Configurer la validation de signature du package IOx

Contenu

Introduction

Ce document décrit en détail comment créer et utiliser des paquets signés sur la plate-forme IOx.

Conditions préalables

Conditions requises

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Connaissances Linux de base
- Comprendre le fonctionnement des certificats

Components Used

Les informations contenues dans ce document sont basées sur les versions de matériel et de logiciel suivantes :

- Périphérique compatible IOx configuré pour IOx :
  Adresse IP configuréeSystème d'exploitation invité (GOS) et Cisco Application Framework (CAF) qui s'exécuteTraduction d'adresses de réseau (NAT) configurée pour l'accès à CAF (port 8443)
- Hôte Linux avec SSL (Secure Sockets Layer) ouvert installé

- Fichiers d'installation du client IOx téléchargeables à l'adresse : https://software.cisco.com/download/release.html?mdfid=286306005&softwareid=286306762

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Informations générales

Depuis la version IOx, la signature du package d'application AC5 est prise en charge. Cette fonctionnalité permet de s'assurer que le package d'application est valide et que celui installé sur le périphérique provient d'une source fiable. Si la validation de signature du package d'application est activée sur une plate-forme, seules les applications signées peuvent être déployées.

Configuration

Ces étapes sont requises pour utiliser la validation de signature de package :

1. Créez une clé et un certificat d'autorité de certification.
2. Générez une ancre d'approbation à utiliser sur IOx.
3. Importez le point d'ancrage d'approbation sur votre périphérique IOx.
4. Créez une clé spécifique à l'application et une demande de signature de certificat (CSR).
5. Signer le certificat spécifique à l'application avec l'utilisation de l'autorité de certification.
6. Empaquetez votre application IOx, signez-la avec le certificat spécifique à l'application.
7. Déployez votre package IOx signé sur un périphérique compatible signature.

Note: Pour cet article, une CA autosignée est utilisée dans un scénario de production. La meilleure option est d'utiliser une autorité de certification officielle ou l'autorité de certification de votre société pour signer.

Note: Les options de l'autorité de certification, des clés et des signatures sont choisies uniquement à des fins de travaux pratiques et peuvent devoir être adaptées à votre environnement.

Étape 1. Créer une clé d'autorité de certification et un certificat

La première étape consiste à créer votre propre CA. Pour ce faire, il suffit de générer une clé pour l'autorité de certification et un certificat pour cette clé :

Afin de générer la clé CA :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl genrsa -out rootca-key.pem 2048
Generating RSA private key, 2048 bit long modulus
.....................................................................................+++
.......+++
e is 65537 (0x10001)
```

Afin de générer le certificat d'autorité de certification :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl req -x509 -new -nodes -key rootca-key.pem -sha256 -
days 4096 -out rootca-cert.pem
You are about to be asked to enter information that is incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name (DN).
There are quite a few fields but you can leave some blank
For some fields there can be a default value,
If you enter '.', the field can be left blank.
-----
Country Name (2 letter code) [XX]:BE
State or Province Name (full name) []:WVL
Locality Name (eg, city) [Default City]:Kortrijk
Organization Name (eg, company) [Default Company Ltd]:Cisco
Organizational Unit Name (eg, section) []:IOT
Common Name (eg, your name or your server's hostname) []:ioxrootca
Email Address []:
```

Les valeurs du certificat CA doivent être ajustées pour correspondre à votre cas d'utilisation.

Étape 2. Générer un ancrage de confiance pour une utilisation sur IOx

Maintenant que vous disposez de la clé et du certificat nécessaires pour votre autorité de certification, vous pouvez créer un bundle d'ancre de confiance à utiliser sur votre périphérique IOx. L'ensemble d'ancrages d'approbation doit contenir la chaîne de signature de l'autorité de certification complète (dans le cas où un certificat intermédiaire est utilisé pour la signature) et un fichier info.txt qui est utilisé pour fournir les métadonnées (formulaire libre).

Tout d'abord, créez le fichier info.txt et mettez-y quelques métadonnées :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ echo "iox app root ca v1">info.txt
```

Si vous avez éventuellement plusieurs certificats CA, pour former votre chaîne de certificats CA, vous devez les assembler en un seul .pem :

```
cat first_cert.pem second_cert.pem > combined_cert.pem
```

> **Note**: Cette étape n'est pas requise pour cet article, car un certificat racine d'autorité de certification unique est utilisé pour le signe direct, ce n'est pas recommandé pour la production et la paire de clés d'autorité de certification racine doit toujours être stockée hors connexion.

La chaîne de certificats CA doit être nommée ca-chain.cert.pem. Préparez donc ce fichier :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ cp rootca-cert.pem ca-chain.cert.pem
```

Enfin, vous pouvez combiner les fichiers ca-chain.cert.pem et info.txt dans un tar compressé :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ tar -czf trustanchorv1.tar.gz ca-chain.cert.pem info.txt
```

Étape 3. Importer un ancrage de confiance sur un périphérique IOx

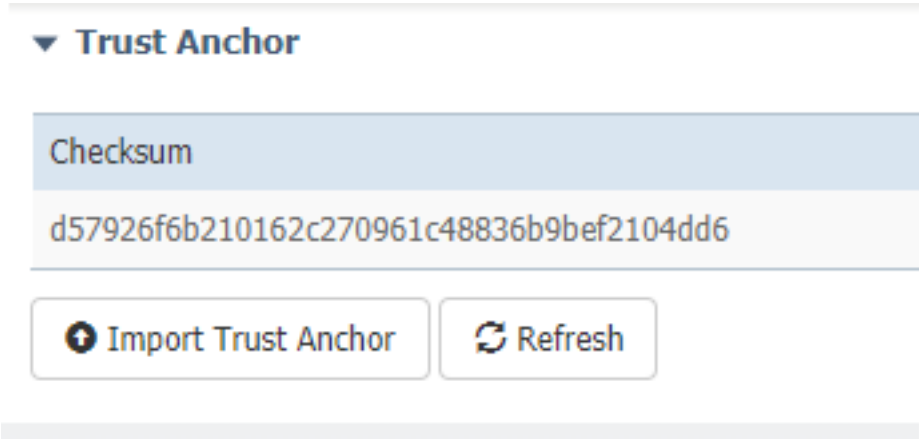
Le fichier trustanchorv1.tar.gz que vous avez créé à l'étape précédente doit être importé sur votre périphérique IOx. Les fichiers du bundle sont utilisés pour vérifier si une application a été signée avec un certificat signé par l'autorité de certification appropriée avant d'autoriser une installation.

L'importation de l'ancre de confiance peut être effectuée via l'indicateur :

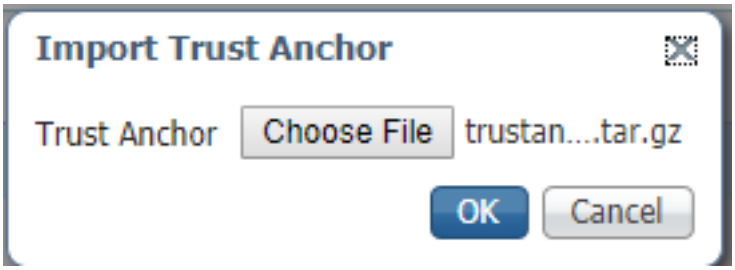
```
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages trustanchor set
trustanchorv1.tar.gz
Currently active profile :  default
Command Name:  plt-sign-pkg-ta-set
Response from the server:  Imported trust anchor file successfully
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages enable
Currently active profile :  default
Command Name:  plt-sign-pkg-enable
Successfully updated the signed package deployment capability on the device to true
```

Une autre option consiste à importer l'ancrage d'approbation via Local Manager :

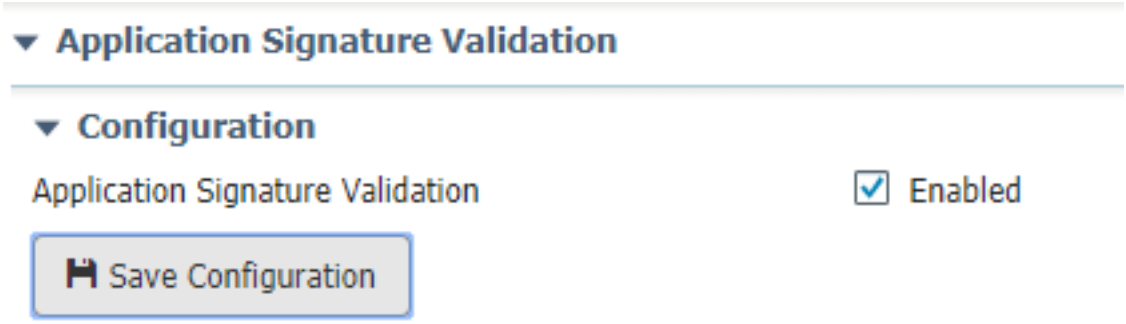
Accédez à **System Setting > Import Trust Anchor** comme indiqué dans l'image.



Sélectionnez le fichier que vous avez généré à l'étape 2. et cliquez sur **OK** comme indiqué dans l'image.



Après avoir importé correctement l'ancrage d'approbation, cochez **Enabled** for **Application Signing Validation** et cliquez sur **Save Configuration** comme indiqué dans l'image :



Étape 4. Créer une clé spécifique à l'application et un CSR

Ensuite, vous pouvez créer une paire de clés et de certificats utilisée pour vous connecter à votre application IOx. La meilleure pratique consiste à générer une paire de clés spécifique pour chaque application que vous prévoyez de déployer.

Tant que chacun de ces documents est signé avec la même autorité de certification, ils sont tous considérés comme valides.

Afin de générer la clé spécifique à l'application :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl genrsa -out app-key.pem 2048
Generating RSA private key, 2048 bit long modulus
.......+++
...+++
e is 65537 (0x10001)
```

Afin de générer le CSR :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl req -new -key app-key.pem -out app.csr
You are about to be asked to enter information that is incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name (DN).
There are quite a few fields but you can leave some blank.
For some fields there can be a default value,
If you enter '.', the field can be left blank.
-----
Country Name (2 letter code) [XX]:BE
State or Province Name (full name) []:WVL
Locality Name (eg, city) [Default City]:Kortrijk
Organization Name (eg, company) [Default Company Ltd]:Cisco
Organizational Unit Name (eg, section) []:IOT
Common Name (eg, your name or your server's hostname) []:ioxapp
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Comme pour l'autorité de certification, les valeurs du certificat de demande doivent être ajustées pour correspondre à votre cas d'utilisation.

Étape 5. Signer un certificat spécifique à l'application avec CA

Maintenant que vous avez les conditions requises pour votre CA et votre application CSR, vous pouvez signer le CSR avec l'utilisation de CA. Le résultat est un certificat spécifique à l'application signé :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl x509 -req -in app.csr -CA rootca-cert.pem -CAkey
rootca-key.pem -CAcreateserial -out app-cert.pem -days 4096 -sha256
Signature ok
subject=/C=BE/ST=WVL/L=Kortrijk/O=Cisco/OU=IOT/CN=ioxapp
Getting CA Private Key
```

Étape 6. Empaquetez votre application IOx et signez-la avec un certificat spécifique à l'application

Àce stade, vous êtes prêt à emballer votre application IOx et à la signer avec la paire de clés générée à l'étape 4. et signé par l'autorité de certification à l'étape 5.

Le reste du processus de création de la source et de package.yaml pour votre application reste inchangé.

application IOx de package avec l'utilisation de la paire de clés :

```
[jedepuyd@KJK-SRVIOT-10 iox_docker_pythonsleep]$ ioxclient package --rsa-key ../signing/app-
key.pem --certificate ../signing/app-cert.pem .
Currently active profile :  default
Command Name:  package
Using rsa key and cert provided via command line to sign the package
Checking if package descriptor file is present..
Validating descriptor file /home/jedepuyd/iox/iox_docker_pythonsleep/package.yaml with package
schema definitions
Parsing descriptor file..
Found schema version  2.2
Loading schema file for version  2.2
Validating package descriptor file..
File /home/jedepuyd/iox/iox_docker_pythonsleep/package.yaml is valid under schema version 2.2
Created Staging directory at :  /tmp/666018803
Copying contents to staging directory
Checking for application runtime type
Couldn't detect application runtime type
Creating an inner envelope for application artifacts
Excluding  .DS_Store
Generated  /tmp/666018803/artifacts.tar.gz
Calculating SHA1 checksum for package contents..
Package MetaData file was not found at  /tmp/666018803/.package.metadata
Wrote package metadata file :  /tmp/666018803/.package.metadata
Root Directory :  /tmp/666018803
Output file:  /tmp/096960694
Path:  .package.metadata
SHA1 : 2a64461a921c2d5e8f45e92fe203127cf8a06146
Path:  artifacts.tar.gz
SHA1 : 63da3eb3d81e13249b799bf57845f3fc9f6f2f94
Path:  package.yaml
SHA1 : 0e6259e49ff22d6d38e6d1913759c5674c5cec6d
Generated package manifest at  package.mf
Signed the package and the signature is available at package.cert
Generating IOx Package..
Package generated at /home/jedepuyd/iox/iox_docker_pythonsleep/package.tar
```

Étape 7. Déployez votre package IOx signé sur un périphérique prenant en charge

les signatures

La dernière étape du processus consiste à déployer l'application sur votre périphérique IOx. Il n'y a aucune différence par rapport à un déploiement d'application non signé :

```
[jedepuyd@KJK-SRVIOT-10 iox_docker_pythonsleep]$ ioxclient app install test package.tar
Currently active profile :  default
Command Name:  application-install
Saving current configuration
Installation Successful. App is available at :
https://10.50.215.248:8443/iox/api/v2/hosting/apps/test
Successfully deployed
```

Vérification

Utilisez cette section pour confirmer que votre configuration fonctionne correctement.

Afin de vérifier si une clé d'application est correctement signée avec votre autorité de certification, vous pouvez faire ceci :

```
[jedepuyd@KJK-SRVIOT-10 signing]$ openssl verify -CAfile rootca-cert.pem app-cert.pem
app-cert.pem: OK
```

Dépannage

Cette section fournit des informations que vous pouvez utiliser pour dépanner votre configuration.

Lorsque vous rencontrez des problèmes avec le déploiement d'applications, vous pouvez voir l'une des erreurs suivantes :

```
[jedepuyd@KJK-SRVIOT-10 iox_docker_pythonsleep]$ ioxclient app install test package.tar
Currently active profile :  default
Command Name:  application-install
Saving current configuration
Could not complete your command :  Error. Server returned 500
{
 "description": "Invalid Archive file: Certificate verification failed: [18, 0, 'self signed
certificate']",
 "errorcode": -1,
 "message": "Invalid Archive file"
}
```

Un problème s'est produit lors de la signature du certificat d'application avec l'utilisation de l'autorité de certification ou il ne correspond pas à celui du bundle d'ancrage approuvé.

Utilisez les instructions mentionnées dans la section Vérifier pour vérifier vos certificats et également l'ensemble d'ancres de confiance.

Cette erreur indique que votre paquet n'a pas été signé correctement, vous pouvez regarder dans l'étape 6. encore une fois.

```
[jedepuyd@KJK-SRVIOT-10 iox_docker_pythonsleep]$ ioxclient app install test2 package.tar
Currently active profile :  default
```

```
Command Name:  application-install
Saving current configuration
Could not complete your command :  Error. Server returned 500
{
 "description": "Package signature file package.cert or package.sign not found in package",
 "errorcode": -1009,
 "message": "Error during app installation"
}
```