TPM Key Certification

Sarah L. Johnson

November 15, 2022

Relevant Background Definitions

Key Attributes

FixedTPM: non-duplicable
Restricted: operations are limited to TPM-generated data

Key Types

Primary: Created by TPM based on the current Primary Seed when executing the TPM2_CreatePrimary command. May be persisted within the TPM. Otherwise must be recreated after a TPM reset.

Ordinary: Created by TPM based on seed taken from the RNG when executing the TPM2_Create command. Must be the child of another key. May be persisted within the TPM or persisted external to the TPM in the form of an encrypted key blob. The blob is only loadable using the parent key's authorization.

Certificate

Type: X.509 digital certificate
Public key and data about the subject (i.e., identity) signed by certificate authority
$\mathrm{cert_K} := [(\mathrm{K}, \mathrm{ID})]_{\mathrm{CA}^{-1}}$
Validating a certificate requires a certificate chain in order to verify a chain of trust to a trust anchor (e.g., a root certificate)

Credential

Private key and certificate
$(\mathrm{K}^{-1}, \mathrm{cert_K})$

Secure Device Identifier

Definition

Identifier that is cryptographically bound to a device

Requirements

Attestation Key (AK): FixedTPM Restricted signing key
Device Identification Key (DevID): FixedTPM not-Restricted signing key

Initial Keys (IAK/IDevID)

Created by OEM at manufacturing time
Should be Primary keys
Recommended to be used only for enrollment of an LAK/LDevID
Typically the IAK certificate is the trust anchor for certificates created later

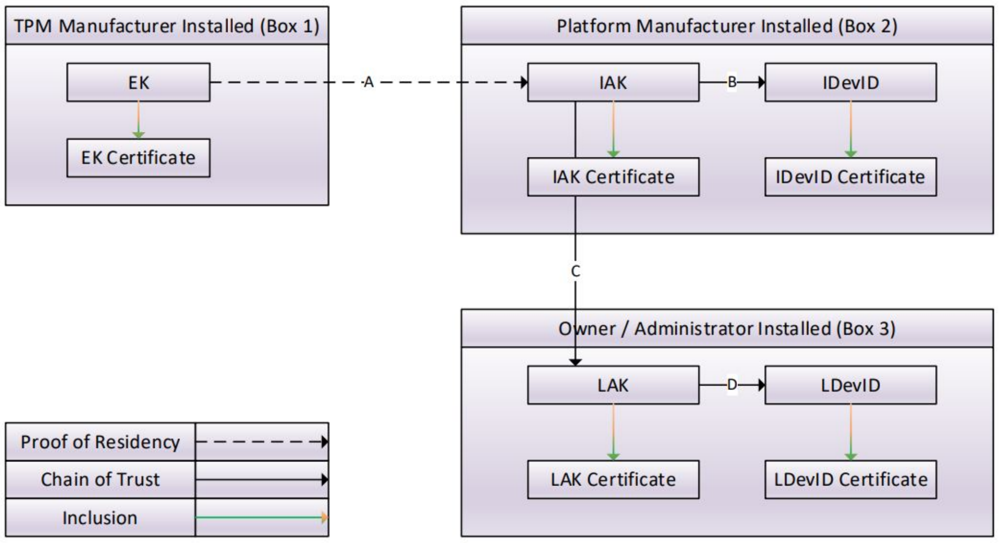
Local Keys (LAK/LDevID)

Created by owner
Should be Ordinary keys
Recommended to be used only for one application

Note regarding Endorsement Key (EK)

Cannot be used as a secure device identifier because it is a storage key (not a signing key) and identifies a TPM (not a device)

Relationships of Keys/Certificates



Box 1: The EK Certificate is signed by the TPM Manufacturer and binds the EK to a specific TPM from that manufacturer
Line A: The IAK is verified by the OEM's CA to have the correct key properties and to be resident in the same TPM as the EK
Line B: The IDevID is verified by the OEM's CA to have the correct key properties and to be resident in the same TPM as the IAK
Box 2: The IAK Certificate and IDevID Certificate is signed by the device OEM's CA and binds the IAK and IDevID to a specific device identity (e.g., model and serial numbers)
Line C: The LAK is verified by the Local CA to have the correct key properties and to be resident in the same TPM as the IAK
Line D: The LDevID is verified by the device owner's CA to have the correct key properties and to be resident in the same TPM as the LAK
Box 3: The LAK Certificate and LDevID Certificate is signed by the device owner's CA

extra

Prove a new key belongs to a specific device: 1) binding the TPM to the OEM device and 2) binding an AK to a TPM using the EK
An AK certificate is the trust anchor for certificates created later (because it is the certificate that ties a new key to the same TPM containing both keys)
The OEM CA makes an assertion by signing an IAK certificate that is a primary security dependency for later DevID certificate creation
An OEM-supplied IAK certificate is a definitive assertion to applications that need proof that the IAK belongs to a specific device

Certificate Authority

General Requirements

Must verify TPM residency of a key

Must evaluate TPM data in the certificate signing request

Should support a standard certificate tranport protocol that provides protection from replay attacks and provides confidentiality and integrity (e.g., Enrollment over Secure Transport (EST))

OEM Creation of IAK Certificate based on EK Certificate

Procedure assures that the new IAK is resident in the same TPM as the EK and that the EK resident in this TPM corresponds to the EK certificate

Detailed Procedure:

1. Platform creates the IAK
2. Platform builds the certificate signing request (i.e., the TCG-CSR-IDEVID structure)
   (a) Platform identity information (includes the device model and serial number)
   (b) The EK certificate
   (c) The IAK public area
3. Platform uses TPM2_Hash followed by TPM2_Sign on the CSR using the new IAK (proves control of the IAK to the CA)
4. Platform sends the signed CSR to the CA
5. The CA verifies the received data
   (a) Verify the signature on the CSR using the IAK public key (extracted from the CSR)
   (b) Verify the EK certificate using the TPM manufacturer's public key
   (c) Verify the attributes of the IAK
6. The CA uses TPM2_MakeCredential to create a credential blob
   (a) The cryptographic name of the IAK (hash of IAK public area)
   (b) Nonce
7. The CA encrypts the credential blob using the EK
8. The CA sends the encrypted credential blob to the Platform
9. The Platform uses TPM2_ActivateCredential command to release the nonce (proves that the IAK is loaded on the same TPM as the EK)
   (a) The TPM verifies the IAK's name using the EK
10. The Platform returns the nonce to the CA
11. The CA verifies the nonces match
12. The CA issues the IAK certificate

Symbolic representation of OEM Creation of IAK Certificate based on EK Certificate

1. TPM2_Create to get IAK, IAK^{-1}
2. CSR =
   (a) deviceInfo = (prodModel, prodSerial)
   (b) $\text{cert}_{\text{EK}} = [(\text{EK}, \text{tpmInfo})]_{\text{TPM\_CA}^{-1}}$
   (c) IAK
3. TPM2_Hash and TPM2_Sign to get $[\#\text{CSR}]_{\text{IAK}^{-1}}$
4. send $[\#\text{CSR}]_{\text{IAK}^{-1}}$
5. verify $[\#\text{CSR}]_{\text{IAK}^{-1}}$
   (a) CheckSig $[\#\text{CSR}]_{\text{IAK}^{-1}}$ with IAK
   (b) CheckSig cert_{EK} with TPM_CA
   (c) check attributes of IAK
6. TPM2_MakeCredential to get $\{\text{cred}_{\text{IAK}}\}_{\text{EK}}$ with cred_{IAK} =
   (a) #IAK (name)
   (b) r (nonce)
7. send $\{\text{cred}_{\text{IAK}}\}_{\text{EK}}$
8. TPM2_ActivateCredential to get r'
   (a) Decrypt $\{\text{cred}_{\text{IAK}}\}_{\text{EK}}$ with EK^{-1} and check #IAK
9. send r'
10. check r' = r
11. send $\text{cert}_{\text{IAK}} = [(\text{IAK}, \text{deviceInfo})]_{\text{OEM\_CA}^{-1}}$

Owner Creation of LAK Certificate based on IAK Certificate

Procedure assures that the new LAK is resident in the same TPM as the IAK

Detailed Procedure:

1. Platform creates the LAK
2. Platform uses TPM2_Certify command to certify the new LAK with the IAK, producing a signed TPM2B_Attest structure (proves that the new LAK is on the same TPM as the IAK)
3. Platform builds the certificate signing request (i.e., the TCG-CSR-LDEVID structure) including:
   (a) The IAK certificate (identifies device and binds TPM to the device)
   (b) The signed TPM2B_Attest structure (returned by TPM2_Certify on the LAK)
4. Platform uses TPM2_Hash followed by TPM2_Sign on the CSR using the new LAK (proves control of the LAK to the CA)
5. Platform sends the signed CSR to the CA
6. The CA verifies the received data
   (a) Verify the signature on the CSR using the LAK public key (extracted from the CSR)
   (b) Verify the IAK certificate using the OEM's public key

   (c) Verify the signature on the TPM2B_Attest structure using the IAK public key (extracted from the IAK certificate)
   (d) Verify the attributes of the LAK (must be FixedTPM Restricted signing key)

7. The CA issues the LAK certificate

Symbolic representation of Owner Creation of LAK Certificate based on IAK Certificate

1. TPM2_Create to get LAK, LAK^{-1}
2. TPM2_Certify to get $[\mathrm{attest}_{\mathrm{LAK}}]_{\mathrm{IAK}^{-1}}$
3. CSR =
   (a) $\mathrm{cert}_{\mathrm{IAK}} = [(\mathrm{IAK}, \mathrm{deviceInfo})]_{\mathrm{OEM\_CA}^{-1}}$
   (b) $[\mathrm{attest}_{\mathrm{LAK}}]_{\mathrm{IAK}^{-1}}$
   (c) LAK
4. TPM2_Hash and TPM2_Sign to get $[\#\mathrm{CSR}]_{\mathrm{LAK}^{-1}}$
5. send $[\#\mathrm{CSR}]_{\mathrm{LAK}^{-1}}$
6. verify $[\#\mathrm{CSR}]_{\mathrm{LAK}^{-1}}$
   (a) CheckSig $[\#\mathrm{CSR}]_{\mathrm{LAK}^{-1}}$ with LAK
   (b) CheckSig $\mathrm{cert}_{\mathrm{IAK}}$ with OEM_CA
   (c) CheckSig $[\mathrm{attest}_{\mathrm{LAK}}]_{\mathrm{IAK}^{-1}}$ with IAK
   (d) check attributes of LAK
7. send $\mathrm{cert}_{\mathrm{LAK}} = [(\mathrm{LAK}, \mathrm{deviceInfo})]_{\mathrm{Local\_CA}^{-1}}$