Konfigurieren der IOx-Paketsignaturvalidierung

Inhalt

Einführung

In diesem Dokument wird detailliert beschrieben, wie signierte Pakete auf der IOx-Plattform erstellt und verwendet werden.

Voraussetzungen

Anforderungen

Cisco empfiehlt, über Kenntnisse in folgenden Bereichen zu verfügen:

- Grundlegendes Linux-Wissen
- Funktionsweise von Zertifikaten

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf den folgenden Software- und Hardwareversionen:

- IOx-fähiges Gerät, das für IOx konfiguriert ist:
  IP-Adresse konfiguriertGast-Betriebssystem (GOS) und Cisco Application Framework (CAF), die ausgeführt werdenNetwork Address Translation (NAT), konfiguriert für den Zugriff auf CAF (Port 8443)
- Linux-Host mit installiertem offenen Secure Sockets Layer (SSL)

- IOx-Client-Installationsdateien, die unter folgender Adresse heruntergeladen werden können: https://software.cisco.com/download/release.html?mdfid=286306005&softwareid=286306762

Die Informationen in diesem Dokument wurden von den Geräten in einer bestimmten Laborumgebung erstellt. Alle in diesem Dokument verwendeten Geräte haben mit einer leeren (Standard-)Konfiguration begonnen. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen eines Befehls verstehen.

Hintergrundinformationen

Seit IOx-Version wird die Signierung von AC5-Anwendungspaketen unterstützt. Diese Funktion stellt sicher, dass das Anwendungspaket gültig ist und das auf dem Gerät installierte Paket von einer vertrauenswürdigen Quelle bezogen wird. Wenn die Signatursignaturüberprüfung für Anwendungspakete auf einer Plattform aktiviert ist, können nur signierte Anwendungen bereitgestellt werden.

Konfigurieren

Die folgenden Schritte sind erforderlich, um die Paketsignaturüberprüfung zu verwenden:

1. Erstellen Sie einen Zertifizierungsstellen-Schlüssel und ein Zertifikat.
2. Generieren Sie einen Vertrauensanker für die Verwendung in IOx.
3. Importieren Sie den Vertrauensanker auf Ihrem IOx-Gerät.
4. Erstellen Sie einen anwendungsspezifischen Schlüssel und eine CSR-Anfrage (Certificate Signing Request).
5. Signieren Sie das anwendungsspezifische Zertifikat mit der CA.
6. Verpacken Sie Ihre IOx-Anwendung, und signieren Sie sie mit dem anwendungsspezifischen Zertifikat.
7. Stellen Sie Ihr signiertes IOx-Paket auf einem signaturfähigen Gerät bereit.

Hinweis: In diesem Artikel wird eine selbstsignierte CA in einem Produktionsszenario verwendet. Die beste Option ist, eine offizielle CA oder die CA Ihres Unternehmens zu unterschreiben.

Hinweis: Die Optionen für CA, Schlüssel und Signaturen werden nur für Laborzwecke ausgewählt und müssen möglicherweise an Ihre Umgebung angepasst werden.

Schritt 1: CA-Schlüssel und -Zertifikat erstellen

Der erste Schritt besteht darin, eine eigene CA zu erstellen. Dies kann einfach durch Generieren eines Schlüssels für die CA und eines Zertifikats für diesen Schlüssel erfolgen:

So generieren Sie den CA-Schlüssel:

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

So generieren Sie das Zertifizierungsstellenzertifikat:

```
[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 []:
```

Die Werte im Zertifizierungsstellenzertifikat müssen an Ihren Anwendungsfall angepasst werden.

Schritt 2: Trust Anchor für IOx erstellen

Da Sie jetzt über den erforderlichen Schlüssel und das erforderliche Zertifikat für Ihre CA verfügen, können Sie ein Ankerpaket für Vertrauensstellungszwecke für die Verwendung auf Ihrem IOx-Gerät erstellen. Das Vertrauensanker-Paket muss die vollständige Zertifizierungsstellenkette der Zertifizierungsstelle (falls Zwischenzertifikat für die Signierung verwendet wird) und eine Datei info.txt enthalten, die zur Bereitstellung der Metadaten (frei zugängliches Formular) verwendet wird.

Erstellen Sie zunächst die Datei info.txt, und fügen Sie einige Metadaten hinzu:

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

Wenn Sie mehrere Zertifizierungsstellenzertifikate besitzen, müssen Sie diese optional in einem .pem zusammenfassen, um die Zertifizierungsstellenkette zu bilden:

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

> **Hinweis**: Dieser Schritt ist für diesen Artikel nicht erforderlich, da ein einzelnes CA-Root-Zertifikat zum Direktzeichen verwendet wird, wird dies für die Produktion nicht empfohlen und das Root-CA-Schlüsselpaar muss immer offline gespeichert werden.

Die Zertifizierungsstellenkette der Zertifizierungsstelle muss den Namen ca-chain.cert.pem tragen. Bereiten Sie diese Datei also vor:

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

Schließlich können Sie die Dateien ca-chain.cert.pem und info.txt in einem gezippten Tar

kombinieren:

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

Schritt 3: Trust Anchor auf IOx-Gerät importieren

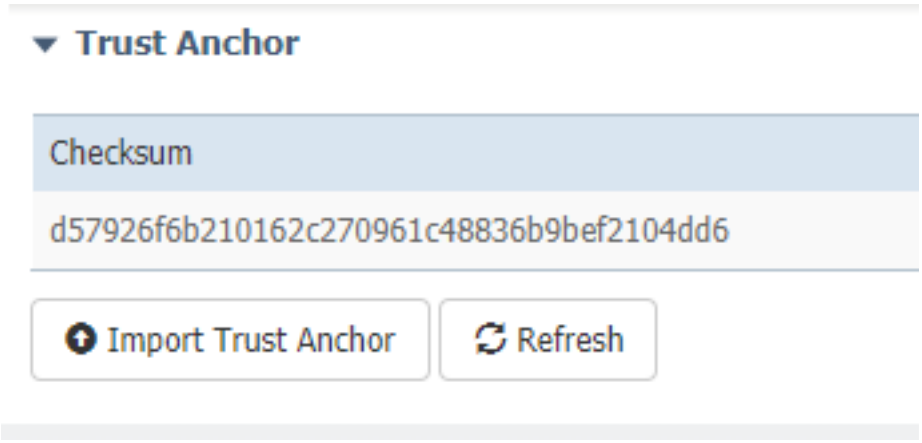
Der im vorherigen Schritt erstellte trustanchorv1.tar.gz muss auf das IOx-Gerät importiert werden. Die Dateien im Paket werden verwendet, um zu überprüfen, ob eine Anwendung mit einem Zertifikat signiert wurde, das von einer Zertifizierungsstelle signiert wurde, bevor sie eine Installation zulässt.

Der Import des Vertrauensankers kann über den ioxlcient erfolgen:

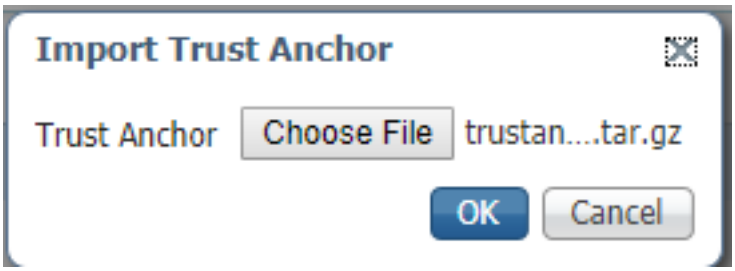
```
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages trustanchor set
trustanchorv1.tar.gz
Currently active profile :  default
Command Name:  plt-sign-pkg-ta-set
Response from the server:  Imported trust anchor file successfully
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages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

Eine weitere Option besteht darin, den Vertrauensanker über den lokalen Manager zu importieren:

Navigieren Sie zu **Systemeinstellungen > Vertrauenswürdigen Anker importieren**, wie im Bild gezeigt.

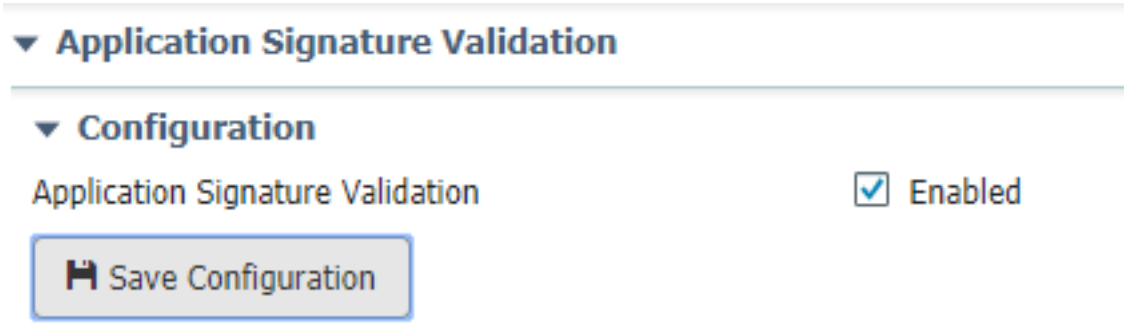


Wählen Sie die Datei aus, die Sie in Schritt 2 generiert haben. und klicken Sie auf **OK**, wie im Bild gezeigt.



Nachdem Sie den Vertrauensanker erfolgreich importiert haben, aktivieren Sie **Enabled** for

Application Signing Validation (Aktiviert für die **Validierung der Anwendungssignatur),** und klicken Sie auf **Save Configuration**, wie im Bild gezeigt:



Schritt 4: Erstellen eines anwendungsspezifischen Schlüssels und CSR

Anschließend können Sie ein Schlüssel- und Zertifikatspaar erstellen, das zur Anmeldung bei der IOx-Anwendung verwendet wird. Die Best Practice besteht darin, für jede Anwendung, die Sie bereitstellen möchten, eine bestimmte Tastatur zu generieren.

Solange alle mit derselben CA signiert sind, gelten sie als gültig.

So generieren Sie den anwendungsspezifischen Schlüssel:

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

So generieren Sie die 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 []:
```

Wie bei der CA müssen die Werte im Anwendungszertifikat an Ihren Anwendungsfall angepasst werden.

Schritt 5: Signieren eines anwendungsspezifischen Zertifikats mit CA

Nachdem Sie nun die Anforderungen für Ihre CA und Anwendungs-CSR erfüllt haben, können Sie die CSR mit CA signieren. Das Ergebnis ist ein signiertes anwendungsspezifisches Zertifikat:

```
[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
```

Schritt 6: Verpacken Sie die IOx-Anwendung, und signieren Sie sie mit einem anwendungsspezifischen Zertifikat.

An diesem Punkt sind Sie bereit, Ihre IOx-Anwendung zu packen und mit der generierten Tastatur aus Schritt 4 zu signieren. und von der Zertifizierungsstelle in Schritt 5 unterzeichnet.

Der restliche Prozess zum Erstellen der source und package.yaml für die Anwendung bleibt unverändert.

Paket-IOx-Anwendung mit der Verwendung von Tastenfeld:

```
[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
```

Schritt 7: Bereitstellen des signierten IOx-Pakets auf einem signaturfähigen Gerät

Der letzte Schritt des Prozesses ist die Bereitstellung der Anwendung auf Ihrem IOx-Gerät. Im Vergleich zu einer nicht signierten Anwendungsbereitstellung besteht kein Unterschied:

```
[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
```

Überprüfen

In diesem Abschnitt überprüfen Sie, ob Ihre Konfiguration ordnungsgemäß funktioniert.

Um zu überprüfen, ob ein Anwendungsschlüssel korrekt mit Ihrer CA signiert ist, können Sie Folgendes tun:

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

Fehlerbehebung

Dieser Abschnitt enthält Informationen, die Sie zur Fehlerbehebung bei Ihrer Konfiguration verwenden können.

Wenn Probleme bei der Anwendungsbereitstellung auftreten, kann einer der folgenden Fehler auftreten:

```
[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"
}
```

Beim Signieren des Anwendungszertifikats mithilfe der CA ist etwas schief gegangen, oder es stimmt nicht mit dem Zertifikat im Ankerpaket überein.

Überprüfen Sie anhand der Anweisungen im Abschnitt Überprüfen Ihre Zertifikate und auch das Ankerpaket.

Diese Fehlermeldung weist darauf hin, dass Ihr Paket nicht richtig signiert wurde. Sie können Schritt 6 aufrufen. wieder.

```
[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"
}
```