Securing FDO Credentials in the TPM

Review Draft, October 10, 2023

REVIEW DRAFT

This version:

https://fidoalliance.org/specs/FDO/securing-fdo-in-tpm-v1.0-rd-20231010/securing-fdo-in-tpm-v1.0-rd-20231010.html

Issue Tracking:

GitHub

Editors:

Geoffrey Cooper (Intel)

Andreas Fuchs (Infineon)

Version:

1.0

Abstract

This document is a specification for storing FIDO Device Onboard credentials in a TPM.

Table of Contents

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current FIDO Alliance publications and the latest revision of this technical report can be found in the FIDO Alliance specifications index at https://fidoalliance.org/specifications/.

This document was published by the FIDO Alliance as a Review Draft Specification.

This document is intended to become aFIDO Alliance Proposed Standard.

If you wish to make comments regarding this document, pleaseContact Us.

All comments are welcome.

This is a Review Draft Specification and is not intended to be a basis for any implementations as the Specification may change. Permission is hereby granted to use the Specification solely for the purpose of reviewing the Specification. No rights are granted to prepare derivative works of this Specification. Entities seeking permission to reproduce portions of this Specification for other uses must contact the FIDO Alliance to determine whether an appropriate license for such use is available.

Implementation of certain elements of this Specification may require licenses under third party intellectual property rights, including without limitation, patent rights. The FIDO Alliance, Inc. and its Members and any other contributors to the Specification are not, and shall not be held, responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

THIS FIDO ALLIANCE SPECIFICATION IS PROVIDED “AS IS” AND WITHOUT ANY WARRANTY OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY EXPRESS OR IMPLIED WARRANTY OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

1. SCOPE§

1.1. Key Words§

The keywords “MUST,” “MUST NOT,” “REQUIRED,” “SHALL,” “SHALL NOT,” “SHOULD,” “SHOULD NOT,” “RECOMMENDED,” “MAY,” and “OPTIONAL” in this document normative statements are to be interpreted as described in RFC-2119, Key words for use in RFCs to Indicate Requirement Levels.

1.2. Statement Type§

Please note a very important distinction between different sections of text throughout this document. There are two distinctive kinds of text: informative comment and normative statements. Because most of the text in this specification will be of the kind normative statements, the authors have informally defined it as the default and, as such, have specifically called out text of the kind informative comment. They have done this by flagging the beginning and end of each informative comment and highlighting its text in gray. This means that unless text is specifically marked as of the kind informative comment, it can be considered a kind of normative statements.

Start of informative comment

This is the first paragraph of 1–n paragraphs containing text of the kind *informative comment*...

This is the second paragraph of text of the kind*informative comment* ...

This is the nth paragraph of text of the kind*informative comment* ...

End of informative comment

2. TERMS AND DEFINITIONS§

Terminology from the FDO specification is used. In particular:

Device
: The Device being onboarded

Owner
: The Server onboarding the device

Device Manufacturer
: The entity that creates a board-level Device. This might also be the entity that initializes FDO credentials

Device Credentials
: Those FDO credentials that are stored in the Device and used during onboarding

Ownership Voucher
: A data structure that contains the Owner’s credentials, which are created when the Device Credentials are stored, and are passed through the supply chain until they reach the Owner

Device Certificate
: An X.509 certificate that identifies the device during onboarding

Device Key
: The private key associated with the public key in the Device Certificate. The Device proves ownership of the private key with digital signatures during the FDO TO1 and TO2 protocols.

FDO
: FIDO Device Onboard, an onboarding protocol from the FIDO Alliance.

ROE
: Restricted Operating Environment – A restricted operating environment in which FDO operates. The ROE might be a dedicated processor or a mode of operation. In this document, we also discuss creating an ROE during early system startup.

FDO Protocol
: Device Initialize (DI) protocol, Transfer Ownership (TO) protocols 0, 1, 2 (TO0, TO1, TO2)

FSIM
: FIDO ServiceInfo Module

FSIM protocols
: Sub protocols of the FDO TO2 Protocol that effect changes in the device for onboarding

TO0, TO1
: Transfer Ownership 1, Transfer Ownership 2 Protocols. These form the “Rendezvous” sub protocol within FDO, which allows the Device to determine the Owner’s IP address. Once this address is known, the Device runs the TO2 protocol.

TO2
: Transfer Ownership 2 protocol. The main onboarding protocol within FDO. The TO2 protocol performs authentication via mutual attestation, key exchange, creates a secure tunnel and does onboarding operations via FSIM’s. Although normally invoked after the Rendezvous protocol, the TO2 protocol contains all the security measures to stand alone.

3. INTRODUCTION§

3.1. Use Cases for FDO with TPM§

Start of informative comment

FIDO Device Onboard, or FDO, is a protocol for automatic onboarding of IoT and other devices. FDO is a Proposed Standard from the FIDO Alliance. This document refers to **FDO version 1.1** [FDO-Specification].

The TPM [TPM] provides basic cryptographic operations that are needed for FDO functions, such as key storage and asymmetric cryptography. The TPM is an implementation aid for FDO because it implements some of the required cryptographic mechanisms, such as: key storage, digital signatures, and HMAC/crypto-hash computation.

The keys used for FDO are naturally stored in the TPM and can be locked to the TPM for added security. The TPM also provides mechanisms for providing attested, trusted storage of keys and other data, sufficient to store other FDO credentials.

EPID keys are not covered in this specification.

The compatibility of a given TPM with a given set of legal FDO keys is not covered or guaranteed by this specification.

End of informative comment

When this standard is implemented, it becomes possible to:

- Store FDO device credentials in a TPM
- Restrict access to these credentials to meet the security model for running FDO
- Determine if a given TPM has FDO credentials stored in it, and if FDO is scheduled to be run
- Use the TPM credentials to determine if FDO is to be run
- Run FDO using TPM credentials
- Update the TPM to new credentials, or delete the old credentials, based on a successful FDO run
- Lock access to FDO credentials until the computer system is rebooted

3.2. FIDO Device Onboard§

Start of informative comment

FDO allows manufactured devices (IOT devices, servers, headless computers) to be installed without a login or local party for authorization. As such, FDO offers the promise of quick, reliable, and secure onboarding of devices on a large scale, suitable for a wide variety of IOT installations, including industrial devices, smart building installations, medical IOT, and smart retail applications.

In FDO, a manufacturer can initialize a device with FDO credentials and send it down the supply chain. At the same time, the manufacturer sends a digital object, the Ownership Voucher, along the paper-trail of the supply chain (e.g., along with ship confirm documents). The Ownership Voucher is self-securing and provides all the information needed for target server or cloud to identify itself to the device and onboard the device. In the case where all devices onboard to a single cloud-based server, the Ownership Voucher can be transmitted directly to that server, perhaps bypassing several supply chain steps.

When a device with FDO is ready to be installed, a technician physically places the device in its target location and connects it to a network that can reach the onboarding server, either in a local network or over the Internet. Then the technician powers on the device, completing his/her activity.

Depending on the network level protocol, a network-level onboarding protocol might be invoked to establish network-level connectivity. This network-level protocol is outside the scope of this document.

After network connectivity is established, the device autonomously uses FDO credentials to find the correct owner and securely onboard to the owner's server.

FDO identifies itself to its onboarding server using device attestation techniques over HTTP ("REST-type") protocols. The attestation is based on the Entity Attestation Token [EAT] and CBOR [RFC8949] standards from the IETF. The owner's server attests to the device using the Ownership Voucher. The device attestations use public key signatures that can be computed in a TPM, based on keys stored in the TPM.

Once the device/server attestation is complete, the FDO device and server set up an encrypted tunnel, over which additional "FSIM" protocols (aka. FDO ServiceInfo modules) are run to download keys, certificates or other credentials and content into the device. The flexibility of this layer permits devices with built-in package upgrade (e.g., Linux RPM) to update or add software as well. This allows additional flexibility to accommodate older devices, or devices with varying capabilities.

The FSIM mechanism permits multiple credentials to be provisioned, allowing for the common case where an IOT server uses different credentials for various device functions, such as data upload, device management, and device control. For example, a server might prefer a certificate authentication for device management, a token-based authentication for data upload, yet use SSH for remote device maintenance.

As a final step, FDO onboarding updates the device credentials and stores a new ownership voucher on the server. This prevents previous credentials from being exploited by an attacker who later penetrates the supply chain. It also allows FDO to be run again, if needed.

End of informative comment

3.3. FDO Credential Provisioning§

Start of informative comment

This specification permits FDO device credentials to be provisioned into a TPM at various points in the TPM or device lifecycle.

The device credentials are described in the FDO 1.1 specification in Section 3.4.1. The FDO specification separates the device key from the device credentials (i.e., it does not appear in Section 3.4.1). However, in the TPM, the device key for FDO is one of the credentials provisioned by the TPM. This key can be provisioned as defined in this specification or can be the LDevID or IDevID [TPMIdentKeys] stored in the TPM. See the DeviceKeyType value, defined in section 1.1.

It is also possible for FDO credentials to be added to the computer system after manufacturing. For example, an OEM might embed FDO software, and a VAR later adds FDO credentials that the software will use. Later, when the computer system is installed in its target application environment, FDO runs.

On the other extreme, a TPM vendor might embed FDO credentials in the TPM before the TPM is integrated into a computer system. FDO credentials are accessed, and when a computer system boot with all these conditions are met:

- A TPM is integrated into a completed computer system
- The computer system FDO software is installed
- Credentials for FDO are integrated into the TPM

In different situations, the order in which these conditions are met can vary.

This specification supports these use cases through features:

- Ability to preserve the Device Key (e.g., as IDevID) and reinitialize all other FDO parameters
- Ability to remove or reinitialize all FDO parameters
- Ability to update FDO parameters according to the FDO TO2 protocol specification

End of informative comment

For the purposes of this document, the term “FDO credentials” SHALL include all credentials in the FDO specification Section 3.4.1. The FDO device key is also included in the FDO specification. Instead of a “native” FDO device key, an alternative key (such as an IDevID or LDevID key following [TPMIdentKeys]) MAY be used. In those cases, the respective specifications SHALL apply instead.

To run FDO, a device SHALL be provisioned with FDO software before FDO is to be run. This software provisioning can take place when the device is first manufactured, or later in the supply chain.

After a device is provisioned with FDO credentials and software, the normal boot mechanism SHALL be adjusted so that FDO ROE software runs when the device is booted. The initial value of the Active variable SHALL be set to True, such that FDO onboards on the first boot after manufacturing.

In all situations for provisioning FDO credentials, a mechanism SHALL exist to sign the device ownership voucher to a key owned by its supply-chain recipient. The public key is signed, and the private key remains with the recipient. This recipient MAY be the recipient of the device itself or a later supply-chain entity that will take control of the device as the FDO owner, such as an application server or cloud server.

3.3.1. Device Manufacturer Provisioned FDO Credentials§

Start of informative comment

In this flow, a TPM is embedded in a computer system. The device manufacturer (computer system manufacturer) stores FDO credentials into the TPM as part of the manufacturing process. The device is provisioned with software that includes FDO. The device is shipped (via a supply chain) to a customer site, where the device is first powered up. Then FDO runs to onboard the device.

End of informative comment

The device manufacturer MAY provide a software or hardware mechanism to prevent the device from running FDO when it is first booted. This can be used in system recovery, remanufacture, or if the system OS needs to be modified. This mechanism is independent of FDO and of this specification.

3.3.2. TPM Vendor Provisioned FDO Credentials§

Start of informative comment

The TPM vendor can provision a TPM with FDO credentials while the TPM is being manufactured. In this case, the Ownership Voucher is also created at the TPM manufacturer. The TPM vendor endorses the Ownership Voucher to the device manufacturer; the device manufacturer re-endorses the Ownership Voucher with the device containing the TPM.

End of informative comment

When a TPM is manufactured to contain FDO device credentials the device manufacturer must ensure that each TPM device has a corresponding ownership voucher.

3.3.3. Supply-Chain Provisioned FDO Credentials§

Start of informative comment

A supply chain entity can provision a TPM with FDO credentials:

- By onboarding the device and updating the device credentials within the FDO TO2 protocol. This does not replace all the device credentials
- By installing device credentials into a TPM that does not have them, either because it never had FDO credentials or because it was reset.
- By replacing the device credentials in a TPM with new credentials

In all these cases, the supply chain entity obtains a corresponding Ownership Voucher, which it passes on to allow the device to be onboarded. If the device previously had FDO credentials, the previous Ownership Voucher is nullified by all the above operations and cannot be used to onboard any device. It is possible that other supply chain entities still have copies of this Ownership Voucher, but these copies are also nullified.

End of informative comment

When FDO credentials in a device are destroyed or replaced by other credentials, all available copies of the Ownership Voucher SHOULD be destroyed.

3.4. Device Startup with FDO§

Start of informative comment

FDO requires a set of FDO device credentials to be stored on a device. As mentioned above, we include the FDO device key in the TPM device credentials. When the device boots, the device operating environment (e.g., the OS start up scripting) discovers the FDO credentials and determines that onboarding is needed. In this case, the operating environment runs FDO onboarding before normal system bring-up, allowing an automated onboarding to happen. Afterwards, the credentials are updated to allow the system to boot into normal operation with the additional service information that has been supplied by FDO.

Some FDO credentials are protected with confidentiality, availability and integrity. FDO also requires other credentials to be stored with availability and integrity. In this specification, the TPM provides both these conditions for storage of credentials.

It is convenient to store all the FDO parameters in one place, so that FDO software can find all the parameters, and that all the parameters are protected with a similar mechanism. Since the FDO keys belong in the TPM, this suggests that other FDO parameters also be stored in the NVRAM of the TPM. There are also some advantages to storing all FDO parameters in the TPM, rather than just the keys:

- Storage of FDO parameters in a TPM makes it easy for system providers to guarantee the storage requirements needed by FDO.
- FDO implementations for different operating systems can use the same credentials stored within a TPM. This includes a mechanism for determining whether there are FDO credentials present, and whether the system needs to boot into FDO or normal operation.
- The FDO parameters can be installed before a final decision on the operating system is determined, then used in whichever operating system is running as the system is brought up for the first time.
- FDO parameters might be installed in a hardware TPM implementation before it is installed into a computer system.

This specification does not explain how to store some FDO credentials in the TPM and some outside the TPM, although this is permitted in the FDO specification.

End of informative comment

3.5. The FDO Restricted Operating Environment (ROE)§

Start of informative comment

The FDO 1.1 specification indicates that FDO be run in a Restricted Operating Environment (ROE). In this document, the ROE must have access to the FDO parameters within the TPM. Ideally, no other system code has access to the FDO parameters.

This document provides for various implementations of the ROE:

- If the processor supports a ROE mode directly, the TPM can be accessible only during this mode of operation
- A co-processor can act as a ROE for another processor.
- The operating system can support a ROE mode using OS protections, and protect the TPM's stored FDO parameters so that they are only accessible in this mode
- The operating system startup can implement a ROE mode where FDO parameters are accessible in the TPM, then the parameters are made inaccessible until the system reboots. Since FDO is normally run shortly after system boot, this mechanism can be used to protect FDO parameters from most programs that run on the operating system.

End of informative comment

3.6. Overview of FDO information elements§

Start of informative comment

FDO contains a set of information elements, called the device credentials. with different access privileges for the different FDO on device roles that need to be stored on the FDO device:

- **Active**: This flag states whether the FDO protocols shall be executed by the ROE. It is initialized to true during device or TPM manufacturing. It is readable by the ROE. The ROE can set it to false and the OS/App can set it to true / false (for execution of the FDO resale protocol).
- **ProtVer**: This field describes the supported FDO protocol version. It is initialized during device or TPM manufacturing. It can be read by the ROE and OS/App.
- **DeviceInfo**: This field describes the device. It is initialized during device or TPM manufacturing. This value can be also be used to identify the particular TPM device before it is associated with a computer system. For example, the DeviceInfo value can correspond to a bar code on the TPM part manifest, box or container that can be scanned during computer system manufacturing. Then this code can be replicated on the box of the computer system containing this TPM. The DeviceInfo can also contain information that identifies the TPM or the device to the FDO Owner. If part of all of the DeviceInfo is intended to describe the device itself, but the TPM is initialized before the device is known, the string “TPM” should be used for this purpose.

  The value of **devmod:device** can be used to describe the device during FDO ServiceInfo.
- **GUID**: This describes a unique id for the device for this instance of the FDO credentials. It can be read and written by the ROE and cannot be accessed by the OS/App.
- **RVInfo**: This describes the information on the Rendezvous server. It can be read and written by the ROE and cannot be accessed by the OS/App.
- **PubKeyHash**: This references the public key of the initial ownership voucher’s signature. It is initialized during device or TPM manufacturing. It can be read and written by the ROE and cannot be accessed by the OS/App.
- **HMAC Secret**: This is used to calculate the HMAC of the FDO information. The resulting HMAC is used in the ownership voucher. It is created during device or TPM manufacturing. The ROE can use it for HMAC calculation and can request the TPM to change to a new HMAC Secret. The HMAC Secret is shielded and cannot be read or written by any role.
- **Device Key**: This is used to identify the FDO device. It is created during device or TPM manufacturing. The ROE and OS/APP can read the public key and use it for signing. The private key cannot be read, written, altered or recreated by any role. A key provisioned in accord with another specification can be used, especially that the IDevID or an LDevID be used for FDO. This specification only applies to device keys that are present in the TPM.

See also FDO 1.1 specification, section 3.4.1.

End of informative comment

3.7. Mapping of FDO roles to TPM-enabled devices§

Start of informative comment

The FDO specifications defines the following components and roles to be implemented on the device:

- Device ROE: A Restricted Operating Environment; See section 4.5
- Device ROE App: The application that is installed in the ROE; for FDO this is the FIDO Device Onboard Device software
- Management Agent: The entity on the device that uses the FIDO Device Onboard Device software
- Device OS/App (not part of FDO description): The regular OS and Application running the Device's functional software.

For a TPM-enabled device, this specification will support two different kinds of mapping these components.

Devices with a logical separation between ROE and Application:

- TPM: The secure storage and cryptographic operations with authorizations parts of the Device ROE / ROE App.
- Device ROE App: A process running on the device that is logically separated from the main Device OS/App. The ROE App requires a secure storage for authentication credentials (against the TPM) that are not accessible from the Device OS/App.
- The Device OS/App original functional application.

> NOTE: This separation might be implemented using hardware separation, virtualization, containers or user and process isolation features of an underlying hypervisor or kernel.

For purposes of FDO, a time-based separation of ROE and Application mode is possible. This is because FDO only needs to run during early system operation, and then does not need to run again until the Device is rebooted. The TPM provides secure storage for FDO parameters and helps the FDO implementation with cryptographic operations. Examples include:

- The BIOS or other early-boot firmware acts as ROE to run FDO
- ROE runs during system pre-boot (e.g., 'initrd') and never runs co-resident with the OS
- ROE runs during system boot, as an early process in the OS initialization (e.g., /etc/rc or systemd). In this case, the ROE runs as soon as other required subsystems are initialized, such as the networking services required by FDO.

  In such Devices, FDO software terminates the ROE "phase of operation" on every system boot, whether or not FDO has run. This requires that some FDO support code always be part of the operating system boot procedure. For example, in Linux systems, an FDO script can run in "systemd".

End of informative comment

The separation between the ROE and the main application is a responsibility of the overall system. There are two ways to implement this:

- **Logical Separation**:

  The system SHALL provide authentication secrets only to the ROE and hide them from the rest of the system as well as external parties.

- **Time-based separation**:

  On every system start-up, the system SHALL start the ROE before the rest of the system and the ROE SHALL set the WriteLocks and ReadLocks on the respective NV indices when handing over control to later boot stages. System software enabled before the setting of WriteLocks and ReadLocks is considered part of

the ROE. Note that some of these ROE components MAY continue to be used by later boot stages (e.g. network stack).

3.8. Discontinuing FDO usage on devices§

Start of informative comment

It is possible that a device is provisioned with FDO credentials, and a subsequent decision is made not to use FDO. For example, the device can be manufactured to support FDO, then re-imaged with an operating system that does not support FDO. The “left over” FDO credentials represent a vulnerability to the device, as follows:

- The ownership voucher for the device, that is cryptographically linked to FDO parameters in the device TPM, might not be protected well, since it is not intended to be used. It is more likely to come into the possession of a potential attacker
- The OS of the device is likely to be upgraded over time. Eventually, a version of the OS might support FDO in TPM. This will “re-activate” any lingering credentials and the ownership voucher that corresponds to them
- FDO implementations are designed to give the FDO Owner (server) supervisory or root access to the device. An attacker who has an ownership voucher for the device can take over arbitrary control of the device when it reboots with FDO active.

Therefore, it is a best security practice to destroy the Device-resident FDO credentials in this case. This also applies to the Ownership Voucher, but destroying it is impractical if it has been previously distributed in the supply chain. So long as the device credentials are destroyed, no damage can be done using a left-over Ownership Voucher.

The exception is if FDO credentials are planned to be used, but the operating system support is not yet present. This can occur when a standard operating system image is applied to the device.

End of informative comment

If a decision is made to install software that will never support an FDO ROE on the device, then the FDO credentials SHALL be destroyed from the TPM by deleting or overwriting the NV entries described in section § 4.2 Handles for FDO Credentials. The corresponding Ownership Voucher SHOULD be destroyed.

4. Storage of FDO Credentials in the TPM§

The FDO Public Device Credentials (§ 4.1 FDO Device Credentials Overview) are stored as data in an NV Index in the TPM. The FDO Secret Device Credentials (§ 4.1 FDO Device Credentials Overview) are created and stored as persistent Objects in the TPM.

The FDO Credentials MAY be pre-provisioned by the TPM manufacturer or MAY be provisioned by the Device manufacturer using standard TPM commands. These credentials MUST be accessed only by the ROE, except for the Active flag.

To enable the FDO software to locate the FDO Credentials in the TPM, the NV Index and persistent Object handles defined in section § 4.2 Handles for FDO Credentials SHALL be used.

The following is vendor-specific:

- The FDO Device certificate content.
- Whether the FDO Device key is a IDevID or LDevID
- Whether the device certificate is stored in the TPM (The FDO protocol does not require the certificate to be stored in the FDO Device at all)

- Optional: a copy of the device Ownership Voucher.

4.1. FDO Device Credentials Overview§

The FDO device credentials are defined in the FDO 1.1 specification[FDO-Specification] as the DeviceCredential structure, expressed in the CBOR [RFC8949] encoding using CDDL [RFC8610]. Consistent with the letter and spirit of the FDO specification, this information is split differently in the TPM:

- DCActive.*: The DeviceCredential.DCActive flag is stored in a separate TPM location. This value MAY be accessed or written during normal OS operation.
- DCKey.*: The DeviceCredential.DCHmacSecret is stored in a separate TPM location. The Device Key is also stored in this location, unless it is the IDevID or an LDevID key.
- DCTPM.*: The rest of “DeviceCredentials” is stored in the CBOR format “DCTPM” in a single TPM location (see below)
- Device Key certificate: (Aka Device Certificate). This is an X.509 certificate that contains the public key of the Device private key. The Device Key certificate is part of the Ownership Voucher.
- DCOV.*: The Ownership Voucher, if stored in the TPM, is stored in a dedicated location, in CBOR format.

The CBOR structure of DCTPM is defined in section 5.5 using CDDL.

DCTPM.* section contains two variable length items (RVInfo and DeviceInfo). The item’s length can change during the FDO TO2 protocol. This specification recommends a reasonable limit for these two items. An implementation SHOULD decide to add additional space to allow more flexibility for the TO2 protocol by increasing the size of these items unless extra TPM storage is unavailable.

When initializing the DCTPM.* NV value, 67 bytes are fixed in size; the DeviceInfo and RVInfo fields can vary. The NV index’s size SHALL be at least 384 bytes (leaving 317 bytes variable for DeviceInfo and RVInfo). This leaves modest expansion room for the variable fields. A size of 512 bytes is recommended (leaving 445 bytes variable). The content SHALL be encoded in CBOR.

The following FDO parameters require availability and integrity.

DCActive.*	**Binary Value**	**Meaning**
Active	1 byte (1=true,0=false)	Determines whether FDO should be attempted at the next boot of the computer. Modified by FDO and/or OS-resident software See § 4.4 FDO Active flag
DCTPM.*	**CBOR Type**	**Meaning**
[ ]	CBOR array header	DCTPM is a CBOR array, the first datum is a CBOR array header. See CBOR definition of DCTPM: § 4.5 FDO Public Device Credentials. Some fields in DCTPM change over time, but DCTPM is usually read/written as a unit, due to the variable encoding of CBOR
ProtVer	CBOR uint16	Encoding of FDO version (e.g., Version 1.1 is 101)
DeviceInfo	tstr	String. Identifies device type/model to server implementation, allows (Owner) server to select onboarding procedure or script. Changes after each onboard. Leave space for expansion.

Guid	bstr	Random bit string used to identify the onboarding instance; Changes after each onboard. Leave space for expansion.
RVInfo	CBOR	CBOR-encoded instructions to find the Rendezvous Server, defined in FDO specification. Changes after each onboard. Leave space for expansion.
PubKeyHash	256- or 384-bit Hash	Hash of public key in the Ownership Voucher. For a new device, this is a public key from the device manufacturer.
DeviceKeyType	Enum	0: FDO key (device key is derived from Unique String; see§ 4.6 FDO Device Key) 1: The IDevID in the TPM 2: An LDevID in the TPM
DeviceKeyHandle	TPM_Handle	MUST be the handle of the device key. This SHOULD be the handle of one of these: • The FDO key (DeviceKey, below) • The IDevID in the TPM • An LDevID in the TPM

Table 1 **– FDO Public Device Credentials**

The following FDO parameters require confidentiality, availability and integrity.

The FDO device key is a key in this TPM, either the IDevID, an LDevID, or the DeviceKey (specified below). The FDO device key MAY be used to authenticate an FDO TLS connection, provided the key does not leave the ROE. An IDevID MAY be used for both FDO and IDevID specific functions. Otherwise the FDO device key SHOULD be used only for FDO.

The HMAC Secret (HmacSecret) is reinitialized every time FDO TO2 is run to completion (i.e., onboarding succeeds).

For each of FDO device key and HMAC Secret, a Unique String determines the value of the item. If an existing value is to be preserved, the same Unique String MAY be used; if a different value is needed, a new Unique String MAY be chosen at random. For purposes of typographic spacing, the abbreviation **"U/S"** is used for Unique String in this document.

No OS access is provided for these items.

DCKey.*	**Type**	**Size (bits)**	**ROE Access only**	**U/S NV Index**	**Meaning**
HmacSecret	Key	256 384	HMAC, Recreate	3	HMAC secret
DeviceKey	Key	256 384	Sign, Recreate	4	ECDSA NIST P-256 / P-384 key. Handle(s) MAY be created only when DCTPM.DeviceKeyHandle references this DeviceKey.

Table 2 **– FDO Secret Device Credentials**

DeviceKey is defined as ECDSA NIST P-256, P-384 or EPID in the FDO specification. FDO device attestation is based on the Entity Attestation Token, so additional crypto modes in this specification might also apply to FDO. Other key mechanisms defined in the TPM are likely to be operable with the FDO specification, but will not interoperate with other FDO implementations.

The device certificate corresponding to the device key is not required to be stored on the FDO device. It resides in the Ownership Voucher. However, the device certificate can also be stored in the TPM. This permits the Device Key to be used with TLS. This might also be required if the Device Key is the IDevID.

If the TPM part is initialized with FDO before it is placed into a computer system, then the DeviceInfo will not reference the device, only the TPM. This requires the FDO Owner (server) code to determine the device type after the TO2 connection is initiated. The existing FSIM field: **devmod:device** can be used for this purpose, and the device is already mandated to send this field. See FDO 1.1 Specification, section 3.8.2.

DCOV.*	Type	ROE access	OS access	Meaning
DeviceOV (optional)	Ownership Voucher	Yes	Yes	An optional copy of the Ownership Voucher that corresponds to the credentials stored in Table 1 & Table 2.

***Table 3* – Ownership Voucher storage in TPM (DCOV.*)**

The DCOV.* field is provided as a convenience for a TPM that is programmed in the factory, before it is inserted into a computer system. When the device is merged with a computer system, the ownership voucher (OV) has to be associated with the device. This can be accomplished by sending the OV separately, and then correlating the correct OV with the chip that has the associated DeviceCredentials during device assembly. Alternately, the Ownership Voucher can be computed at the same time as the DeviceCredentials (e.g., at the TPM manufacturer), then inserted into this field. Then, during device manufacture, the OV can be read from the TPM and associated with the device in the factory's databases. The OV is read using a factory program that is neither the ROE nor the ultimate OS.

Note that the Ownership Voucher MUST be extended to a key owned by the factory to be useful in this case or to a key pair that is stored appended to the DeviceOV in COSE key format

Once the DeviceOV field is read, it MUST be deleted from the TPM.

4.2. Handles for FDO Credentials§

This section describes the handles that are standardized by TCG for storing FDO Credentials in the TPM.

Table 4 specifies:

- Persistent object handles for the FDO Device key
- Persistent object handles for the FDO HMAC secret

Table 5 specifies:

- NV Index handles for the FDO Public Device Credentials
- NV Index handles for the (optional) FDO Ownership Voucher (DCOV.*)
- NV Index handles for the (optional) FDO Device certificate

Note that the ownership voucher permits only one choice of Device Key and HMAC secret. Only one of these parameters can be populated in a given TPM device.

NOTE: TODO: The TPM handles in Table 4 and Table 5 are not yet allocated by TCG. **The values presented are appropriate for testing, but not for released products.** Current expectation from TCG is that FIDO will be delegated to allocate the range 0x01d10000-0x01d100ff. FIDO has yet to develop governance around this allocation or to determine how to record or disclose these decisions.

FDO Parameter	Index handle	Description
FDO Device keys		
FDO Device key	~~0x81020002~~	Persistent (Primary) Object in the Endorsement hierarchy. Contains ECC NIST P-256 / P-384 key. User program SHALL query object size to determine which size key is present. If `DCTPM.DeviceKeyType` indicates, this value is unused and an IDevID or LDevID key is used as the FDO key.
FDO HMAC secrets		
FDO HMAC	~~0x81020003~~	Persistent (Ordinary) Object in the Endorsement hierarchy. Contains HMAC secret (SHA-256 / SHA-384). User program SHALL query object size to determine which size HMAC is present.

***Table 4* – Persistent object handles for FDO Credentials**

✧ ✧ ✧

FDO Parameter	Index handle (Hex)	Description
FDO NV Handles	~~0x01D10000-0x01D10005~~ (inclusive)	Reserved range for FDO on TPM
DCActive.*	~~0x01D10000~~	NV Index for OS action flag as byte value (1=true,0=false) See § 4.4 FDO Active flag
DCTPM.*	~~0x01D10001~~	NV Index for DCProtVer, DCDevinceInfo, DCGuid, DCPubKeyHash, DCRVInfo in CBOR encoding. See § 4.5 FDO Public Device Credentials
DCOV.*	~~0x01D10002~~	NV Index for DCOV in CBOR encoding. See § 5 FDO Ownership Voucher
HMAC Secret U/S	~~0x01D10003~~	Unique String to generate HMAC secret based on SHA-256 or SHA-384. Application uses length of the index to determine SHA size
Device Key U/S	~~0x01D10004~~	Unique String to generate Device Key, ECC NIST P-256 / P-384. The X and Y coordinates of the unique string SHALL be concatenated and stored in the NV index, so that the

		applicaton can use the length of the index to determine the size of the unique string.
FDO Certificate	~~0x01D10005~~	(Optional) X.509 Certificate for FDO Device key

***Table 5* – NV Index handles for FDO Credentials**

Note that different handle values are used for different key sizes, the TPM SHALL be provisioned with only one key size for each FDO parameter. The Device key and its corresponding Device certificate are stored at the same offset within their respective handle range.

4.3. Authorization for FDO Credentials§

Start of informative comment

The authorization requirement for FDO keys are:

- During ROE: no restriction on usage

  FIDO Device Onboard is performed at the beginning of a device's lifetime, when the operating system of a new device starts for the first time. At this point, the device has only generic content and functions (these are installed only at the end of the onboarding process). The system of the device trusts the TPM on first use. As a consequence, no authorization or a simple authorization (to prevent mistakes) is used.
- During device operation (in the field):

  The OS can extract and delete the Ownership Voucher as well as change the Active flag.

  Furthermore, if an IDevID or LDevID is used for the FDO Device Identity key, the access rights of these keys persist. If an FDO Device Identity key is used then it is not accessible from the OS.

The policies and template used to enforce this behavior are defined in Section 7.

FDO keys	FDO Device key	FDO HMAC key
Store	Primary object in Endorsement hierarchy (optionally persistent)	Primary object in Endorsement hierarchy (optionally persistent)
Create	TPM2_CreatePrimary w/ Endorsement authorization using DeviceKey Unique String NV Index	TPM2_CreatePrimary w/ Endorsement authorization using HMAC Unique String NV Index
Persist / Unpersist (optional)	TPM2_EvictControl w/ Owner authorization	TPM2_EvictControl w/ Owner authorization
Use	TPM2_Sign w/ Object *authPolicy*	TPM2_HMAC w/ Object *authPolicy*
Change	TPM2_NV_Write of the Unique String NV Index	TPM2_NV_Write of the Unique String NV Index
Delete	TPM2_Clear w/ Platform or Lockout authorization TPM2_UndefinSpace of the Unique String NV Index (Same key can be recreated as long as EPS does not change and same Nonce NV	TPM2_Clear w/ Platform or Lockout authorization TPM2_UndefinSpace of the Unique String NV Index (Same key can be recreated as long as EPS does not change and same Nonce NV

	does not change and same Nonce NV Index is populated)		does not change and same Nonce NV Index is populated)

***Table 6* – Authorization for FDO keys (created with default Template)**

FDO data	Unique String for DCKey.*	DCActive.*	DCTPM.*
Store	Populated as NV Index	Populated as NV Index	Populated as NV Index
Create	TPM2_NV_DefineSpace w/ Platform or Owner authorization	TPM2_NV_DefineSpace w/ Platform or Owner authorization	TPM2_NV_DefineSpace w/ Platform or Owner authorization
Read	TPM2_NV_Read w/ Authorization NV Index *authValue* Before TPM2_NV_ReadLock	TPM2_NV_Read w/o Authorization (empty) NV Index *authValue*	TPM2_NV_Read w/ Authorization NV Index *authValue* Before TPM2_NV_ReadLock
Write	TPM2_NV_Write w/ Authorization NV Index *authValue* TPM2_NV_WriteLock	TPM2_NV_Write w/o Authorization (empty) NV Index *authValue*	TPM2_NV_Write w/ Authorization NV Index *authValue* TPM2_NV_WriteLock
Delete	TPM2_UndefineSpace w/ Platform or Owner authorization	TPM2_UndefineSpace w/ Platform or Owner authorization	TPM2_UndefineSpace w/ Platform or Owner authorization

FDO data	Unique string for DCKey.*	DCOV.*	FDO Device cert
Store	Populated as NV Index	Populated as NV Index	Populated as NV Index
Create	TPM2_NV_DefineSpace w/ Platform or Owner authorization	TPM2_NV_DefineSpace w/ Owner authorization	TPM2_NV_DefineSpace w/ Platform or Owner authorization
Read	TPM2_NV_Read w/ Authorization NV Index *authValue* Before TPM2_NV_ReadLock	TPM2_NV_Read w/ Owner authorization	TPM2_NV_Read w/ Authorization NV Index *authValue* Before TPM2_NV_ReadLock
Write	TPM2_NV_Write w/ Authorization NV Index *authValue* TPM2_NV_WriteLock	TPM2_NV_Write w/ Owner authorization	TPM2_NV_Write w/ Authorization NV Index *authValue* TPM2_NV_WriteLock
Delete	TPM2_UndefineSpace w/ Platform or Owner authorization	TPM2_UndefineSpace w/ Owner authorization	TPM2_UndefineSpace w/ Platform or Owner authorization

***Table 7* – Authorization for FDO data**

End of informative comment

4.4. FDO Active flag§

The FDO Active bit SHALL be stored as an NV Index in the TPM as a 1 byte Boolean with the values 0x01 (true) and 0x00 (false).

4.5. FDO Public Device Credentials§

The FDO Public Device Credentials (defined in § 4.1 FDO Device Credentials Overview) are stored as an NV Index in the TPM. The NV Index data field SHALL contain the DCTPM structure as is specified in CBOR DCCL below. The CDDL types protver, Guid, RendezvousInfo, and Hash are as defined in the FDO 1.1 specification [FDO-Specification].

```
DCTPM = [
    DCProtVer:    protver,
    DCDeviceInfo: tstr,
    DCGuid:       Guid
    DCRVInfo:     RendezvousInfo,
    DCPubKeyHash: Hash
    DeviceKeyType: uint
    DeviceKeyHandle: uint
]
```

4.6. FDO Device Key§

Start of informative comment

As stated above, the IDevID or an LDevID key can be used as the Device key. The creation and maintainance of such a key is outside the scope of this document.

This specification also defines a Device key, stored in the TPM, used only for FDO. This section defines such a Device key.

The Device key is created and stored in the TPM as Object.

Lifetime:

Depending on the device's intended use or application, there are two potential policies for the lifetime of the Device Key:

Case 1: The Device key serves as permanent identity of the device (following IEEE 802.1AR[802.1AR-2018]), in this case, the Device key must not change. The reader can consider if it is more appropriate to use the IDevID as the Device key in this case.

Case 2: In cases where Privacy is of concern, the Device key must be replaced (and a new Device certificate must be issued) when the device is reinitialized. However, it might be necessary to persist the Device key across several invocations of FDO to completely initialize the device.

TPM key type:

The Device key is created as Primary Key in the Endorsement hierarchy. This ensures that the Device key can be recreated as long as the Endorsement Primary Unique String (EPS) does not change.

To allow the Device key to be replaced during ownership transfer (without changing the EPS), the creation of the Device key involves a nonce (random value), which is stored in an NV Index. The nonce is provided to the TPM during Device key creation in the *inSensitive.data* field of the TPM2_CreatePrimary command.

During device reinitialization, if the device key should be replaced, the nonce in the NV Index is updated with a

new random value (this new nonce is then used in the creation of the new device key). This ensures that a new device key is chosen.

Note: Primary keys are derived in a deterministic way from a Primary Unique String, which allows the same Primary key to be recreated as long as the Primary Unique String does not change and the same Template and unique string is used.

Note: The EPS does typically not change during the lifetime of the TPM since this would invalidate the EK certificate issued by the TPM manufacturer.

Storage:

The Device key can be stored persistently in the TPM or recreated when needed.

- Option 1: Persist Device key

  The Device key is persisted in the TPM during provisioning by the (TPM or device) manufacturer (for the persistent object handle, refer to § 4.2 Handles for FDO Credentials).

  Note: If the persistent Device key is deleted using TPM2_EvictControl or TPM2_Clear with Owner authorization, it can still be recreated as long as the EPS does not change.

- Option 2: Recreate Device key

  The Device key is not persisted in the TPM, but recreated with TPM2_CreatePrimary (after a TPM Reset) when needed using Endorsement authorization and the same Template.

  Note: The Endorsement authorization is initialized to Empty Buffer when the TPM is in factory state or cleared. That means, when FIDO Device Onboard is performed, the Device key might be created with a NULL Password. During later operations of the device, the Privacy Administrator might choose to populate the Endorsement authorization. Thus, the ROE would not be able to recreate the Device key unless it knows the Endorsement authorization or the TPM is cleared.

End of informative comment

When an LDevID or IDevID key is **not** used as the Device key:

1. Device key SHALL be created as Primary key in the Endorsement hierarchy.
2. The Device key SHALL be created using one of the default Templates for the FDO Device key (defined in § 4 Storage of FDO Credentials in the TPM).
3. The Device key creation SHALL include a unique string, which is provided to the TPM in the *inSensitive.data* field of the TPM2_CreatePrimary command
   - The unique string used for key creation SHALL be populated in an NV Index
4. The Device key SHOULD be persisted within the TPM (see option 1 above).
   - If persisted, the Device key SHALL be stored using the persistent object handle for the FDO Device key (defined in § 4.2 Handles for FDO Credentials).
5. The size and structure of the unique string SHALL be as follows:

   For the ECC NIST P-256 key, the unique string SHALL be 64 bytes. For the ECC NIST P-384 key, the unique string SHALL be 96 bytes. The first half (32 or 48 bytes, respectively) SHALL contain the X coordinate and the second half SHALL contain the Y coordinate. These coordinates SHALL be stored in big endian format.

Note: Default Templates for the FDO Device key are defined for ECC NIST P-256 and NIST P-384

4.7. FDO HMAC Secret§

Start of informative comment

The HMAC secret, which is part of the FDO Secret Device Credentials (defined in), is created in the TPM as a Primary Object based on a Unique String stored in NV. Optionally, it can be stored persistently.

Lifetime:

The HMAC secret is changed when the FDO Public Device Credentials change.

The same HMAC secret can still be used after a TPM2_Clear was issued (which removes all persistent as well as RNG-created keys) such as to preserve the FDO credentials over a resale when user data is usually deleted.

Note: When the FDO Public Device Credentials are updated during the TO2 protocol, a new HMAC value is computed over the credentials for the new Ownership Voucher.

Storage:

The HMAC secret's unique string is stored in an NV index under the Platform hierarchy. The HMAC secret itself can be stored persistently in the TPM for improved performance or to remain usable in case the Endorsement Hierarchy password is set.

Note: Primary keys are derived in a deterministic way from a Primary Unique String, which allows the same Primary key to be recreated as long as the Primary Unique String does not change and the same Template and unique string is used.

End of informative comment

1. The HMAC secret SHALL be created as Primary key under the Endorsement hierarchy using a unique string from an NV index.
2. The HMAC secret SHOULD be created using one of the default Templates for the FDO HMAC secret, see below (§ 8.2 FDO Device key & HMAC key Templates).
3. The HMAC secret MAY be persisted within the TPM.
    1. If persisted, the HMAC secret SHALL be stored using the persistent object handle for FDO HMAC secret (§ 4.2 Handles for FDO Credentials).
    2. If persisted, the HMAC secret SHALL be evicted during TO2.
4. The HMAC secret unique string SHALL be recreated during TO2.
5. The size and structure of the unique string SHALL be as follows:

    For a HMAC-SHA256, the unique string SHALL contain 32 bytes. For a HMAC-SHA384, the unique string SHALL contain 48 bytes.

Note: Default Templates for the FDO HMAC secret are defined for HMAC-SHA256 and HMAC-SHA384.

4.8. FDO Device certificate§

The certificate chain for the Device key is present in the Ownership Voucher. Optionally, the device certificate or certificate chain MAY also be stored in the TPM as NV Index; see Table 5.

If the Device Key is the IDevID or the LDevID, the device certificate is stored as described in the TCG DevID specification [TPMIdentKeys].

5. FDO Ownership Voucher§

When the TPM manufacturer wishes to use the TPM chip itself to deliver the Ownership Voucher to the Device Manufacturer, the TPM manufacturer's Ownership Voucher MAY be stored in the TPM as NV Index; see Table 5

This facility is provided for situations where the TPM manufacturer desires to place FDO credentials in the TPM, to avoid the need for an out of band transfer of the Ownership Voucher to the Device manufacturer. The Ownership Voucher MUST be pre-signed to a public key owned by the Device Manufacturer.

Since the Ownership Voucher is normally stored outside the Device, the Device SHALL delete the Ownership Voucher after it has been extracted.

Upon storing the Ownership Voucher in the TPM, the DC.Active flag SHALL be set to False. The Device FDO software SHALL extract the Ownership Voucher before setting the DC.Active flag to be True.

Start of informative comment

This mechanism only applies to situations where the TPM manufacturer can target specific TPM chips to a specific Device manufacturer (e.g., it might require a large batch of TPM's).

The Ownership Voucher is protected by a signature of the TPM manufacturer, so this mechanism is equivalent to sending the Ownership Voucher out of band, from a security point of view.

Were the DC.Active flag set to True before the Ownership Voucher is extracted, the Device would hang in FDO waiting for an Owner that will never arrive.

End of informative comment

5.1. TPM Operations on FDO Credentials§

5.1.1. Locating FDO Credentials in TPM§

To determine whether any FDO Credentials are stored in the TPM, the ROE shall query the TPM for the presence of the handles defined in this specification.

Start of informative comment

The following commands can be used:

- TPM2_GetCapability (capability = TPM_CAP_HANDLES, property = TPM_HT_NV_INDEX) returns a list of NV Index handles present in the TPM.
- TPM2_GetCapability (capability = TPM_CAP_HANDLES, property = TPM_HT_PERSISTENT) returns a list of persistent object handles present in the TPM.

End of informative comment

If the TPM has been provisioned with FDO Credentials, the TPM SHALL at least store

- one Public Device Credentials NV Index
- one HMAC Unique String NV Index
- If the public device credentials indicate the key type is zero (0) (FDO Key), then:
    - one Device Key Unique String NV Index

5.1.2. Deleting FDO Credentials in TPM§

To invalidate the FDO credentials in the TPM, the following credentials SHALL be deleted:

- one Public Device Credentials NV Index
- one HMAC Unique String NV Index
- If the public device credentials indicate the key type is zero (0) (FDO Key), then:
  - one Device Key Unique String NV Index
- the optional HMAC persistent object, if present
- the optional Device Key persistent object, if present
- Device certificate NV Index (using TPM2_UndefineSpace)

MAY be deleted

- LDevID used for the Device Key.

IDevID used for Device Key is managed according to the rules for IDevID in[TPMIdentKeys]

6. DEVICE INITIALIZATION AND STARTUP§

6.1. Device Initialization for FDO§

When the TPM is initialized for FDO, the FDO credentials are stored in the TPM, as specified in**Error! Reference source not found.**.

Typically, FDO is run during the next boot after a successful initialization for FDO. As such, Active SHOULD be set to True during FDO credential initialization. If specific manufacturing processes require a boot without FDO, setting Active to True is deferred.

Start of informative comment

TPM initialization for FDO can happen any time after the TPM manufacture, including:

- As part of manufacture of the TPM part
- As part of manufacture of the board-level computer using the TPM
- After completion of the board-level computer, but before operating system installation
- After operating system installation, during the supply chain
- After device has been in operation

End of informative comment

6.2. Device Startup with and without FDO§

Start of informative comment

- Use of SHOULD rather than MUST: A legacy OS that pre-dates this specification might not restrict access to the FDO credentials during runtime.
- NV Indices can be created ahead, but not populated. This is assumed in the discussion of "empty" NV indices.
- Related to items [5] and [6], below: A successful run of FDO causes the credentials to beupdated, so that FDO can run again. In some situations, the product does not wish ever to run FDO again. In this case, the FDO credentials can be deleted from the TPM by the FDO implementation after the successful run of the protocol. In addition, the startup code that checks the FDO credentials is also able to delete these credentials, as a “belt and suspenders” security check.
- As per FDO specification, FDO completes successfully in the Device when the message TO2.Done is transmitted. To ensure that this message is received, the FDO device waits (“dallies”) for a TO2.Done2 message to be received, so it can retransmit the TO2.Done message.

End of informative comment

On device startup:

1. If the FDO software is not installed, the device SHOULD still enforce that the regular OS cannot gain access to the FDO credentials. For authentication driven separation between ROE and OS, the authValues for the FDO credentials SHOULD be set. For time-based separation (using ReadLock and WriteLock), a shim layer SHOULD be started before the OS that sets ReadLock and WriteLock in order to restrict access to the FDO credentials. Also, the device SHALL delete the stored FDO credentials, unless FDO software is intended to be installed later.
2. If the content of FDO NV indices is “empty” (i.e. all Zero or TPMA_NV_WRITTEN is clear), then the ROE or the shim layer SHALL not invalidate access to the FDO credentials and reset the authValue to the empty password. This allows a FDO credential provisioning to be executed in the context of the OS, whilst consumption of the FDO credentials remains limited to the ROE.
3. If FDO software is installed, the FDO software SHALL query the TPM for FDO credentials, as described in 5.10.1.
4. If the FDO software cannot find credentials, FDO software SHALL terminate, and normal system startup ensues.
5. If Active == False, the FDO protocol SHALL NOT be executed, and normal system startup ensues. During this check, FDO credentials MAY be deleted, if it is known that FDO will not run again. Otherwise, modifications of the FDO credentials SHALL NOT occur.
6. If Active == True, FDO SHALL be run.

- If FDO completes successfully, the FDO TPM credentials SHALL be:
  - updated, and Active MAY be set to False
  - deleted
- If FDO does not complete successfully, the FDO TPM credentials SHALL NOT be updated, and Active SHALL remain True.

7. Other FDO Credentials Stored on Device§

Start of informative comment

It is legal for a device to store FDO credentials in a device-specific manner, outside the TPM. In this case, it is device-dependent whether the TPM is queried for credentials.

End of informative comment

A device which has both device-specific FDO credentials and TPM-based FDO credentials:

- MAY try each set of credentials in turn.
- MAY try the credentials in any order.
- SHOULD delete the unused set of credentials.

8. FDO NV and KEY Templates§

This section defines default Templates for the FDO Device key (§ 4.6 FDO Device Key) and FDO HMAC secret (§ 4.7 FDO HMAC Secret). Fields that are common to the default Templates of both keys (object attributes, unique string and authorization) are defined in § 4 Storage of FDO Credentials in the TPM

8.1. Attributes for FDO NV Indices§

Start of informative comment

The FDO keys can change at different stages of the FDO protocol. The HMAC secret changes during every TO2 interaction and the Device ID key can be changed as well.

However, the same keys can still be used after a TPM2_Clear was issued (which removes all persistent as well as RNG-created keys) such as to preserve the FDO credentials over a resale when user data is usually deleted.

In order to preserve the keys over a TPM2_Clear, the HMAC secret as well as the Device ID key are implemented as primary keys derived from the Endorsement Primary Unique String as well as a unique string value that is stored in an NV index. In order to generate a new key, the unique string is regenerated causing the keys to be recreated based on the new unique strings.

The unique strings are stored in an NV index under the Platform hierarchy in order to not be deleted during a TPM2_Clear. The keys themselves can be stored persistently in the TPM for improved performance or to remain usable in case the Endorsement Hierarchy password is set.

> NOTE: More details on the function of the unique string is presented in the TPM Library Specification “Part 1 – Architecture” Section 27.2.7 “unique”.

End of informative comment

The following NV Index attribute settings SHOULD be used for the FDO Public Device Credentials.

Parameter	Type	DCActive	DCTPM.*	DCOV.*
nvIndex	TPMI_RH_NV_INDEX	Refer to § 4.2 Handles for FDO Credentials	Refer to § 4.2 Handles for FDO Credentials	Refer to § 4.2 Handles for FDO Credentials
		TPM_ALG_SHA256	TPM_ALG_SHA256	TPM_ALG_SHA256

nameAlg	TPMI_ALG_HASH	or TPM_ALG_SHA384	or TPM_ALG_SHA384	or TPM_ALG_SHA384
attributes	TPMA_NV	Refer to Table 11	Refer to Table 11	Refer to Table 11
authPolicy	TPM2B_DIGEST			
size	UINT16	0 (0x0000)	0 (0x0000)	0 (0x0000)
buffer	BYTE			
dataSize	UINT16	1	512 (Size of FDO Public Device Credentials)	(Size of FDO Ownership Voucher)
authValue	TPM2B_DIGEST			
size	UINT16	0 (0x0000)	0 if Locking ROE is used = 32 in case of system service ROE	0 (0x0000)
buffer	BYTE		ROE Secret value	

-- Table continues --

Parameter	**Type**	**HMAC Unique String**	**Device Key Unique String**
nvIndex	TPMI_RH_NV_INDEX	Refer to § 4.2 Handles for FDO Credentials	Refer to § 4.2 Handles for FDO Credentials
nameAlg	TPMI_ALG_HASH	TPM_ALG_SHA256 or TPM_ALG_SHA384	TPM_ALG_SHA256 or TPM_ALG_SHA384
attributes	TPMA_NV	Refer to Table 11	Refer to Table 11
authPolicy	TPM2B_DIGEST		
size	UINT16	0 (0x0000)	0 (0x0000)
buffer	BYTE	See Table 12	See Table 12
dataSize	UINT16	sizeof(HMAC key alg)	sizeof(Device Key alg)
authValue	TPM2B_DIGEST		
size	UINT16	0 if Locking ROE is used = 32 in case of system service ROE	0 if Locking ROE is used = 32 in case of system service ROE
buffer	BYTE	ROE Secret value	ROE Secret value

***Table 8* – NV Index Attributes**

✧ ✧ ✧

Parameter	**BitMask**	**DCActive**	**DCTPM, DCKEY, HMAC U/S,**	**DCOV**

			Device Key U/S	
attributes	TPMA_NV_PPWRITE	**0**	**0**	**0**
	TPMA_NV_OWNERWRITE	**1**	**0**	**1**
	TPMA_NV_AUTHWRITE	**1**	**1**	**1**
	TPMA_NV_POLICYWRITE	**0**	**0**	**0**
	TPMA_NV_POLICY_DELETE	**0**	**0**	**0**
	TPMA_NV_WRITELOCKED	**0**	**0**	**0**
	TPMA_NV_WRITEALL	**0**	**0**	**0**
	TPMA_NV_WRITEDEFINE	**0**	**0**	**0**
	TPMA_NV_WRITE_STCLEAR	**0**	**0**	**0**
	TPMA_NV_GLOBALLOCK	**0**	**0**	**0**
	TPMA_NV_PPREAD	**0**	**0**	**0**
	TPMA_NV_OWNERREAD	**1**	**0**	**1**
	TPMA_NV_AUTHREAD	**1**	**1**	**1**
	TPMA_NV_POLICYREAD	**0**	**0**	**0**
	TPMA_NV_NO_DA	**1**	**1**	**1**
	TPMA_NV_ORDERLY	**0**	**0**	**0**
	TPMA_NV_CLEAR_STCLEAR	**0**	**0**	**0**
	TPMA_NV_READLOCKED	**0**	**0**	**0**
	TPMA_NV_WRITTEN	**0**	**0**	**0**
	TPMA_NV_PLATFORMCREATE (*) SHOULD BE (1,1) otherwise (0,0)	**1***	**1***	**0**
	TPMA_NV_TPMA_NV_READ_STCLEAR	**0**	**0**	**0**

***Table 9* – NV Index bits**

> NOTE: HMAC U/S, Device Key U/S -- U/S stands for Unique String.

✧ ✧ ✧

8.2. FDO Device key & HMAC key Templates§

Parameter	Type	Device Key	HMAC key
type	TPMI_ALG_PUBLIC	TPM_ALG_ECC	TPM_ALG_KEYEDHASH
nameAlg	TPMI_ALG_HASH	TPM_ALG_SHA256 or	TPM_ALG_SHA256 or

		TPM_ALG_SHA384 *)	TPM_ALG_SHA384 *)
objectAttributes	TPMA_OBJECT	Refer to Table 11	Refer to Table 11
authPolicy	TPM2B_DIGEST		
size	UINT16	sizeof(nameAlg)	sizeof(nameAlg)
buffer	BYTE	See Table 12	See Table 12
parameters	TPMS_ECC_PARMS or TPMS_KEYEDHASH_PARMS		
symmetric->algorithm	TPMI_ALG_SYM_OBJECT	TPM_ALG_NULL	N/A
symmetric->keyBits	TPMI_AES_KEY_BITS	NULL	N/A
symmetric->mode	TPMI_SYM_MODE	NULL	N/A
symmetric->details		NULL	N/A
scheme->scheme	TPMI_ALG_ECC_SCHEME	TPM_ALG_ECDSA	TPM_ALG_HMAC
scheme->details->hashAlg		TPM_ALG_SHA256 or TPM_ALG_SHA384 *)	TPM_ALG_SHA256 or TPM_ALG_SHA384 *)
curveID	TPMI_ECC_CURVE	TPM_ECC_NIST_P256 or TPM_ECC_NIST_P384 *)	N/A
kdf->scheme	TPMI_ALG_KDF	TPM_ALG_NULL	N/A
kdf->details		NULL	N/A
unique	UNION (see next column)	TPMS_ECC_POINT	TPM2B_DIGEST
x->size	UINT16	sizeof()	N/A
x->buffer	BYTE	Content of Device Key Unique String (1st half)	N/A
y->size	UINT16	sizeof()	N/A
y->buffer	BYTE	Content of Device Key Unique String (2nd half)	N/A
size	UINT16	N/A	sizeof()
buffer	BYTE	N/A	Content of HMAC Unique String

***Table 10* - FDO Device Key Templates**

*) The fields MUST match according to algorithms, either all SHA256 and P256 or all SHA384 and P384

Start of informative comment

The Object attributes define the properties of an Object and are described in the TPM Library specification [TPM 2LIB-2019], Part 2. A short description of the meaning of the attribute settings defined in this section is provided below.

Protection:

- To have assurance that the key material is generated by the TPM and is not be duplicated for use outside of the TPM, *fixedTPM*, *fixedParent* and *sensitiveDataOrigin* are SET
  - As the key is not permitted to be duplicated, *encryptedDuplication* is CLEAR

Usage:

- To define the key as unrestricted signing key, *sign* is SET and *restricted* is CLEAR
  - The key is used to sign externally provided data and not TPM internal states
  - To disallow the key to be used for any other purpose, i.e., encryption or signing X.509 certificates, *decrypt* and *signX509* are CLEAR

DA (Dictionary Attack):

- To disable dictionary attack protection for the Object, *noDA* is SET

Persistence:

- To enable the key to be made persistent, stClear is CLEAR

End of informative comment

The following object attribute settings SHALL be used by the default Templates for the FDO Device key (§ 4.6 FDO Device Key) and the FDO HMAC secret (§ 4.7 FDO HMAC Secret).

Parameter	Type	Device Key and HMAC key
objectAttributes	TPMA_OBJECT	fixedTPM = 1 stClear = 0 fixedParent = 1 sensitiveDataOrigin = 1 userWithAuth = 0 adminWithPolicy = 0 noDA = 0 encryptedDuplication = 0 restricted = 0 decrypt = 0 sign = 1

Table 11 **- Object attributes for FDO keys**

Start of informative comment

As per TPM specification, the Policy for the objects is stored in a digest representation, as follows:

```
Digest ( Digest (PolicyNV) || Digest (PolicySecret ))
```

> NOTE: TODO: We intend to include actual digest values in the table below. These are not available at the time of this writing.

End of informative comment

NameAlg	Device Key	HMAC Secret
Description	PolicyNV(Device Key Unique String, offset = 0, size = 1, operation = GEQ, operand = 0) PolicySecet(Device Key Unique String NV Index)	PolicyNV(HMAC Unique String, offset = 0, size = 1, operation = GEQ, operand = 0 PolicySecret(HMAC Unique String NV index)

Table 12 **– Object policy for FDO keys**

9. IMPLEMENTATION OF FDO USING TPM§

Start of informative comment

The below table summarizes the cryptographic operations that are performed during FDO and the corresponding TPM commands that are used.

Category	Algorithm	Cryptographic primitive	TPM commands
Device HMAC	HMAC-SHA256 HMAC-SHA384	HMAC key generation (for FDO provisioning) HMAC signature	TPM2_Create, TPM2_Load, TPM2_EvictControl (to persist key) TPM2_HMAC or HMAC sequence commands
Owner key Hash	SHA-256, SHA-384	Hash	TPM2_Hash or Hash sequence commands
Owner attestation, RSA	RSA 2048, RSA 3072 PSS, PKCS v1.5 SHA-256, SHA-384	RSA signature verification	TPM2_LoadExternal (to load public key), TPM2_VerifySignature
Owner attestation, ECC	NIST P-256, NIST P-384 SHA-256, SHA-384	ECDSA signature verification	TPM2_LoadExternal (to load public key), TPM2_VerifySignature

Device attestation, DAA	Intel® EPID: EPID1.0, 1.1	Intel® EPID signature generation.	Not supported by TPM. TPM ECDAA and EPID 2.0 are not part of the FDO 1.1 Proposed Standard.
Device attestation, ECDSA	NIST P-256, NIST P-384 SHA-256, SHA-384	ECC key generation (for FDO provisioning) ECDSA signature generation	TPM2_CreatePrimary, TPM2_EvictControl (to persist key) TPM2_Sign
Key exchange, RSA	RSA 2048, RSA 3072 OAEP SHA-256	RSA key transport	TPM2_LoadExternal (to load public key), TPM2_RSA_Encrypt
Key exchange, ECC	NIST P-256, NIST P-384	ECDH key agreement (with 2 ephemeral keys)	TPM2_LoadExternal (to load public key), TPM2_ECDH_KeyGen (TPM2_ECDH_ZGen)
Key derivation	KDF in CTR mode using HMAC-SHA256, HMAC-SHA384 (SP800-108)	Key derivation function	No direct support by TPM via KDF command, TPM2_HMAC or HMAC sequence commands can be used for processor derivation. (*)
Session, encryption	AES-128, AES-256 CTR, CBC	Encryption/decryption	TPM2_EncryptDecrypt2 (not included in PC-Client TPM Profile (PTP))
Session, HMAC	HMAC-SHA256 HMAC-SHA384	HMAC signature	TPM2_HMAC or HMAC sequence commands
Session, AEAD	AES-128, AES-256 GCM, CCM	Authenticated encryption	TPM2_EncryptDecrypt2 supports this function in the TPM library specification, but not as a mandatory part of the PC Client TPM Profile Specification

***Table 13* – Cryptographic operations used for FDO**

*) Both FDO and TPM refer to NIST SP-800-108 section 5.1 “KDF in Counter Mode.” In the algorithm, the ‘L’ value indicates the number of bits of KDF being created. The FDO spec uses the actual number of bits needed, and the TPM specification (Part 1, section 28, “Object Derivation”) pegs this value at 8192. Since the algorithm includes abs(L) in the hash, the resultant key will be different between the two specifications. The TPM supports the current FDO specification using hash commands, but the key results in system memory, and must be subsequently copied to the TPM.

In order to deploy FDO credentials to the TPM and assert correct and unique deployment, several TPM functionalities surrounding TPM2_ActivateCredential(), TPM2_Certify(), TPM2_CertifyCreation(), Authentication

or Audit sessions can be used.

End of informative comment

Index§

Terms defined by this specification§

References§

Normative References§

[EAT]
G. Mandyam; L. Lundblade; J. O'Donoghue. *The Entity Attestation Token (EAT) draft-ietf-rats-eat*. Standards Track. URL: https://datatracker.ietf.org/doc/draft-ietf-rats-eat

[FDO-Specification]
Geoffrey Cooper; et al. *FIDO Device Onboard Specification*. 19 April 2022. Proposed Standard. URL: https://fidoalliance.org/specs/FDO/FIDO-Device-Onboard-PS-v1.1-20220419/FIDO-Device-Onboard-PS-v1.1-20220419.html

[RFC8949]
C. Bormann; P. Hoffman. *Concise Binary Object Representation (CBOR)*. December 2020. RFC. URL: https://www.rfc-editor.org/rfc/rfc8949.html

[TPM]
TPM Main Specification. URL: http://www.trustedcomputinggroup.org/resources/tpm_main_specification

Informative References§

[802.1AR-2018]
802.1AR-2018 - IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity. June 14, 2018. Standard. URL: https://standards.ieee.org/standard/802_1AR-2018.html

[RFC8610]
H. Birkholz; C. Vigano; C. Bormann. *Concise Data Definition Language (CDDL): A Notational Convention to*

Express Concise Binary Object Representation (CBOR) and JSON Data Structures. June 2019. Proposed Standard. URL: https://tools.ietf.org/html/rfc8610

[TPM2LIB-2019]
Trusted Platform Module Library Specification, Family "2.0", Level 00, Revision 01.59 November 8, 2019. URL: https://trustedcomputinggroup.org/wp-content/uploads/TCG_TPM2_r1p59_Part1_Architecture_pub.pdf

[TPMIdentKeys]
TPM 2.0 Keys for Device Identity and Attestation, V1.00, R12. October 8, 2021. URL: https://trustedcomputinggroup.org/wp-content/uploads/TPM-2p0-Keys-for-Device-Identity-and-Attestation_v1_r12_pub10082021.pdf