Samsung KNOX

Samsung KNOX
TZ Client Certificate Manager (CCM)
PKCS11 usage information

SAMSUNG

Copyright Notice

Copyright © 2014 Samsung Electronics Co. Ltd. All rights reserved. Samsung is a registered trademark of Samsung Electronics Co. Ltd. Specifications and designs are subject to change without notice. Non-metric weights and measurements are approximate. All data were deemed correct at time of creation. Samsung is not liable for errors or omissions. All brand, product, service names and logos are trademarks and/or registered trademarks of their respective owners and are hereby recognized and acknowledged.

About this guide

This guide describes the PKCS11 standard API's supported by TZ CCM and how they can be exercised

1 Introduction

Trustzone (TZ) based Client certificate management (CCM) manages keys and certificates particularly used for client authentication purposes. This TZ application supports proprietary interfaces to

- install encrypted PKCS8 private key/certificate file
- request for TZ key pair generation followed by certificate signing request issuance
- capabilities to leverage default pre-installed client certificates, which are signed by the device root key

Once the keys are created within TZ, it is never seen in the normal world. One can only exercise these keys using the standard PKCS11 cryptographic token standard, which mandates password based logins for sensitive cryptographic operations. This document focuses on the latter half and briefly describes how the PKCS11 interfaces can be exercised.

What is the PKCS11 standard?

There are many portable devices out there like Smartcards, CAC cards, PCMCIA cards etc. which have the ability to store keys securely, under the control of a single user. With such devices, sensitive private keys never leave the device and cryptographic operations are performed on the card itself. PKCS11 standard is a standard programming interface to talk to such cards.

Where can I read more about PKCS11?

There are many links out there in the internet, but here are a few which can get you started quickly:

- http://www.cryptsoft.com/pkcs11doc/v220/
- http://www.nlnetlabs.nl/downloads/publications/hsm/hsm_node1.html
- The standard document:
    - ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf

How can I learn more about exercising PKCS11 as an application developer?

- JAVA interface
    - http://docs.oracle.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html
- OpenSSL engine API:
    - http://www.openssl.org/docs/crypto/engine.html

 Copyright © 2014 Samsung Electronics Co. Ltd.

2 TZ CCM PKCS11 specifications

At the native layer, the PKCS11 interfaces are exposed and implemented by a shared library by the name "**libtlc_tz_ccm.so**". Any PKCS11 cryptoki library has a static **CK_FUNCTION_LIST** structure, and a pointer to it may be obtained by the **C_GetFunctionList** function. This function in the TZ CCM PKCS11 library is named "**TZ_CCM_C_GetFunctionList**"

The table below summarizes the list of PKCS11 interfaces supported. The return values and API behaviors are compliant with the PKCS11 standard and the library expects the caller to use them in a standard way. We capture nuances of TZ CCM, if any, in the second column:

API	TZ CCM Notes
C_Initialize	*Synopsis*: Initializes the TZ CCM Trustzone application, if the Trusted boot measurements match Samsung authorized values
C_Finalize	*Synopsis*: Closes the TZ CCM application. Unloads the TZ Application, if there are not outstanding clients being serviced
C_OpenSession, C_CloseSession	*Synopsis*: Opens a session with CCM
C_Login, C_Logout	*Synopsis*: Login to a slot with a previously registered password (through the installation API's) Logout of the slot
C_GetSlotList	*Synopsis*: Obtain the list of slots
C_FindObjectsInit, C_FindObjects, C_FindObjectsFinal	*Synopsis*: Initialize, continue and finish an object search operation
C_SignInit, C_Sign	*Synopsis*: Initialize and perform signing operation
C_DecryptInit, C_Decrypt	*Synopsis*: Initialize and perform a decryption operation
C_EncryptInit, C_Encrypt	*Synopsis*: Initialize and perform a encryption operation
C_Digest	*Synopsis*: perform a digest operation
C_VerifyInit, C_Verify	*Synopsis*: Initialize and perform signature verification operation
C_GetInfo	*Synopsis*: General TZ CCM information
C_GetSessionInfo	*Synopsis*: PKCS11 session information
C_GetSlotInfo	*Synopsis*: PKCS11 slot information
C_GetTokenInfo	*Synopsis*: PKCS11 token information

The table below summarizes the list of UNSUPPORTED PKCS11 interfaces:

API
C_InitToken
C_InitPIN, C_SetPIN
C_UnwrapKey, C_WrapKey
C_SeedRandom, C_GenerateRandom
C_DeriveKey, C_GenerateKey, C_GenerateKeyPair
C_SetAttributeValue, C_SetOperationState
C_WaitForSlotEvent
C_CancelFunction
C_CloseAllSessions
C_GetMechanismList, C_GetMechanismInfo
C_GetOperationState
C_GetInfo, C_GetSessionInfo, C_GetSlotInfo, C_GetTokenInfo
C_CreateObject, C_CopyObject, C_DestroyObject, C_GetObjectSize

Also, none of the crypto “UPDATE/FINAL” operations are supported.

For Ex: **C_DigestUpdate, C_DigestFinal**