TRUSTED® COMPUTING GROUP

SPECIFICATION

Hardware Requirements for a Device Identifier Composition Engine

Version	1.0
Revision	0.91
January 17, 2024	

Contact: admin@trustedcomputinggroup.org

Public Review

Work in Progress
This document is an intermediate draft for comment only and is subject to change without notice. Readers should not design products based on this document.

DISCLAIMERS, NOTICES, AND LICENSE TERMS

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

CONTENTS

1 SCOPE

This specification describes the hardware requirements and process for creating an identity value that is derived from a Unique Device Secret (UDS) and the identity (a condensed cryptographic representation) of the First Measured Code (FMC). This specification calls the derived value the Compound Device Identifier (CDI). The composition of the Compound Device Identifier may include hardware state or configuration that influences the execution of the FMC.

One of the possible uses of the Compound Device Identifier is to attest to the trustworthiness of an embedded device.

The intended audience for this document is designers of programmable components when they do not have access to a TPM.

1.1 Key Words

The key words “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 statement.

EXAMPLE: 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* ...

To understand the TCG specification the user must read the specification. (This use of MUST does not require any action).

End of informative comment

2 REFERENCES

[1] Trusted Computing Group, "TCG Glossary," 2017. [Online]. Available: https://www.trustedcomputinggroup.com.

[2] Trusted Computing Group, "DICE Layering Architecture," July 2020. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-layering-architecture/.

[3] National Institute of Standards and Technology, "Recommendation for Key Management," May 2020. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-57-part-1/rev-5/final.

[4] Trusted Computing Group, "DICE Protection Environment (DPE)," May 2023. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-protection-environment.

[5] Internet Engineering Task Force, "Internet Security Glossary, Version 2," 2007. [Online]. Available: https://tools.ietf.org/html/rfc4949.

[6] GlobalPlatform Technology, "Root of Trust Definitions and Requirements," 2018. [Online]. Available: http://globalplatform.org.

3 TERMS AND DEFINITIONS

For the purposes of this document, the following terms and definitions apply.

3.1 Glossary

TERM	DEFINITION
Compound Device Identifier	A Compound Device Identifier (CDI) is a value used to identify the FMC and the system executing the FMC.
Device	A device is a platform that integrates a programmable component with other optional programmable components and peripherals.
Digest	A digest is the result of a hash operation.
Device Identifier Composition Engine	A Device Identifier Composition Engine (DICE) is immutable or updated through a secure process controlled by its manufacturer. The DICE creates the CDI.
DICE Root of Trust	Any Root of Trust that contains, or comprises, DICE. See *Root of Trust,* [1].
First Measured Code	First Measured Code (FMC) is the code and/or configuration information that is executed and/or takes effect immediately following the Device Identifier Composition Engine (DICE). FMC is synonymous with Layer 0 in the DICE Layering Architecture specification in [2].
Measurement	A measurement is a cryptographic hash (or equivalent) of code or data.
Unique Device Secret	A Unique Device Secret (UDS) is normally known only to a DICE and used in the creation of a CDI by the DICE. Depending on the provisioning process, it may be known to the manufacturer or owner.

3.2 Acronyms

ABBREVIATION	DESCRIPTION
CDI	Compound Device Identifier
DICE	Device Identifier Composition Engine
FMC	First Measured Code
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TCI	TCB Component Identifier (see [2])
TPM	Trusted Platform Module
UDS	Unique Device Secret

3.3 Symbols

The following symbols are also defined for use in this document:

A || B	concatenation of B to A
F()	denotes a function F
H()	denotes a hash function
HMAC(k, m)	denotes the HMAC function over message *m* using key *k*

4 INTRODUCTION

A Device Identifier Composition Engine is a capability of a device's Root of Trust (RoT). This specification uses the term *the DICE*, or simply *DICE* to refer to functionality that includes: UDS, TCI, and CDI values, the ability to create TCI and CDI values, and the ability to store, protect, and convey these values securely.

A Compound Device Identifier (CDI) is derived using both a Unique Device Secret (UDS) and the measurement of the FMC (i.e., a TCB Component Identifier, or TCI) [2]. A CDI can optionally include measurements of hardware state information and configuration data that could influence execution of the FMC. The CDI is generated by DICE, which has exclusive access to the UDS after reset and before transferring control to the measured FMC. The general process is shown in Figure 1 with an illustration of the computation of the CDI. The UDS is provisioned by the manufacturer in any way that is consistent with this specification. Any revision or change in the UDS or any of the measured components results in a different value for the CDI.

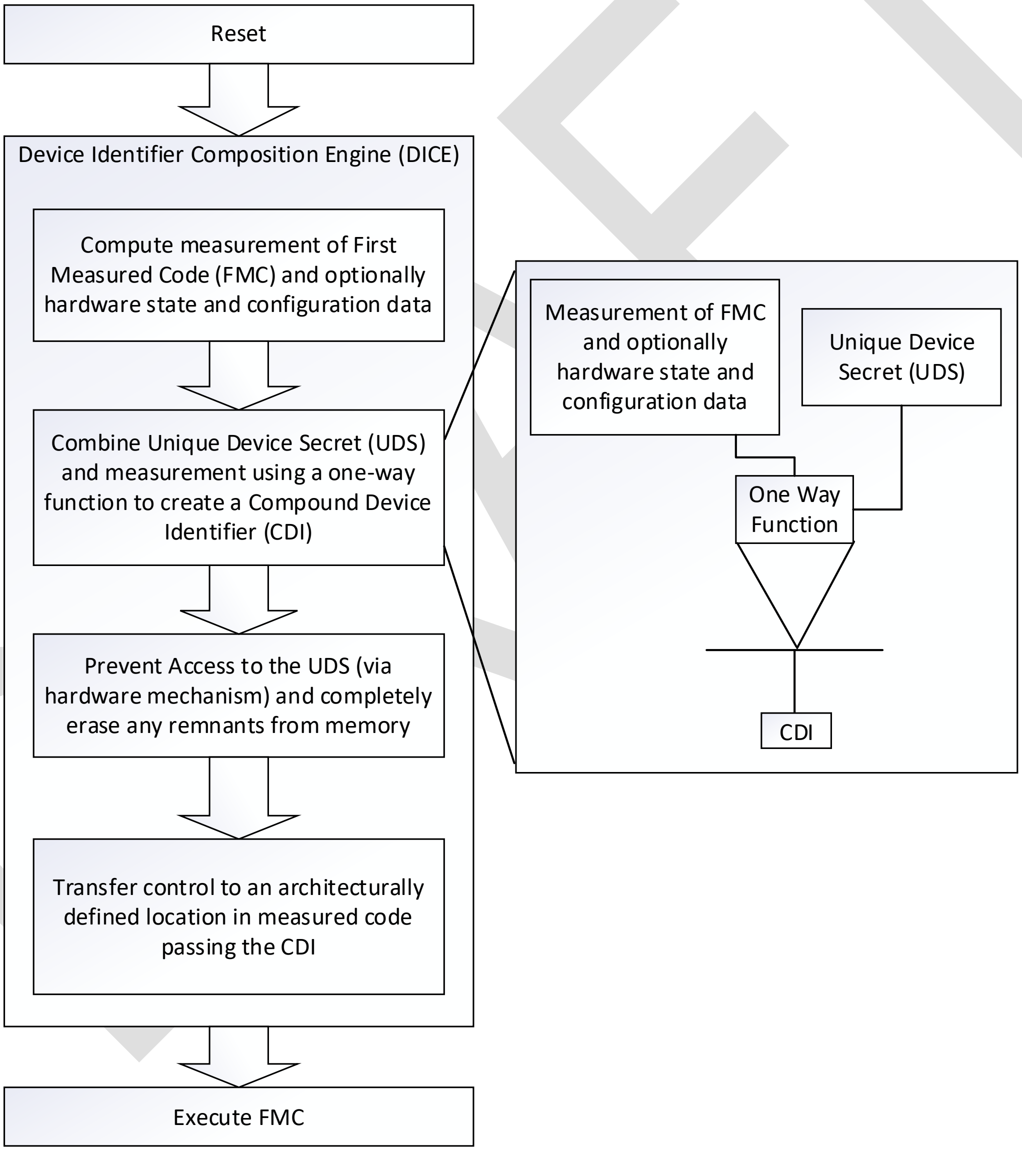
Reset
Device Identifier Composition Engine (DICE)
Compute measurement of First Measured Code (FMC) and optionally hardware state and configuration data
Combine Unique Device Secret (UDS) and measurement using a one-way function to create a Compound Device Identifier (CDI)
Prevent Access to the UDS (via hardware mechanism) and completely erase any remnants from memory
Transfer control to an architecturally defined location in measured code passing the CDI
Execute FMC
Measurement of FMC and optionally hardware state and configuration data
Unique Device Secret (UDS)
One Way Function
CDI

Figure 1: Compound Device Identifier Derivation Process

The UDS and the measurement of the FMC must be cryptographically mixed in such a way that it is infeasible to use the CDI and the code measurement to recover the UDS. This may be accomplished by the DICE using a secure hash algorithm to hash the concatenation of the two values. Alternatively, the two values could be used in a MAC function, for example HMAC, with the UDS as the MAC key. An HMAC would provide a higher level of protection for the UDS than would a simple hash, as described in NIST SP800-57 [3]. The specific method to combine the values is the manufacturer's choice because it does not affect interoperability.

The secure hash function is:

$$\mathbf{H}\ (\ \mathit{UDS}\ ||\ \mathbf{H}\ (\ \mathit{FMC}\ )\ ) \tag{1}$$

The secure HMAC function is:

$$\mathbf{HMAC}\ (\ \mathit{UDS},\ \mathbf{H}\ (\ \mathit{FMC}\ )\ ) \tag{2}$$

Where *UDS* is the Unique Device Secret, and *FMC* is the First Measured Code.

The device is responsible, where required, to protect access (read, write, and modify) to the CDI. It may not be possible for the FMC to protect the CDI. How protection of the CDI is achieved is outside the scope of this specification.

A benefit of the CDI is that the CDI has a different value when the FMC changes. For example, if the FMC was replaced by malware, the previous CDI is not available to that malware. Likewise, if the FMC is updated with a security patch, a new CDI is generated. The FMC might use a secure update process that makes the code essentially immutable for end customers in the absence of assistance from the manufacturer. An example is if the update process for mutable code verifies the cryptographic signature of updates before allowing them to be installed, based on a public key for which the manufacturer owns the private key. Each successive version of the FMC installed results in a different CDI value.

This specification has requirements for two different categories of immutability for the DICE. For simple devices, it might be possible for the DICE and all its dependencies to be invariant and not change after manufacturing. More complex systems might have a DICE that could be influenced directly or indirectly by the manufacturer, such as a CPU vendor or a baseboard management controller vendor. According to the requirements in this specification, modifications to the DICE engine or its dependencies are not reflected in the resulting CDI. It may be necessary for a manufacturer or one of their suppliers to influence the DICE directly or indirectly to balance risks associated with complex systems. The protection mechanisms for modifying the DICE or its dependencies are the basis for confidence when identifying changes in the UDS, the FMC, or measured configuration changes via the resulting CDI. The protection mechanisms for modifying the DICE or its dependencies must be inherently trusted. The strength of protections for the update process for the DICE or its dependencies are central to a customer's ability to trust the CDI.

4.1 First Measured Code

The first measured code is the code and/or configuration information that is executed immediately following the Device Identifier Composition Engine. Essentially, FMC is the first stage of functionality after DICE measurement during device initialization or boot (such as a bootloader, boot ROM, or FPGA bitstream). The engine that performs the computation of the CDI may be updated, but those updates are not measured and included in the CDI and must be inherently trusted.

Since the CDI, and thus the identity of the device, will change whenever the FMC is changed, it is up to the manufacturer to decide how frequently FMC will change throughout a device's lifetime. For example, FMC could be implemented in ROM and/or OTP fuses in a device, such that the device identity will only change when ROM or OTP is changed by the manufacturer. If FMC changes often, the volatility of a CDI could be intentional, and reflected in updated device identifiers. The manufacturer must cope with this, and the correlation (or lack thereof) between CDIs is outside the scope of this specification.

Start of Informative Comment

The term First Measured Code replaces the term First Mutable Code in previous versions of this specification. This change was made because use of the previous term, First Mutable Code, implied that the code that executes immediately following DICE was required to be physically changeable or updatable by some process. This is not true for all use cases. The code that is executed immediately following DICE, which is also referred to as *Layer 0*, may not be updatable despite it being referred to as First Mutable Code.

Even though the mutability of Layer 0 may differ between implementations of this specification, all implementations must compute the CDI value using a measurement of Layer 0. So, given that Layer 0 may not literally be the first mutable code on a device, and that the first measurement in a DICE measurement chain will always correspond to Layer 0 regardless of mutability, Layer 0 is now referred to as First Measured Code.

End of Informative Comment

5 REQUIREMENTS

5.1 Unique Device Secret Properties

The following requirements apply to a Unique Device Secret:

1) UDS values MUST be uncorrelated and statistically unique.
2) The UDS MUST NOT be used as an identity value by any other entity.
3) The UDS MUST be of a length sufficient to support the security strength of cryptographic algorithms and key sizes required by current best practice, e.g. [3]

Start of Informative Comment

Requirements for specific cryptographic algorithms are outside the scope of this specification. For guidance on security strength and its correlation to cryptographic algorithm and key-size selection, see [3].

Note that the practical maximum security strength of functions that rely on the UDS, i.e., any function using DICE-derived secrets or keys, is determined by the size of the UDS. Any key that is derived from the UDS (and therefore, CDI) cannot be of greater security strength than the UDS.

For example, the attestation process for a device reports the software state and identity of the device. The length of the UDS is an upper bound on the security strength of attestation functions because it represents a limit on meaningful attestation key size.

End of Informative Comment

5.2 Device Identifier Composition Engine Properties

A Device Identifier Composition Engine is a capability of a device's Root of Trust (RoT). This specification allows for two classifications of the DICE. One category of DICE is immutable, and the other is securely updatable by the manufacturer of the DICE.

The following requirements apply:

1) The DICE MUST have exclusive read access to a UDS.

Start of Informative Comment

This means that the packaging of the programmable component that implements DICE will normally preclude use, reading, and observation of the UDS by an entity other than DICE.

Typically, read access to the storage location containing the UDS will be enabled when a hardware event, such as a reset, causes the DICE to begin execution. Then read access of the storage location will be disabled by a software command. Other implementations are possible.

End of Informative Comment

2) If the device has a debug port or debug mode:
   a) The debug port or debug mode SHALL only be enabled at reset or when explicitly enabled by software that executes after the DICE.
   b) When the debug port or debug mode is enabled, any attempt to read the UDS (including from the DICE) SHALL be denied or produce a value that is uncorrelated with the UDS.

Start of Informative Comment

Any constant value such as all 0's is an uncorrelated value.

End of Informative Comment

5.2.1 Immutable DICE Properties

The following requirements apply to immutable DICE implementations:

1) The DICE implemented on the device SHALL be immutable.
2) An immutable DICE SHALL be immutable by the end of the manufacturing process of the device.

5.2.2 Updatable DICE Properties

The following requirements apply to updatable DICE implementations:

1) An updatable DICE SHALL only be updated through a secure process controlled by the DICE manufacturer.
2) After updating through a manufacturer-controlled process the DICE SHALL comply with this specification.
3) The manufacturer-controlled update process SHOULD NOT change the CDI.

Start of Informative Comment

Update of a DICE implementation does not normally result in a change in CDI value because the DICE itself is not measured. If a secure manufacturer-controlled process changes the DICE or one of its dependencies, and this results in a change to a CDI value, usage of a mechanism for re-endorsing or re-certifying the new device identity is strongly recommended.

End of Informative Comment

5.3 Compound Device Identifier Properties

The specification of normative requirements for the CDI, other than the requirements stated in section 5.4 is outside the scope of this specification.

Start of Informative Comment

The device may need dedicated hardware to protect access (read, write, and modify) to the CDI. If the device executes code before the FMC, hardware is responsible for preventing access to the CDI.

If hardware is unavailable to protect the CDI, then the FMC provided by the manufacturer is responsible for reducing opportunities for exposure of the CDI to unauthorized entities. Devices that leak a CDI produced from measurement of authorized FMC may be vulnerable to a replay attack and impersonation.

Device manufacturers are encouraged to use best practices (for example: ISO/IEC 27034) to prevent leakage of the CDI. Measures taken may include:

- Avoiding design, coding, and logic errors.
- Erasure of the CDI immediately after its use (e.g., in RAM, registers, cache).
- Use of DICE Protection Environment (DPE) [3].

A CDI that has been leaked should be made obsolete; there are a variety of techniques to achieve this. One example is to update the FMC. This will result in a new measurement for FMC and therefore a new CDI value. This is especially useful if the code responsible for the leak was the FMC itself. Updating the code to fix the bug naturally results in the obsolescence of the original (leaked) CDI value. A change to CDI may necessitate re-endorsment or recertification of the new device identity.

End of Informative Comment

5.4 DICE Operation

The following requirements apply to DICE operation:

1) The DICE SHALL execute without interference or alteration each time the device is reset.
2) The DICE SHALL NOT transfer execution to any code that is not part of the DICE Root of Trust, other than the FMC.

3) Before execution of the FMC, the DICE Root of Trust SHALL measure the FMC and the DICE SHALL combine the UDS with the measurement of the FMC in such a way that the UDS cannot be deduced from the CDI, even if the measurement is known.
4) The DICE SHALL create this CDI using a one-way function that preserves the security strength of the UDS.

Start of Informative Comment

As noted previously, see section 5.1, the practical maximum security strength of functions that rely on the UDS is determined by the length of the UDS and, consequently, the CDI. For guidance on security strength and its correlation to cryptographic algorithm and key-selection, see [3].

One example of a function that preserves the security strength of the UDS is an HMAC with a suitable hash algorithm. An HMAC provides as many bits of security strength as the number of bits in a digest, while the same algorithm used in a hash would only provide about half as many bits. Using a UDS as an HMAC key would make the security strength as strong as the UDS.

End of Informative Comment

5) Before execution of the FMC, access to the UDS SHALL be disabled until the next reset.

Start of Informative Comment

Examples of methods for disabling access to the UDS include: placing the UDS into read-once memory; using an explicit software instruction; using hardware that can determine whether the instruction pointer is inside the range of DICE instructions and only allowing access to the UDS from that range; execution of DICE on a secure coprocessor and allowing access to the UDS only from the secure coprocessor. Other implementations are possible.

End of Informative Comment

6) The DICE SHALL write the CDI to a location to which the measured FMC has exclusive access as long as the FMC requires exclusive access.

Start of Informative Comment

The FMC is expected to use and erase the CDI and any values that could be used to determine the CDI. While the FMC uses the CDI, it needs the ability to prevent access to the CDI and to prevent disclosure of the value.

Access includes read, write, and modify.

Use of the CDI by the FMC is outside the scope of this specification.

The device starts execution of the FMC at an architecturally defined address in the range of the FMC that was measured.

End of Informative Comment