hostname-verifier=STRICT

Sur Windows :

Ficher de configuration : <CHEMIN_KEYCLOAK>\conf\keycloak.conf

truststore-paths=<CHEMIN_KEYCLOAK>\conf\tls\keycloak-global-truststore.jks
tls-hostname-verifier=STRICT

Si d'autres Truststores doivent être chargés (par exemple avec un Truststore spécifique par application), ceux-ci peuvent être ajoutés au paramètre **truststore-paths** séparés par une virgule.

Ex :

truststore-paths=trusstore1.jks,trustore2.jks……

Niveau de vérification du certificat présenté

Il est possible d'élever ou de réduire le niveau de vérification du certificat présenté.
Pour ce faire, il est nécessaire de modifier le paramètre **tls-hostname-verifier**.

Liste des niveaux disponibles :

- STRICT : Le DN du certificat présenté doit être exactement celui du hostname présenté
- WILDCARD : Le DN présenté peut être un sous-domaine du hostname présenté (ex : *.mon_domaine.fr)
- ANY : Pas de vérification avec le hostname présenté. Ce paramètre **ne doit pas** être utilisé en production.

Autre méthode de gestion des Ttrusstores

Même si cette méthode n'est pas recommandée à des fins de sécurité, il est possible de paramétrer les certificats de confiance autrement.

Keycloak a implémenté un nouveau répertoire par défaut dans /opt/keycloak/conf/ truststores (Linux) et <CHEMIN_KEYCLOAK>\conf\truststores (Windows)

L'ensemble des certificats de confiance doivent alors être placés dans ce répertoire, sans mot de passe, au format Keystore ou PEM.

Keycloak chargera l'ensemble des éléments trouvés dans ce répertoire au lancement.

Dans ce mode de configuration, le paramètre **truststore-paths** n'est plus obligatoire dans le fichier de configuration.
Le paramètre **tls-hostname-verifier** est toujours à configurer.

ATTENTION : Ce mode de configuration n'est pas à privilégier, du fait de la simplicité d'un attaquant d'inclure des certificats dans ce répertoire sans contrôle de l'administrateur de la solution. Le paramètre **truststore-paths** permet par exemple de placer les Truststores dans un répertoire moins accessible et non connu d'un attaquant sans manipulation des fichiers de configuration.

Faites un rebuild de la configuration Keycloak et redémarrez le service

Cas pratique : Implémentation mTLS avec Pro Santé Connect

Le service Pro Santé Connect demande aux fournisseurs de service de communiquer avec les endpoints PSC au moyen d'une authentification mTLS.

L'implémentation de ce type d'authentification technique avec Keycloak se déroulera en deux étapes :

- Démarches administratives pour la récupération du certificat nécessaire : Cette étape sera documentée par les équipes Pro Santé Connect de l'ANS sur le site prévu à cet effet
- Prise en compte du certificat d'authentification client dans l'architecture Keycloak : Les étapes d'implémentation du certificat sont décrites ICI.

L'implémentation technique pourra être répétée pour chaque les certificats d'authentification client demandé auprès de Pro Santé Connect pour les différents services gérés.