

Configura convalida firma pacchetto IOx

Sommario

[Introduzione](#)

[Prerequisiti](#)

[Requisiti](#)

[Componenti usati](#)

[Premesse](#)

[Configurazione](#)

[Passaggio 1. Creazione della chiave e del certificato CA](#)

[Passaggio 2. Generare il trust anchor da utilizzare su IOx](#)

[Passaggio 3. Importazione del trust anchor sul dispositivo IOx](#)

[Passaggio 4. Creazione della chiave specifica dell'applicazione e di CSR](#)

[Passaggio 5. Firmare il certificato specifico dell'applicazione con la CA](#)

[Passaggio 6. Creare il pacchetto dell'applicazione IOx e firmarla con il certificato specifico dell'applicazione](#)

[Passaggio 7. Distribuire il pacchetto IOx firmato su un dispositivo abilitato alla firma](#)

[Verifica](#)

[Risoluzione dei problemi](#)

Introduzione

In questo documento viene descritto in modo dettagliato come creare e utilizzare pacchetti firmati sulla piattaforma IOx.

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- Conoscenze base di Linux
- Informazioni sul funzionamento dei certificati

Componenti usati

Le informazioni fornite in questo documento si basano sulle seguenti versioni software e hardware:

- Dispositivo compatibile con IOx configurato per IOx:
 - Indirizzo IP configurato
 - Sistema operativo guest (GOS) e Cisco Application Framework (CAF) in esecuzione
 - Network Address Translation (NAT) configurato per l'accesso a CAF (porta 8443)
- Host Linux con SSL (Secure Sockets Layer) aperto installato

- File di installazione del client IOx scaricabili da:

<https://software.cisco.com/download/release.html?mdfid=286306005&softwareid=286306762>

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Premesse

A partire dalla versione IOx, è supportata la firma dei pacchetti dell'applicazione AC5. Questa funzionalità consente di garantire che il pacchetto dell'applicazione sia valido e che quello installato nel dispositivo sia ottenuto da una fonte attendibile. Se la convalida della firma del pacchetto dell'applicazione è attivata in una piattaforma, è possibile distribuire solo le applicazioni firmate.

Configurazione

Per utilizzare la convalida della firma del pacchetto, è necessario eseguire i passaggi seguenti:

1. Creare una chiave e un certificato CA.
2. Generare un trust anchor da utilizzare in IOx.
3. Importare il trust anchor sul dispositivo IOx.
4. Creare una chiave specifica dell'applicazione e una richiesta di firma del certificato (CSR).
5. Firmare il certificato specifico dell'applicazione utilizzando la CA.
6. Creare il pacchetto dell'applicazione IOx e firmarlo con il certificato specifico dell'applicazione.
7. Distribuire il pacchetto IOx firmato in un dispositivo abilitato alla firma.

Nota: Per questo articolo, in uno scenario di produzione viene utilizzata una CA autofirmata. L'opzione migliore consiste nell'utilizzare un'autorità di certificazione ufficiale o l'autorità di certificazione della società per firmare.

Nota: Le opzioni per la CA, le chiavi e le firme vengono scelte solo per scopi di laboratorio e potrebbe essere necessario modificarle in base all'ambiente.

Passaggio 1. Creazione della chiave e del certificato CA

Il primo passaggio consiste nella creazione di una CA personalizzata. A tale scopo, è sufficiente generare una chiave per la CA e un certificato per tale chiave:

Per generare la chiave 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)
```

Per generare il certificato CA:

```
[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 []:
```

I valori nel certificato CA devono essere regolati in modo da corrispondere allo Use Case.

Passaggio 2. Generare il trust anchor da utilizzare su IOx

Ora che si dispone della chiave e del certificato necessari per la CA, è possibile creare un pacchetto trust anchor da utilizzare sul dispositivo IOx. Il bundle trust anchor deve contenere la catena di firma CA completa (nel caso in cui per la firma vengano utilizzati certificati intermedi) e un file info.txt utilizzato per fornire i metadati (in formato libero).

Creare innanzitutto il file info.txt e inserirvi alcuni metadati:

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

Facoltativamente, se si dispone di più certificati CA, per formare la catena di certificati CA è necessario riunirli in un unico file con estensione pem:

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

Nota: Questo passaggio non è necessario per questo articolo, poiché per la firma diretta viene utilizzato un singolo certificato radice CA, questa operazione non è consigliata per la produzione e la coppia di chiavi radice CA deve essere sempre archiviata offline.

La catena di certificati CA deve essere denominata ca-chain.cert.pem. Preparare il file:

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

Infine, è possibile combinare ca-chain.cert.pem e info.txt in un tar compresso:

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

Passaggio 3. Importazione del trust anchor sul dispositivo IOx

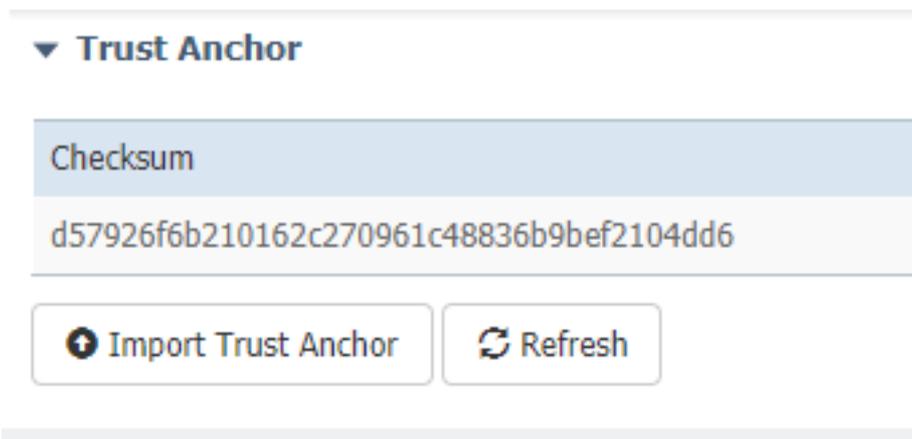
Il file trustanchorv1.tar.gz creato nel passaggio precedente deve essere importato nel dispositivo IOx. I file nel bundle vengono utilizzati per verificare se un'applicazione è stata firmata con un certificato CA dalla CA corretta prima di consentire un'installazione.

L'importazione del trust anchor può essere eseguita tramite ioxclient:

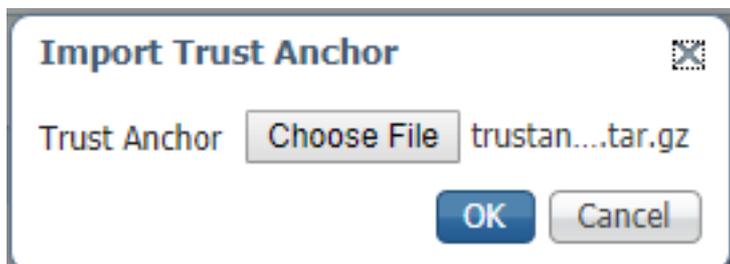
```
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages trustanchor set trustanchorv1.tar.gz
Currently active profile : default
Command Name: plt-sign-pkg-ta-set
Response from the server: Imported trust anchor file successfully
[jedepuyd@KJK-SRVIOT-10 signing]$ ioxclient platform signedpackages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

Un'altra opzione consiste nell'importare il trust anchor tramite Gestione locale:

Passare a **Impostazioni di sistema > Importa trust anchor** come mostrato nell'immagine.



Selezionare il file generato al punto 2 e fare clic su **OK** come mostrato nell'immagine.



Dopo aver importato correttamente il trust anchor, selezionare **Enabled** (Attivato) per **Convalida firma applicazione** e fare clic su **Save Configuration** (Salva configurazione) come mostrato nell'immagine:

▼ Application Signature Validation

▼ Configuration

Application Signature Validation

Enabled

 Save Configuration

Passaggio 4. Creazione della chiave specifica dell'applicazione e di CSR

È quindi possibile creare una coppia di chiavi e certificati da utilizzare per accedere all'applicazione IOx. È consigliabile generare una coppia di chiavi specifica per ogni applicazione che si intende distribuire.

Finché sono firmate con la stessa CA, tutte sono considerate valide.

Per generare la chiave specifica dell'applicazione:

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

Per generare la RSI:

```
[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 []:
```

Come per la CA, i valori nel certificato dell'applicazione devono essere adattati in base allo Use Case.

Passaggio 5. Firmare il certificato specifico dell'applicazione con la CA

Dopo avere definito i requisiti per la CA e il CSR dell'applicazione, è possibile firmare il CSR

utilizzando CA. Il risultato è un certificato firmato specifico dell'applicazione:

```
[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
```

Passaggio 6. Creare il pacchetto dell'applicazione IOx e firmarla con il certificato specifico dell'applicazione

A questo punto, è possibile creare il pacchetto dell'applicazione IOx e firmarla con la coppia di chiavi generata dal passaggio 4. e firmata dalla CA nel passaggio 5.

Il resto del processo di creazione dell'origine e del package.yaml per l'applicazione rimane invariato.

creare il pacchetto dell'applicazione IOx con l'utilizzo della coppia di chiavi:

```
[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
```

Passaggio 7. Distribuire il pacchetto IOx firmato su un dispositivo abilitato alla firma

L'ultimo passaggio consiste nella distribuzione dell'applicazione sul dispositivo IOx. Non esistono

differenze rispetto a una distribuzione di applicazioni non firmate:

```
[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
```

Verifica

Fare riferimento a questa sezione per verificare che la configurazione funzioni correttamente.

Per verificare che una chiave dell'applicazione sia firmata correttamente con la CA, eseguire questa operazione:

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

Risoluzione dei problemi

Le informazioni contenute in questa sezione permettono di risolvere i problemi relativi alla configurazione.

Quando si verificano problemi con la distribuzione delle applicazioni, è possibile che si verifichi uno dei seguenti errori:

```
[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"
}
```

Si è verificato un errore durante la firma del certificato dell'applicazione con l'utilizzo della CA o il certificato non corrisponde a quello nel bundle di ancoraggio attendibile.

Utilizzare le istruzioni indicate nella sezione Verifica per controllare i certificati e anche il pacchetto di ancoraggio attendibile.

Questo errore indica che il pacchetto non è stato firmato correttamente. È possibile esaminare di nuovo il passaggio 6.

```
[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"  
}
```