Adobe ADT Code Signing

SafeNet Luna HSM Integration Guide

gemalto
security to be free

Document Part Number: 007-000262-001, Rev. A

Release Date: January 2019

Contents

Preface

This document guides administrator users through generating a signing key on a SafeNet Luna HSM or HSM on Demand service and using the signing key with Adobe ADT Signing utility to sign an .air file.

Scope

This guide demonstrates using a SafeNet Luna HSM or HSM on Demand signing key to sign a .air file using the Adobe ADT Signing utility.

Document Conventions

This section provides information on the conventions used in this template.

Notes

Notes are used to alert you to important or helpful information. These elements use the following format:

NOTE: Take note. Contains important or helpful information.

Cautions

Cautions are used to alert you to important information that may help prevent unexpected results or data loss. These elements use the following format:

CAUTION: Exercise caution. Caution alerts contain important information that may help prevent unexpected results or data loss.

Warnings

Warnings are used to alert you to the potential for catastrophic data loss or personal injury. These elements use the following format:

WARNING: Be extremely careful and obey all safety and security measures. In this situation you might do something that could result in catastrophic data loss or personal injury.

Command Syntax and Typeface Conventions

Convention	Description
bold	The bold attribute is used to indicate the following: • Command-line commands and options (Type **dir /p**.) • Button names (Click **Save As**.) • Check box and radio button names (Select the **Print Duplex** check box.) • Window titles (On the **Protect Document** window, click **Yes**.) • Field names (**User Name:** Enter the name of the user.) • Menu names (On the **File** menu, click **Save**.) (Click **Menu** > **Go To** > **Folders**.) • User input (In the **Date** box, type **April 1**.)
italic	The italic attribute is used for emphasis or to indicate a related document. (See the *Installation Guide* for more information.)
`Consolas`	Denotes syntax, prompts, and code examples.

Support Contacts

Contact Method	Contact Information	
Address	Gemalto 4690 Millennium Drive Belcamp, Maryland 21017, USA	
Phone	US	1-800-545-6608
	International	1-410-931-7520
Technical Support Customer Portal	https://supportportal.gemalto.com Existing customers with a Technical Support Customer Portal account can log in to manage incidents, get the latest software upgrades, and access the Gemalto Knowledge Base.	

1
Introduction

Overview

Adobe AIR Developer Tool (ADT) is a multi-purpose, command-line tool for developing AIR applications. ADT is a Java program included in the AIR SDK. The SDK includes a script file for invoking ADT. Please refer to the below URL to download and configure Adobe AIR SDK:

https://www.adobe.com/devnet/air/air-sdk-download.html

You can perform the following tasks using ADT:

- Package an Signed AIR application as an .air installation file.
- Package an Signed AIR application as a native installer file type—for example, as a .exe installer file on Windows, .ipa on iOS, or .apk on Android.
- Package a native extension as an AIR Native Extension (ANE) file.
- Sign an AIR application with a digital certificate.
- Change (migrate) the digital signature used for application updates.
- Determine the devices connected to a computer.
- Create a self-signed digital code signing certificate.

Digitally signing your AIR installation files with a certificate issued by a recognized certification authority (CA) provides significant assurance to your users that the application they are installing has not been accidentally or maliciously altered and identifies you as the signer (publisher). AIR displays the publisher name during installation when the AIR application has been signed with a certificate that is trusted, or which chains to a certificate that is trusted on the installation computer.

This guide demonstrates how to sign an Adobe ADT application using a signing key generated on a SafeNet Luna HSM or HSM on Demand service.

Using a SafeNet Luna HSM or HSM on Demand service to generate the RSA keys for Adobe ADT provides the following benefits:

- Secure generation, storage and protection of the signing private keys on FIPS 140-2 level 3 validated hardware.
- Full life cycle management of the keys.
- Access to the HSM audit trail**.
- Take advantage of cloud services with confidence.
- Significant performance improvements by off-loading cryptographic operations from signing servers.

**HSMoD services do not have access to the secure audit trail

3rd Party Application Details

- Adobe AIR SDK & Compiler or Adobe Flex SDK.

Supported Platforms

List of the platforms which are tested with the following HSMs:

SafeNet Luna HSM: SafeNet Luna HSM appliances are purposefully designed to provide a balance of security, high performance, and usability that makes them an ideal choice for enterprise, financial, and government organizations. SafeNet Luna HSMs physically and logically secure cryptographic keys and accelerate cryptographic processing.

The SafeNet Luna HSM on premise offerings include the SafeNet Luna Network HSM, SafeNet PCIe HSM, and SafeNet Luna USB HSMs. SafeNet Luna HSMs are also available for access as an offering from cloud service providers such as IBM cloud HSM and AWS cloud HSM classic.

Platforms	Java Version
Windows 10	JDK8

SafeNet DPoD: SafeNet Data Protection on Demand (DPoD) is a cloud-based platform that provides on-demand HSM and Key Management services through a simple graphical user interface. With DPoD, security is simple, cost effective and easy to manage because there is no hardware to buy, deploy and maintain. As an Application Owner, you click and deploy services, generate usage reports and maintain just the services you need.

Platforms	Java Version
Window 10	JDK8

Prerequisites

Before you begin

Before starting the integration of Adobe ADT with SafeNet HSM, ensure that you have configured SafeNet Luna Network HSM or provisioned SafeNet HSM Data Protection on Demand service as per the requirement.

Configuring SafeNet Luna HSM

Before you get started:

1. Ensure that the HSM is setup, initialized, provisioned and ready for deployment. Refer to the HSM product documentation for help.
2. Create a partition on the HSM that will be later used by Adobe ADT to generate a signed AIR Application.
3. If using a SafeNet Luna Network HSM, register a client for the system and assign the client to the partition to create an NTLS connection. Initialize Crypto Officer and Crypto User roles for the registered partition.

4. Ensure that the partition is successfully registered and configured. The command to see the registered partitions is:

```
# /usr/safenet/lunaclient/bin/lunalcm

lunacm.exe (64-bit) v7.2.0-219. Copyright (c) 2018 SafeNet. All rights reserved.

Available HSMs:

Slot Id ->              0
Label ->                Adobe
Serial Number ->        1213475834492
Model ->                LunaSA 7.2.0
Firmware Version ->     7.2.0
Configuration ->        Luna User Partition With SO (PW) Signing With Cloning Mode
Slot Description ->     Net Token Slot

Current Slot Id: 0
```

NOTE: Follow the SafeNet Network Luna HSM documentation for detailed steps for creating an NTLS connection, initializing the partition and initializing the user roles.

SafeNet Luna HSM HA (High-Availability) Setup

Refer to the SafeNet Luna HSM documentation for HA steps and details to configure two or more HSMs on host systems. You must enable the HAOnly setting in HA for failover to work so that if the primary becomes unavailable all calls are automatically routed to the secondary until primary is available again.

Provision your HSM on Demand service

This service provides your client machine with access to an HSM partition for storing cryptographic objects used by your applications. Partitions can be assigned to a single client, or multiple clients can be assigned to, and share, a single partition.

To provision your application partition, start by initializing the following roles:

- **Security Officer (SO)** - responsible for setting the partition policies and for creating the Crypto Officer.
- **Crypto Officer (CO)** - responsible for creating, modifying and deleting crypto objects within the partition. The CO can use the crypto objects and create an optional, limited-capability role called Crypto User.
- **Crypto User (CU)** - an optional role that can use crypto objects to perform cryptographic operations but cannot modify the objects.

NOTE: Refer the “SafeNet Data Protection on Demand Application Owner Quick Start Guide” to configure the HSM on Demand service and create a service client.

The HSM service client package is a zip file that contains system information needed to connect your client machine to an existing HSM on Demand service.

Constraints on HSM on Demand Services

HSM on Demand Service in FIPS mode

HSMoD services operate in a FIPS and non-FIPS mode. If your organization requires non-FIPS algorithms for your operations, ensure you enable the Allow non-FIPS approved algorithms check box when configuring your HSM on Demand service. The FIPS mode is enabled by default.

Refer to the *Mechanism List* in the *SDK Reference Guide* for more information about available FIPS and non-FIPS algorithms.

Verify HSM on Demand <slot> value

LunaCM commands work on the current slot. If there is only one slot, then it is always the current slot. If you are completing an integration using HSMoD services, you need to verify which slot on the HSMoD service you send commands to. If there is more than one slot, then use the slot set command to direct a command to a specified slot. You can use slot list to determine which slot numbers are in use by which HSMoD service.

Using SafeNet HSM in FIPS Mode

Under FIPS 186-3/4, the RSA methods permitted for generating keys are 186-3 with primes and 186-3 with aux primes. This means that RSA PKCS and X9.31 key generation is no longer approved for operation in a FIPS-compliant HSM. If you are using the SafeNet Luna HSM or HSM on Demand service in FIPS mode, you have to make the following change in the Chrystoki.conf (Linux) or crystoki.ini (Windows) configuration file:

```
[Misc]
RSAKeyGenMechRemap=1
```

The above setting redirects the older calling mechanism to a new approved mechanism when SafeNet Luna HSM or HSMoD is in FIPS mode.

Install Java Development Kit

Ensure that the Java Development Kit (JDK) is installed on your server or local computer. You can run the commands in this instruction wherever you have the keytool command available.

2

Adobe AIR Code Signing using SafeNet HSMs

Configuring Java for Adobe AIR Code Signing using SafeNet HSMs

You can use the Java keytool utility to generate signing keys and certificate on SafeNet HSMs and Adobe ADT Utility to sign the AIR file.

To configure the java.security file

1. Edit the Java Security Configuration file java.security located in the directory <JDK_installation_directory>/jre/lib/security

   Add the Luna Provider to the **java.security** file as shown below:

   ```
   security.provider.1=sun.security.provider.Sun
   security.provider.2=sun.security.rsa.SunRsaSign
   security.provider.3=com.sun.net.ssl.internal.ssl.Provider
   security.provider.4=com.sun.crypto.provider.SunJCE
   security.provider.5=sun.security.jgss.SunProvider
   security.provider.6=com.sun.security.sasl.Provider
   security.provider.7=org.jcp.xml.dsig.internal.dom.XMLDSigRI
   security.provider.8=sun.security.smartcardio.SunPCSC
   security.provider.9=com.safenetinc.luna.provider.LunaProvider
   ```

   Save the changes to the **java.security** file.

To create HSM keystore

1. Copy the LunaAPI.dll and LunaProvider.jar file from the <Luna_installation_directory>/jsp/lib folder to the Java extension folder under <JDK_installation_directory>/jre/lib/ext.
2. Set the environment variables for JAVA_HOME and PATH.

   NOTE: For Windows set the JDK bin folder in PATH variable under System Environments.

3. Create a blank file named lunastore and add the following entry where <Partition Name> would be your Luna HSM partition label:

   ```
   tokenlabel:<partition_label>
   ```

   Save the file in the current working directory.

To generate key for signing the .air file

1. Generate a key pair using the Java keytool utility in the keystore.

```
keytool -genkeypair -alias lunakey -keyalg RSA -sigalg SHA256withRSA -keypass temp123# -keysize
2048 -keystore lunastore -storepass temp123# -storetype luna
```

```
What is your first and last name?
[Unknown]:  HSM
What is the name of your organizational unit?
[Unknown]:  HSM
What is the name of your organization?
[Unknown]:  Gemalto
What is the name of your City or Locality?
[Unknown]:  MyCity
What is the name of your State or Province?
[Unknown]:  MyState
What is the two-letter country code for this unit?
[Unknown]:  IN
Is CN=HSM, OU=HSM, O=Gemalto, L=MyCity, ST=MyState, C=IN correct?
[no]:  yes
```

A new key pair generates on the SafeNet Luna HSM or HSMoD service.

NOTE: The command above used “temp123#” as storepass which is the partition Crypto Officer Pin you set when initializing the CO role for the partition.

2. Verify that the private key is in the SafeNet Luna HSM or HSMoD service.

```
keytool -list -v -storetype luna -keystore lunastore
```

The system prompt to enter the keystore password and after providing the password it display the contents.

```
Enter keystore password:

Keystore type: LUNA
Keystore provider: LunaProvider

Your keystore contains 1 entry

Alias name: lunakey
Creation date: Apr 16, 2018
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=HSM, OU=HSM, O=Gemalto, L=MyCity, ST=MyState, C=IN
Issuer: CN=HSM, OU=HSM, O=Gemalto, L=MyCity, ST=MyState, C=IN
Serial number: 1353bc67
Valid from: Mon Apr 16 12:01:45 PDT 2018 until: Sun Jul 15 12:01:45 PDT 2018Certificate
fingerprints:
MD5:  90:D9:4A:25:DD:C4:9E:7F:55:60:3D:ED:D0:84:18:C1
SHA1: 01:FF:94:6B:24:3C:FB:5F:05:F9:7F:AC:3A:3B:4D:AB:0D:9A:69:36
SHA256:
FD:09:09:3A:71:1C:69:A1:24:5E:78:AB:BB:7C:0C:D9:81:02:64:D2:AE:7C:A1:00:91:21:EA:41:9E:3D:FA:0D
Signature algorithm name: SHA256withRSA
Version: 3
*******************************************
*******************************************
```

3. Generate a certificate request from a key in the keystore. The system will prompt you for the keystore password.

```
keytool -certreq -alias lunakey -sigalg SHA256withRSA -file certreq_file -storetype luna
keystore lunastore
```

Enter the keystore password:
The certreq_file is generated in the current directory.

NOTE: After creating your certificate request, ensure that you keep track of your keystore file as it contains the private key. In addition, you require the keystore file to install your Code Signing Certificate.

4. Submit the CSR file to your Certification Authority (CA). Have the CA authenticate the request with the Code Signing template and return a signed certificate or a certificate chain. Save the reply and the root certificate of the CA in the current working directory.
5. Import the CA's Root certificate and signed certificate or certificate chain in to the keystore.

To import the CA root certificate execute the following:

```
keytool -trustcacerts -importcert -alias rootca -file root.cer -keystore lunastore –storetype
luna
```

To import the signed certificate reply or certificate chain execute the following:

```
keytool -trustcacerts -importcert -alias lunakey -file signing.p7b -keystore lunastore -
storetype luna
```

root.cer and **signing.p7b** are the CA Root Certificate and Signed Certificate Chain respectively.

6. Verify the keystore contents using the lunacm utility:

```
lunacm

C:\Program Files\SafeNet\LunaClient>lunacm.exe
lunacm.exe (64-bit) v7.3.0-165. Copyright (c) 2018 SafeNet. All rights reserved.

Available HSMs:

Slot Id ->              0
Label ->                Adobe
Serial Number ->        1238696044960
Model ->                LunaSA 7.4.0
Firmware Version ->     7.4.0
Configuration ->        Luna User Partition With SO (PW) Signing With Cloning Mode
Slot Description ->     Net Token Slot

Current Slot Id: 0

lunacm:> role login -n co

enter password: ********

Command Result : No Error

lunacm:>
The 'Crypto Officer' is currently logged in.  Looking for objects accessible to the 'Crypto
 Officer'.

lunacm:> par con
The 'Crypto Officer' is currently logged in. Looking for objects
 accessible to the 'Crypto Officer'.
```

```
Object list:

Label:         lunakey--cert0
Handle:        1100
Object Type:   Certificate
Object UID:    872512000500001730 1e0800

Label:         lunakey--cert1
Handle:        986
Object Type:   Certificate
Object UID:    882512000500001730 1e0800

Label:         rootca
Handle:        966
Object Type:   Certificate
Object UID:    9d2412000500001730 1e0800

Label:         lunakey
Handle:        295
Object Type:   Private Key
Object UID:    7c2312000500001730 1e0800

Number of objects:  4

Command Result : No Error

lunacm:>
C:\Program Files\SafeNet\LunaClient>
```

Create The Signed AIR application descriptor file:

1. Create an XML file named HelloWorld-app.xml and save it in the project directory.

```
<?xml version="1.0" encoding="UTF-8"?>

<application xmlns="http://ns.adobe.com/air/application/3.1">

<id>samples.flex.HelloWorld</id>

<versionNumber>0.1</versionNumber>

<filename>HelloWorld</filename>

<initialWindow>

<content>HelloWorld.swf</content>

<visible>true</visible>

<width>400</width>

<height>200</height>

</initialWindow>

</application>
```

2. Write the application code and save it to a file named HelloWorld.mxml.

```
<?xml version="1.0" encoding="utf-8"?>

<s:WindowedApplication xmlns:fx="http://ns.adobe.com/mxml/2009"

xmlns:s="library://ns.adobe.com/flex/spark"

xmlns:mx="library://ns.adobe.com/flex/mx"
```

```
title="Hello World">
<s:Label text="Hello AIR" horizontalCenter="0" verticalCenter="0"/>
</s:WindowedApplication>
```

3. Compile the application using the following command:

```
amxmlc HelloWorld.mxml
```

Running amxmlc produces HelloWorld.swf, which contains the compiled code of the application.

4. Test the Application using AIR Debug Launcher (ADL) to launch the application descriptor file.

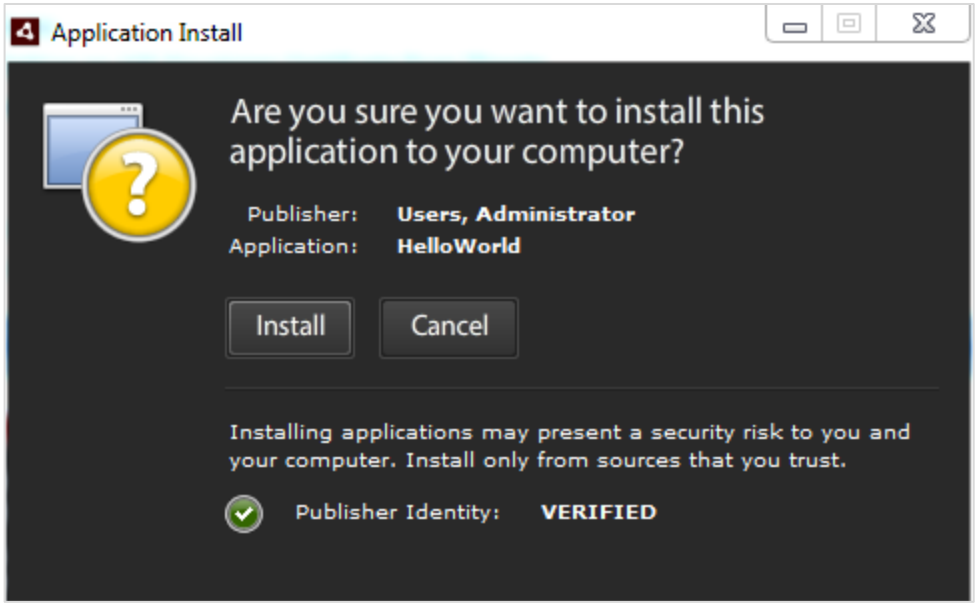
```
adl HelloWorld-app.xml
```

5. Create the AIR installation file using the below command:

```
adt -package -storetype luna -keystore lunastore -tsa none -target air HelloWorldNew.air
HelloWorld.xml HelloWorld.swf
password:
```

It will create a signer AIR Application.

6. Verify the signer AIR application:
   Double click and open the .air application, it will displays the publisher name during installation.



The .air file is signed and verified. The private key and signing certificate are securely stored on the SafeNet Luna HSM or HSM on Demand service. This completes the Adobe ADT Signing integration with SafeNet HSM.