SCMS MANAGER™
SECURITY CREDENTIAL MANAGEMENT SYSTEM

END-ENTITY SECURITY REQUIREMENTS, DESIGN GUIDANCE, AND VALIDATION APPROACH

VERSION 1.00

March 14, 2021

Author
SCMS Manager

TABLE OF CONTENTS

SCMS Manager End-Entity Security Requirements,

Design Guidance, and Validation Approach

1 TERMS AND DEFINITIONS

Application: A process on a processor that is involved in data exchange with another processor.

Authorized Enrollment Technician: Authorized personnel or automated processes that perform device enrollment.

Authorized Personnel: Individuals who are authorized by name or role to physically or logically access the End-Entity or enrollment systems during enrollment.

Connected Architecture: An End-Entity architecture in which the Host Processor and HSM are physically separate but connected using a dedicated physical connection such that physical access would be needed for another processor to write to or read from the connection.

End-Entity: A device that is provisioned with and uses IEEE 1609.2 certificates.

Enrollment: The process of provisioning an End-Entity with an SCMS-provided IEEE 1609.2 Enrollment Certificate

Excluded non-privileged application: An application that does not perform any IEEE 1609.2 signing operations and has been excluded from the boundary based on its inability to materially affect correct and secure operations of any Privileged Application.

FIPS: Federal Information Processing Standards

Hardware Security Module: a physically and logically secure cryptographic processor that 1) performs IEEE 1609.2 private key operations (signing) and 2) protects IEEE 1609.2 secret and private key material. (Devices that employ hardware-based cryptographic storage or operations for non-1609.2 based keys are called “Secure Hardware” in the context of this specification.)

Host Processor: A component of an End-Entity that executes at least one Privileged application.

Integrated Architecture: An End-Entity architecture in which the Host Processor and HSM are physically the same processor, i.e., all secure cryptographic operations and key storage are performed in the same processor as the application operations.

Networked Architecture: An End-Entity architecture in which the Host Processor and HSM are physically separate and connected over a data bus or network that may be accessed by other processors.

Non-Privileged Application: An application running on the Host Processor whose functionality does not require it to invoke IEEE 1609.2 private key signing operations from the HSM, or which requires user authentication before it will invoke those operations.

Other processors: processors that are within the physical boundary of the End-Entity that do not contain an application that performs IEEE 1609.2 signing operations

Physical Boundary: a physical boundary meeting the requirements of FIPS 140-2 Section 4.5 that protects either:

- the Host Processor
- the HSM
- both the Host Processor, HSM and connection between them (if conforming to the Connected Architecture)

An End-Entity may consist of one or more physical boundaries.

Privileged Application: An application running on the Host Processor 1) without user initiation or intervention AND 2) that is able to invoke IEEE 1609.2 private key signing operations from the HSM

Provider Service Identifier: A connected vehicle application identifier that is indicated in an End-Entity certificate

Secure Hardware: A secure, hardware-based cryptographic key storage and cryptographic operations processor that is used for purposes other than IEEE 1609.2 private key storage and signing.

Security Credential Management System (SCMS): A defined Public Key Infrastructure for provisioning and managing IEEE 1609.2-based certificates in the ITS trust domain.

SCMS Enrollment Responsible Party: The entity who is enrolling the End-Entity(s)

Security Critical Functionality: Any operations on the Host Processor that are intended to support correct use of IEEE 1609.2 cryptographic operations by the applications on the Host Processor. This includes both the operation of the applications themselves (so that only appropriate messages are submitted for signing) and also OS level features like access control and lower-layer features like operation of drivers for hardware modules.

Shared Physical Boundary: a 3-dimensional, physical contiguous boundary around BOTH the Host Processor and HSM (in the Connected Architecture) that implements physical security protections to the HSM in accordance with FIPS 140-2, Section 4.5

Targe of Evaluation (TOE): The certification boundary of the End-Entity. Elements outside of the TOE are not in the scope of evaluation and certification.

2 ACRONYMS

AES – Advanced Encryption Standard

ECC – Elliptic Curve Cryptography

ECDSA – Elliptic Curve Digital Signature Algorithm

EE – End-Entity

HMAC – Hash-based Message Authentication Code

HPSE – Hardware Protected Security Environment (described in SAE J3101)

HSM – Hardware Security Module (described in FIPS 140)

IPSEC – Internet Protocol Security

ITE – Institute of Transportation Engineers

MAP – Map Data (a V2X message)

OBE – On-Board Equipment

OBU – On-Board Unit

PSID – Provider Service Identifier

RSA – Rivest, Shamir, and Adleman

RSM – Road Safety Message (a V2X message)

RSU – Road-Side Unit

RTCM – Real Time Correction Message (a V2X message)

SCMS – Secure Certificate Management System

SCP – Secure Copy Protocol

SFTP – Secure File Transfer Protocol

SHA – Secure Hash Algorithm

SSH – Secure Shell

TLS – Transport Layer Security

TMC – Traffic Management Center

TSC – Traffic Signal Controller

USDOT – United States Department of Transportation

V2X – Vehicle to Everything

3 REFERENCE DOCUMENTS

ARC-IT Version 9.0 VS13: Intersection Safety Warning and Collision Avoidance (arc-it.net)

ARC-IT Version 9.0 Device Class 3: Device Class 3 Areas (arc-it.net)

ANSI X9.62: ANSI X9.62 - Public Key Cryptography for the Financial Services Industry: the Elliptic Curve Digital Signature Algorithm (ECDSA) | Engineering360 (globalspec.com)

IEEE 1609.2 IEEE 1609.2a-2017 - IEEE Standard for Wireless Access in Vehicular Environments--Security Services for Applications and Management Messages - Amendment 1

IEEE 1609.2.1 IEEE 1609.2.1-2020 - IEEE Standard for Wireless Access in Vehicular Environments (WAVE)--Certificate Management Interfaces for End Entities

ITE RSU Standard 1.0: To be published at RSU Standardization - Institute of Transportation Engineers (ite.org)

NIST FIPS 140-2: FIPS 140-2, Security Requirements for Cryptographic Modules | CSRC (nist.gov)

NIST FIPS 140-3: FIPS 140-3, Security Requirements for Cryptographic Modules | CSRC (nist.gov)

NIST Risk Management Framework: NVD - 800-53 (nist.gov)

NIST SP 800-53: SP 800-53 Rev. 5, Security and Privacy Controls for Info Systems and Organizations | CSRC (nist.gov)

NIST SP 800-57: SP 800-57 Part 1 Rev. 5, Recommendation for Key Management: Part 1 – General | CSRC (nist.gov)

SAE J3101: J3101: Hardware Protected Security for Ground Vehicles - SAE International

SCMS Manager Publications: Publications – SCMS Manager

Uptane: https://uptane.github.io

SCMS Manager Misbehavior Detection and Reporting Document to be published: Publications - SCMS Manager

4 OVERVIEW

This document provides requirements for End-Entity devices intended to be provisioned with 1609.2 certificates for connected vehicle operations in the United States. In addition to the requirements in this document, SCMS Manager will define a certification process associated with these requirements such that device manufacturers can establish that their devices are conformant with the requirements and so eligible to receive certificates (subject to any additional conditions that SCMS Manager specifies).

This information is in three primary categories: Requirements, Design Guidance, and Validation Approach. Each is described in more detail below.

Requirements. The requirements in this specification are drawn from, and generally consistent with, the Hardware, Software and OS Security Requirements established as part of the SCMS CV Pilots Documentation. Section 4.1 details requirements for End Entities based on three defined architectures. Each architecture is distinguished by the method in which cryptographic security is integrated with application host processing.

Design Guidance. The requirements are stated as a relatively high level with the goal of allowing for innovation in meeting those requirements as technology evolves. However, with technology that is currently available, there are some approaches that are likely to be used to meet those requirements. This section will detail those approaches.

Validation Approach. Validation approaches are expected to change over time and today are primarily self-declaration.

4.1 COMPONENTS, ARCHITECTURE AND TARGET OF EVALUATION

An End-Entity (EE) is defined as an entity that is provisioned by an SCMS provider with IEEE 1609.2 certificates used by one or more privileged applications to sign application data. A privileged application is an application which is entitled, within the design of the system, to create signed IEEE 1609.2 messages without explicit user intervention or initiation. A corrupted privileged application has the potential to create an arbitrary number of incorrect or malicious, but cryptographically valid, signed messages. It is therefore a key security goal of the system to protect not just the integrity of cryptographic operations, but the integrity of all operations of privileged applications. The term *security critical functionality* is used in this document to refer to any functionality within the EE whose correct operation is necessary to ensure that all signed messages are correct in terms of cryptographic properties and correct content.

An EE may be implemented in a variety of architectures depending on the device type and deployment environment. This specification defines and permits three architectures and associated Target of Evaluation (TOE) boundaries for an End-Entity.

Host Processor: The EE shall have a Host Processor, defined as a physical processor with an Operating System and at least one Privileged Application.

Hardware Security Module (HSM): The functionality that is used to 1) securely store IEEE 1609.2 private key material used in signing operations and 2) perform IEEE 1609.2 signing operations. May be distinct from or combined with the host processor.

Privileged Application: A process on the Host Processor that can invoke IEEE 1609.2 signing operations on the HSM without user intervention.

Non-privileged Application: A process on the Host Processor that is unable to invoke IEEE 1609.2 signing operations on the HSM, or that invokes signing operations only after user approval.

End-Entity "Target of Evaluation": The collection of functionality that is to be certified as meeting the requirements of this document: specifically, all Security Critical Functionality within the boundary of the End-Entity.

Exclusions: During certification, some components may be excluded from specific requirements of this specification.

- A non-privileged application may be an '*Excluded non-privileged application'*, defined as a process that 1) is a non-privileged application and 2) cannot materially affect the correct and secure operation of any privileged application.
- An *excluded operating system* is an operating system running on an HSM that 1) conforms to the FIPS 140-2 Section 4.6 definition of a *non-modifiable* Operating Environment and therefore 2) is not required to meet the requirements of Section 5.4.1.

Other Processors: A processor that contains no Privileged Applications or security critical functionality

Integrated Architecture: An EE conforming to an Integrated Architecture is defined as a single, shared processor that 1) executes one or more privileged applications, 2) implements secure hardware-protected key storage and 3) performs all 1609.2-based private key operations (signing). The Target of Evaluation (TOE) for an Integrated Architecture EE is the physical implementation and boundary of the shared processor along with its secure connectivity to the SCMS.

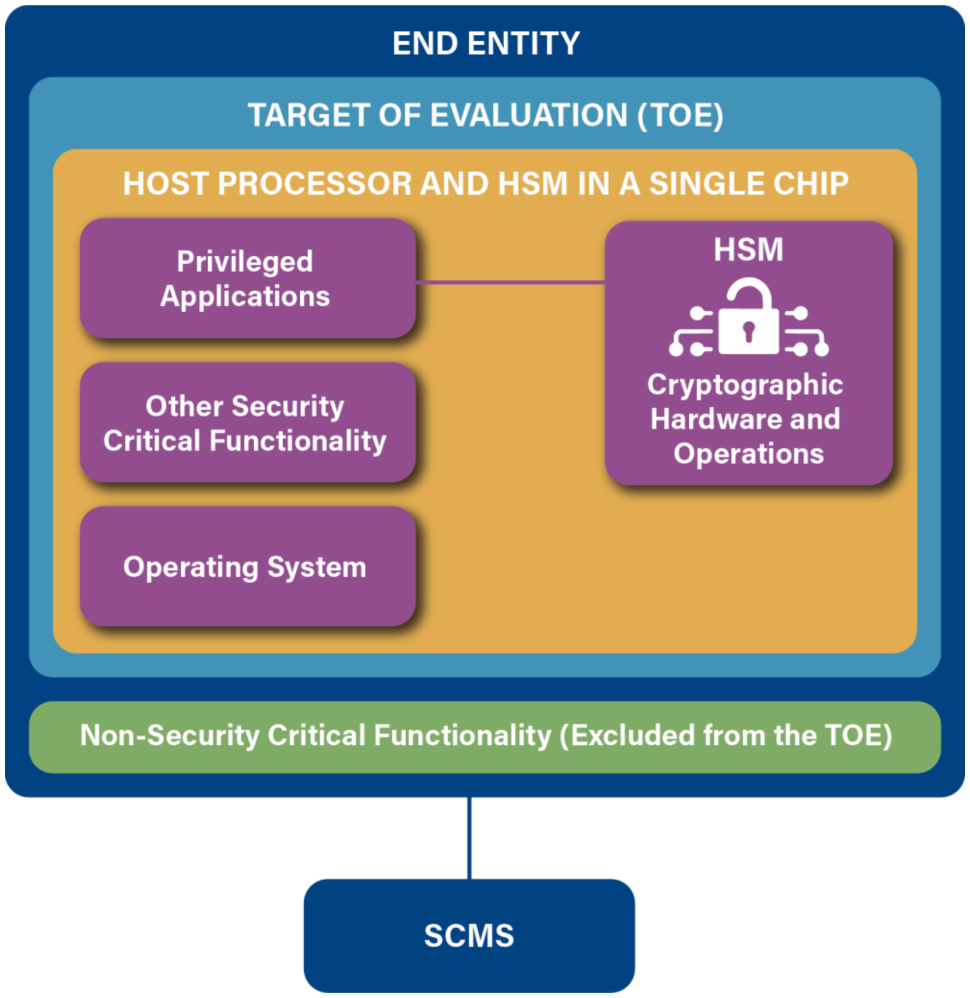
END ENTITY
TARGET OF EVALUATION (TOE)
HOST PROCESSOR AND HSM IN A SINGLE CHIP
Privileged Applications
Other Security Critical Functionality
Operating System
HSM
Cryptographic Hardware and Operations
Non-Security Critical Functionality (Excluded from the TOE)
SCMS

FIGURE 1. INTEGRATED ARCHITECTURE END-ENTITY TARGET OF EVALUATION

Connected Architecture: An EE conforming to a Connected Architecture is defined as having two distinct processors: one Host Processor executing privileged applications, and a distinct HSM implementing hardware-protected key storage and 1609.2-based signing. In this architecture, the Host Processor is connected to the HSM in such a way that the connection is exclusive to the Host Processor, i.e.no other processors have access to the interface between them. The TOE for a Connected Architecture includes the HSM, Host Processor, Dedicated HSM Interface (between the Host Processor and HSM) and SCMS Connection between the Host Processor and SCMS.

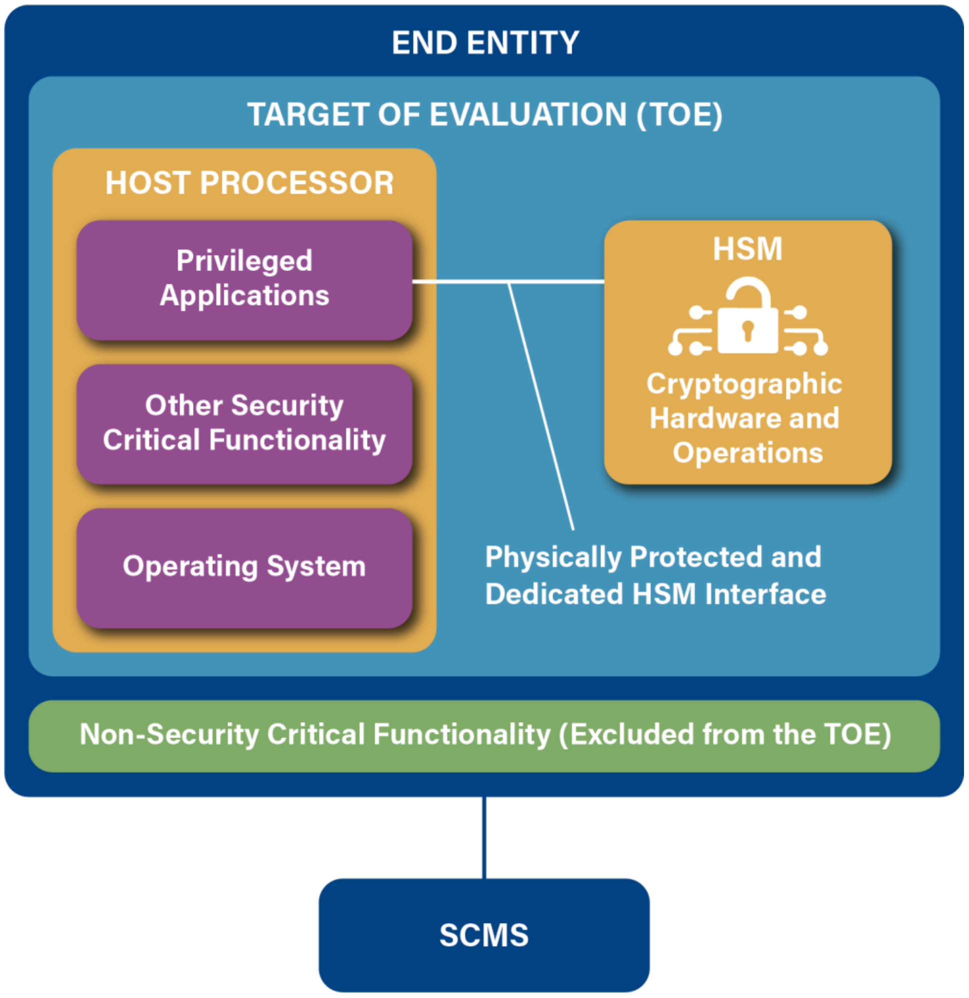
END ENTITY
TARGET OF EVALUATION (TOE)
HOST PROCESSOR
Privileged Applications
Other Security Critical Functionality
Operating System
HSM
Cryptographic Hardware and Operations
Physically Protected and Dedicated HSM Interface
Non-Security Critical Functionality (Excluded from the TOE)
SCMS

FIGURE 2. CONNECTED ARCHITECTURE END-ENTITY TARGET OF EVALUATION

Networked Architecture: An EE conforming to a Networked Architecture is defined as having two distinct processors: one Host Processor executing at least one privileged application, and a distinct HSM implementing hardware-protected key storage and 1609.2-based signing. In this architecture, however, the Host Processor is connected to the HSM over a connection that may be physically accessed by other processors. The TOE for a Networked Architecture includes the HSM and the Host Processor, which are connected across what may be an untrusted network. The TOE includes assurances that the Host Processor and HSM can communicate in a trusted manner across the connecting network.

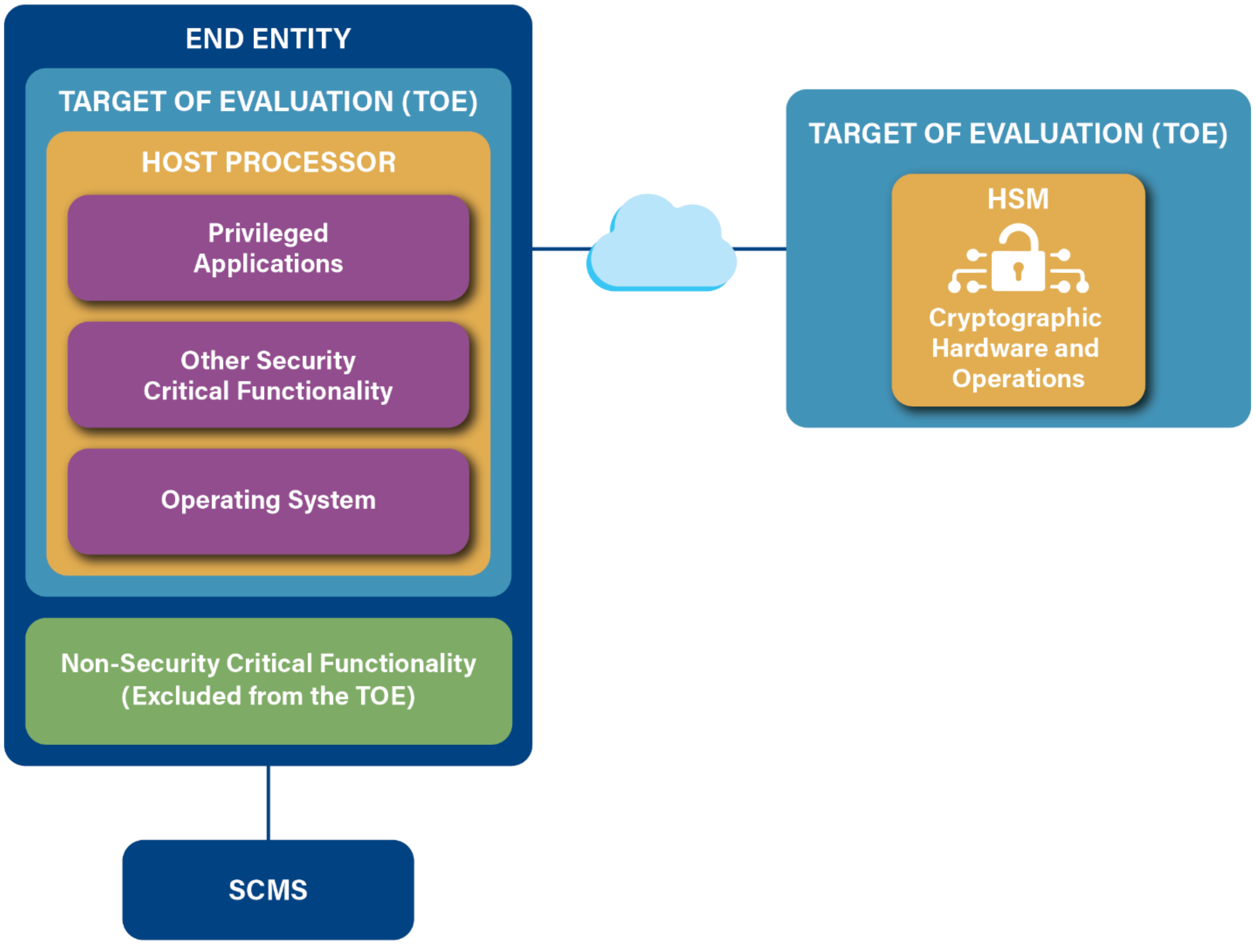
END ENTITY
TARGET OF EVALUATION (TOE)
HOST PROCESSOR
Privileged Applications
Other Security Critical Functionality
Operating System
Non-Security Critical Functionality (Excluded from the TOE)
SCMS
TARGET OF EVALUATION (TOE)
HSM
Cryptographic Hardware and Operations

FIGURE 3. NETWORKED ARCHITECTURE END-ENTITY TARGET OF EVALUATION

4.2 DEVICE CLASSES

Device classes are introduced in the ARC-IT security analysis and are intended to support definitions of security requirements for devices such that devices that run more-critical applications (or, more specifically, that act in more-critical roles within applications) can be subject to more stringent security controls than devices that only run less-stringent applications. For example, a device that is capable of hosting an application to request signal preemption may be required to support user authentication before activating that functionality, while a device that only supports sending Basic Safety Messages may be acceptable without supporting user authentication for those operations. The intent of the class definitions is to create a relatively small set of device types, each of which is suitable for operations of a particular sensitivity level, such that procurement of devices to support application operations can simply reference the device class and not need to determine a collection of application-specific security controls.

This document provides a baseline of security requirements suitable for all devices, including Class 1 as defined in ARC-IT. It is anticipated that security controls and certification for higher device classes will be developed in supplements to this document. Additional information about devices classes can be found at: Device Classes (arc-it.net).

Class 1 devices include those hosting the most common, high profile connected vehicle infrastructure and vehicle-side applications, such as: Signal Phase and Timing (SPaT), MAP, Traveler Information Messages (TIM), Basic Safety Messages (BSMs) per SAE J2945/1, and RTCM.

The class of a device is the class associated with its most sensitive application role. It is not guaranteed that a device that conforms to this document will be considered suitable to run, or to receive certificates for, applications that are higher sensitivity than those listed in the previous paragraph.

4.3 FIPS 140-2 APPLICABILITY

The National Institute of Standards and Technology (NIST) published FIPS 140-3 on March 22, 2019, which supersedes FIPS 140-2. This document refers exclusively to FIPS 140-2 requirements for cryptographic modules, however adherence to equivalent requirements in FIPS 140-3 are acceptable. When referencing FIPS 140-2, specific sections (FIPS 140-2, Sections 4.1 to 4.11) are called out for their relevance to SCMS EE requirements.

A FIPS 140-2 validation certificate is NOT required of an End-Entity to obtain SCMS certification, nor are all IEEE 1609.2 security mechanisms conformant with FIPS 140-2.

FIPS 140-3 is a NIST-tailored version of ISO/IEC 19790:2012(E)/Cor.1:2015(E) and ISO/IEC 24759:207(E). NIST profiled ISO 19790 in special publications SP 800-140, and SP 800-140A through SP 800-140F. The NIST publications are all freely available on their website, whereas ISO publications must be purchased.

If any of the changes in FIPS 140-3 impact SCMS Manager requirements, design guidance, or validation approach, they may be specifically referenced in future versions of this specification. This version of the specification, however, leverages FIPS 140-2 for its widespread availability and ease of reference.

4.4 THE V2X ECOSYSTEM

The successful operation of the ITS system requires trustworthiness among many elements. Entities are today instantiated in three common ITS devices: Onboards Units (OBU), Roadside Unites (RSU) and TMC Appliances. Onboard Units (OBUs) are deployed in vehicles and or as mobile devices and need to trust communications between one another and infrastructure components such as Roadside Units (RSUs).

RSUs are typically mounted along roadways or at traffic intersections and need to trust communications with OBUs and Traffic Management Centers (TMC). Like OBUs, RSUs are generally implemented using Integrated and Connected Architectures, but are not precluded from being implemented in a Network Architecture End-Entity.

A TMC Appliance is an End-Entity generally used in a centralized, protected facility and often conforming to a Networked Architecture (though other architectures are not precluded for TMC Appliances). TMCs using a TMC Appliance may sign centralized V2X messages such as MAP and Traveler Information messages (TIMs).

Future EE device embodiments are expected over time.

In addition to connectivity, software updates and device management activities need to be secured to maintain trust between components. Underlying this trust are private keys and other elements that need to be protected from unauthorized disclosure, manipulation, and misuse.

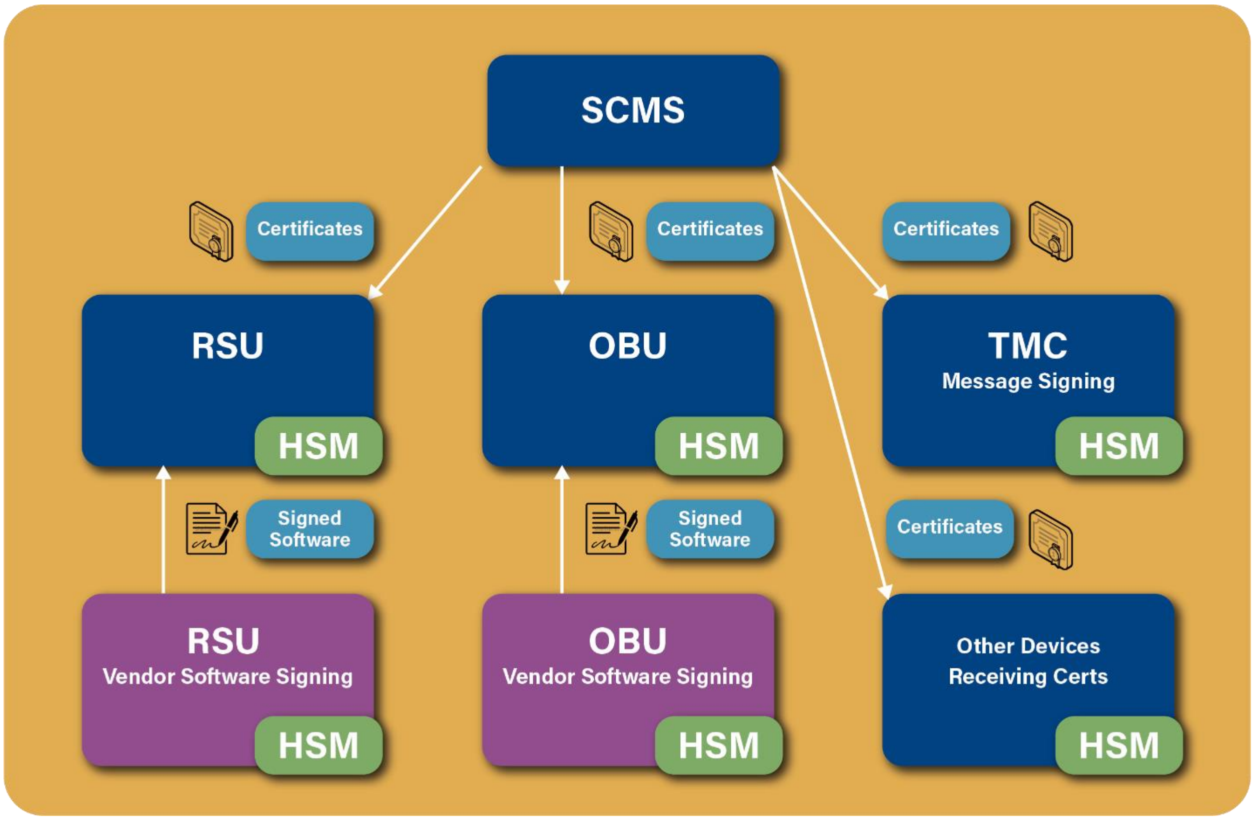

FIGURE 4. V2X ECOSYSTEM

End-Entities embodied in RSUs, OBUs, TMCs and other devices that sign 1609.2 messages shall protect the private key material used to perform those signatures per the requirements of this document.

This document specifies the SCMS Manager conformance requirements for end-entities to be eligible to be provisioned with IEEE 1609.2 certificates from an SCMS Manager-approved SCMS provider. Currently those end-entities are embodied as OBUs, RSUs, and TMC Appliances running applications associated with Class 1 devices in the ARC-IT analysis. As End-Entity embodiments evolve, or as applications with higher security requirements are defined, this document may be updated to address needed changes to security requirements or guidance.

The scope of this specification only includes end entities being provisioned with IEEE 1609.2 certificates for use in V2X message signing or establishment of TLS connections. It does not include security requirements for X.509 certificate issuance.

This specification sets minimum requirements for EEs to receive IEEE 1609.2 certificates and associated security critical materials. It is not intended to restrict manufacturers, integrators, and users of this equipment from implementing additional security measures as appropriate.

As referenced standards are revised, this document will be updated with new profiles and/or guidance, as appropriate.

Requirements related to misbehavior detection are not included in this specification and will be provided in future documents based on application or system level security needs identified in the V2X ecosystem.

The requirements in this document do not apply to EEs exclusively provisioned with test certificates.

5 REQUIREMENTS

This section details the general and architecture-specific End-Entity requirements for devices that use IEEE 1609.2 certificates to sign messages or perform 1609.2-based TLS authentication.

5.1 END-ENTITY IDENTIFICATION

The purpose of End-Entity identification is to provide a unique identifier referencing a specific "Bill of Materials" defining the End-Entity TOE. The purpose is that the identifier and any version information assigned to it points to a validated TOE and any modification of the TOE components triggers an identifier and/or version change.

Certification of an End-Entity shall be unique to a discrete, identified set of hardware, firmware, software, and configuration information within the TOE boundary.

The TOE boundary shall be specified in the documentation. The TOE may be equal to the physical boundary (or boundaries) of the End-Entity, or it may exclude non-security-critical operations. If the TOE excludes non-security-critical operations, the documentation shall explain how the design of the EE ensures that non-security-critical operations can have no impact on security critical operations.

The identification and versioning methodology for the End-Entity shall be documented and provided to SCMS manager as a prerequisite to receiving IEEE 1609.2 certificates. See Section 7 for further discussion of the validation process.

Modification of any component within the TOE boundary shall trigger an identifier or version change, including changes in the areas excluded as non-security-critical.

Modification of any component outside the TOE boundary shall NOT trigger an identifier or version change.

5.2 CRYPTOGRAPHIC SUPPORT

This section details general cryptographic support requirements for the End-Entity. End Entities are expected to utilize cryptography in a range of mandatory and optional protocols.

The requirements of this section do not apply to:

- Excluded non-privileged applications
- Other processors containing ONLY excluded non-privileged applications

5.2.1 MINIMUM CRYPTOGRAPHIC STRENGTHS (GENERAL)

Cryptographic algorithms and key sizes used for security-critical functionality shall support a cryptographic security strength that meets the NIST requirements for 128-bit security for the year 2031 and beyond as discussed in NIST SP800-57.

Security-critical functionality may optionally make use of stronger cryptography (e.g., AES-256 or 384-bit ECC curves).

Non-security critical functionality outside the TOE may use weaker cryptography subject to the condition of section 5.1 that the documentation explains why this functionality cannot affect the security-critical functionality.

5.2.2 IEEE 1609.2 AND IEEE 1609.2.1 SUPPORTED SECURITY FUNCTIONS

The EE shall be compliant with 1609.2a-2017 and guidance note 12.

The EE shall be compliant with 1609.2.1 as profiled by SCMS Manager at Publications – SCMS Manager.

5.2.3 TLS CONNECTIVITY WITH SCMS

TLS1.2 or TLS 1.3 shall be implemented in accordance with IEEE 1609.2.1, Section 5.3 when connecting to the SCMS. Note that IEEE 1609.2.1 specifies server-based certificates for authentication. Client-based authentication is handled separately and therefore there are no private keys to be protected in the End-Entity for TLS in use with connection to the SCMS.

If TLS 1.3 is used to connect to the SCMS, the RFC 8446 session resumption PSK feature shall not be used.

TLS 1.2 is specified in IETF RFC 5246. TLS 1.3 is specified IETF RFC 8446.

TLS 1.2 shall be restricted to support only the following algorithm suites:

ECDHE_ECDSA_WITH_AES_128_CBC_SHA_256
ECDHE_ECDSA_WITH_AES_128_GCM_SHA_256
ECDHE_ECDSA_WITH_AES_256_CBC_SHA_384
ECDHE_ECDSA_WITH_AES_256_GCM_SHA_384

TLS 1.3 shall be restricted to support only the following algorithm suites:
TLS_AES_256_GCM_SHA384
TLS_AES_128_GCM_SHA256

and shall support ECDHE for Key exchange and ECDSA for digital signatures. (TLS 1.3 specifies key exchange and digital signatures separately from the algorithm suites. TLS 1.3 has removed support for CBC mode. Otherwise, the recommendation for algorithm support in 1.3 is the same as 1.2).

5.2.4 OTHER (NON-SCMS) USES OF TLS IN THE TOE

Any other uses of TLS by elements inside the TOE shall use TLS 1.2 or later and shall use a cryptographic suite that meets the NIST 128-bit strength requirement. Authentication may be one-way or mutual based on the requirements of the given application.

If a TLS 1.3 session resumption Pre-Shared Key (PSK) is used, it shall be in accordance with RFC 8446.

5.2.5 OTHER SECURE PROTOCOLS AND ALGORITHMS

This section applies to cryptographic protocols and algorithms used by applications within the TOE.

EEs may employ a variety of protocols for communication with external systems. The requirements in this section apply to communications that may affect critical security functionality. This can include device management, configuration, and software update. The documentation accompanying an EE shall identify communications between a process inside the TOE and a process outside that might affect critical security functionality and shall demonstrate that all those communications meet the following requirements.

All connections with processes outside of the TOE (“external processes”) that may affect security critical functionality shall use cryptographic security controls using conformant with the strength requirements of 5.2.1.

Connections with processes outside of the TOE that do not affect security critical functionality are permitted not to meet these requirements, so long as it is explained why these connections cannot affect security critical functionality.

If a connection with a process outside of the TOE may affect security critical functionality, the EE shall require authentication of the external process. This authentication may be password-based or may be based on public keys or certificates. The authentication mechanism shall establish that the external process has rights to modify functionality on the specific EE device.

If a connection with a process outside of the EE may affect security critical functionality, the EE shall authenticate or identify itself to the external process. This authentication may be at the protocol level (e.g., TLS mutual authentication) or at the application level (e.g., signing inside the application). The authentication shall involve a cryptographic operation using algorithms conformant with the strength requirements of 5.2.1.

5.2.5.1 SSH

This section only applies if Secure Shell is used.

For communications that may affect critical security functionality, SSH-2 shall be implemented in accordance with IETF RFC 4253 and the following configuration:

- cipher=aes128-gcm@openssh.com
- mac=hmac-sha2-256@etm@openssh.com
- kex=ecdh-sha2-nistp256
- key=ecdsa-sha2-nistp256

Other algorithms or key sizes conformant with the strength requirements of 5.2.1 are also acceptable.

SSH version 1, or configurations of SSH v2 other than those identified above, shall not be used for communications that may affect critical security functionality. If any process inside the TOE uses a non-approved version or mode of SSH, this may be done subject to the condition of section 5.1 that the documentation explains why this functionality cannot affect the security-critical functionality.

5.2.5.2 IPSEC

This section only applies if IPSec is used.

IPSEC shall be in accordance with IETF RFC 4301 and configured as follows:
- AES-GCM-128 for confidentiality and authentication
- ECDH-256 for key exchange
- ECDSA-256 for authentication

Other algorithms or key sizes conformant with the strength requirements of 5.2.1 are also acceptable.

SSH version 1, or configurations of SSH v2 other than those identified above, shall not be used for communications that may affect critical security functionality.

5.2.5.3 MESSAGE AUTHENTICATION CODES

Message Authentication Codes (MACs) to support security critical functionality (e.g. HMAC, CMAC) shall be implemented in accordance with NIST SP 800-38x. The underlying cryptographic algorithm shall be any algorithm conformant with the strength requirements of 5.2.1. For example, HMAC using SHA-256 complies with this requirement.

5.2.6 KEY STORAGE

This section applies to key storage performed by applications within the End-Entity.

IEEE 1609.2: All IEEE 1609.2 private keys shall be stored and operated only in processors that meet the HSM security requirements specified in this document.

MAC: MAC keys used for device integrity checks shall be EITHER stored and operated in processors that meet the HSM security requirements or equivalent protection.

MAC keys used to authenticate secure network connections shall be stored and operated in the Host Processor or in a processor that meets the HSM security requirements.

TLS, SSH, and IPSEC: The EE shall either:

- Store and use all secret and private material within the HSM (preferred), or
- Store private key material within the HSM when not in use, allow private and secret material to be used in the host processor when needed, zeroize any keying material when not in use, or
- Store private key material in a keystore managed by the host processor and integrity protected by the HSM, other secure hardware or a mechanism demonstrated to be equivalent to hardware-based protection.

IEEE 1609.2 TLS Private Key Material: Private keys pairwise with IEEE 1609.2 based TLS certificates shall only be stored within the HSM.

5.2.7 RANDOM NUMBER GENERATOR

All hardware and software-based random number generators in the End-Entity used to support security critical functionality shall use a random number generator from the list of approved random number generators in FIPS 140-2 Annex C or random number generators evaluated using AIS 20/31 as described by BSI.

5.3 HOST PROCESSOR REQUIREMENTS

This section lists the requirements for the Host Processor.

5.3.1 MAINTENANCE AND OPERATIONAL STATES

The Host Processor *operational state* shall be defined as an integrity-protected EE mode of operation in which all of the requirements of Sections 5.3.2 to 5.3.5 apply.

A Host Processor *maintenance state* shall be defined as an EE mode of operation in which one or more requirements from Sections 5.3.2 to 5.3.5 are not met.

The *maintenance state* may apply to EEs in (or returned to) the manufacturing environment or EEs undergoing post-manufacturing maintenance operations.

A device may be designed so it can transition from the *operational* state to the *maintenance state*:

If this functionality requires authentication of an authorized role, the state transition:

- may be procedural or automatic
- if procedural, the transition process shall 1) require the user to manually wipe all data from privileged applications from the Host Processor and all associated secret and private keys from the HSM, AND 2) be clearly and unambiguously described in the manufacturer's instructions

If this functionality does not require authentication of an authorized role, invocation of the state transition:

- shall ensure the entity initiating the state transition is physically present
- shall automatically wipe all data from privileged applications from the Host Processor and all associated secret and private keys from the HSM

When in the operational state, any probing or debugging capabilities (such as JTAG), shall be disabled. This disablement may be permanent or temporary. If temporary, it may be re-enabled by going back into the maintenance state (where all sensitive information has been wiped), by a password mechanism that is unique to the device and is of sufficient strength to protect the End-Entity over its expected deployment lifetime, or by a cryptographic mechanism that is unique to the device and invocation (not subject to replay).

5.3.2 SECURE BOOT

The Host Processor shall perform integrity checks on boot to ensure that it is in a known good software state. The integrity checks shall require the use of a protected value such that the integrity cannot be successfully compromised unless the protected value is modified. The value shall be protected by the HSM or hardware-equivalent protection. The design guidance section will provide options for achieving this protection using techniques such as irreversible functions when such approaches have been vetted.

All parts of the End-Entity within the TOE shall be subject to the integrity check.

Examples of these integrity checks include:

- signing the Host Processor Operating System and Application software such that the protected value described above is the verification key. In this example the verification key may be protected by the HSM or "burned" into the Host Processor.
- storing integrity values (e.g., hashes or HMAC values) via the Platform Configuration Registry (PCR) mechanism of the Trusted Computing Group's (TCG) Trusted Platform Module (TPM) on the Host Processor

The End-Entity shall ensure that privileged applications can only operate or access any 1609.2 signing private key operations if the Host Processor Integrity Check has successfully completed.

The Host Processor integrity check shall ensure that any root CA certificates, elector certificates, or other authenticators critical to the security of the EE or its applications have not been modified since they were last accessed.

Any incoming IEEE 1609.2-signed messages that do not pass all the integrity checks shall be dropped and logged.

The End-Entity shall only transmit IEEE 1609.2-signed messages that are known to be valid. If created locally and know to be valid, then no additional checks are required. If the message is received from an external source, the message shall be validated before being transmitted.

5.3.3 OPERATING SYSTEM

The operating system (OS) shall meet the following requirements (derived from FIPS 140-2 section 4.6.1):

- The operating system shall support roles, which are used as specified below. Each privileged application shall map to a role, and additional roles may exist as necessary to support the functionality specified below.
- The discretionary access control mechanisms of the operating system shall be configured to:
    - Specify the set of roles that has execute permissions on each private key stored within the HSM.
    - Specify the set of roles that can modify (i.e., write, replace, and delete) programs and plaintext data stored at specific locations within the Host Processor boundary.
    - Specify the set of roles that can read data stored within the Host Processor boundary and what data can be read by those roles.
    - Specify the set of roles that can enter cryptographic keys (It is permissible for the host to require that all keys are generated on the device and that keys cannot be entered directly).
- The OS shall allow the following roles to operate without explicit authentication by a user:
    - Processes that correspond to privileged applications, i.e., applications that are intended to run without user initiation or intervention, and that have execute access to IEEE 1609.2 private keys.
    - Processes that update or derive IEEE 1609.2 private key material to or in the HSM, i.e., to implement the butterfly key process specified within the SCMS documentation.
- The OS may allow the following roles to operate without explicit authentication or may require authentication:
    - Processes that install new software or firmware if that software or firmware is signed.
    - Processes that write private key material to the HSM (It is permissible for EE to require that all keys are generated on the device and that keys cannot be entered directly).
    - Processes that validate the CTL signature and update IEEE 1609.2 certificate chains.
- The OS may support the following roles and, if it supports them, shall require explicit authentication:
    - Processes that modify or inspect executing processes.
- The OS shall not allow the following roles to exist:
    - Processes that read - or allow to be read - private cryptographic key material from the HSM (NOTE: The HSM also shall NOT provide this functionality).

5.3.4 SECURE UPDATES

The Host Processor shall use the following mechanisms to ensure that its software and firmware can be securely updated:

- All software shall be digitally signed.
- When requested to install software, the Host Processor OS shall ensure that software updates are signed by an authorized entity before proceeding with the installation
- The Host Processor shall reject the installation if the signature or any of the validity checks on the software or its signing certificate fail.
- The software update verification key shall be integrity protected within the HSM or equivalent protection. The update mechanism shall provide software roll-back protections.

5.3.5 INTEGRATED ARCHITECTURE

The Host Processor and HSM shall be the same processor protected within a *physical boundary* that:

- is physically contiguous,
- meets the physical security protections defined in FIPS 140-2, Section 4.5 for a Level 3 cryptographic module

The physical security for an Integrated Architecture is depicted in Figure 5.

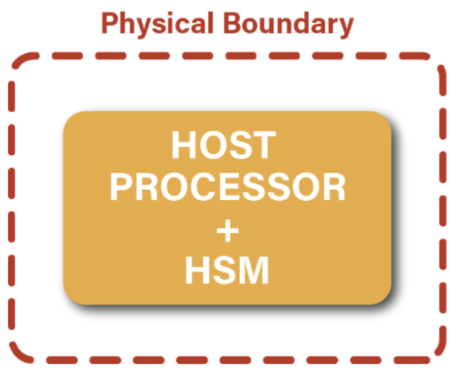

FIGURE 5. HOST PROCESSOR PHYSICAL BOUNDARY FOR INTEGRATED ARCHITECTURE

5.3.6 CONNECTED ARCHITECTURE

The Host Processor and HSM in a Connected Architecture shall be different processors connected by a dedicated, protected physical interface to which no other processor has physical or logical access.

The Host Processor shall be protected as follows:

- Within a single, physically contiguous *physical boundary* that:
    - meets the physical security requirements of FIPS 140-2, Section 4.5 Level 3
    - physically envelopes the Host Processor, HSM AND the physical connection between them,

- prevents direct or indirect HSM access by any external processor (outside of the *single processor physical boundary*). This does not preclude the enclosure having a maintenance access, as long as that does not permit access to the HSM or its connection to the host processor.

The physical boundary protection options for an EE implemented with a Connected Architecture are depicted in Figure 6.

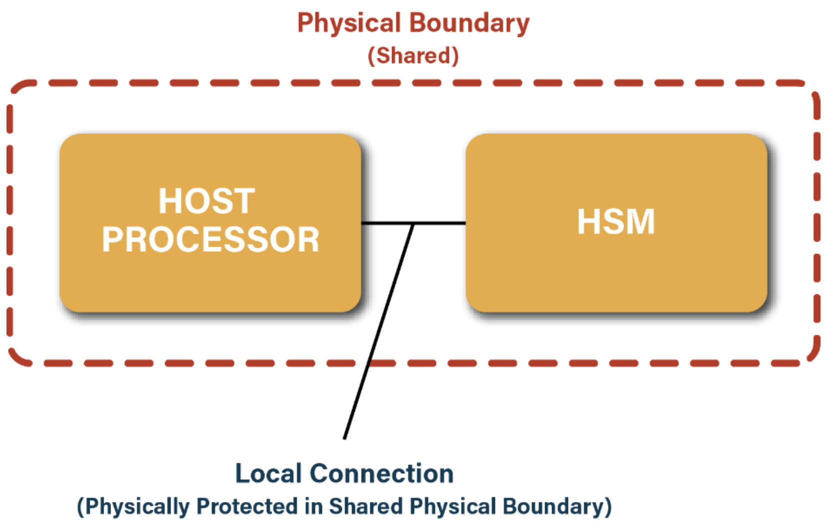

FIGURE 6. CONNECTED ARCHITECTURE PHYSICAL BOUNDARY PROTECTION

5.3.7 NETWORKED ARCHITECTURE

The Host Processor and HSM in a Networked Architecture shall be different processors connected over a non-dedicated network in which some or all of the network pathways between the Host Processor and HSM may be physically unprotected.

The Host Processor shall implement *secure hardware* meeting the requirements of FIPS 140-2, Section 4.5 at a level commensurate with the networked HSM.

The Host Processor shall mutually authenticate itself to the networked HSM using an authentication mechanism based in the *secure hardware* in order to utilize its cryptographic services of the networked HSM.

The physical security boundaries for an EE in the networked architecture are depicted in Figure 7.

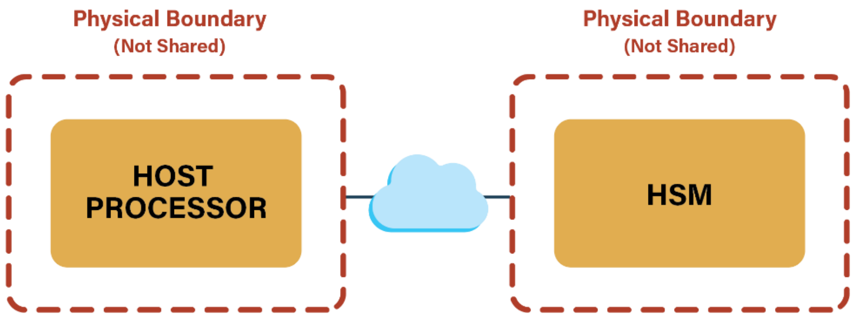

FIGURE 7. PHYSICAL SECURITY BOUNDARIES FOR THE NETWORKED ARCHITECTURE

5.4 HSM REQUIREMENTS

This section lists the requirements for the HSM.

5.4.1 OPERATING SYSTEM

This section is only applicable to HSMs with an operating environment meeting the FIPS 140-2 Section 4.6 definition of '*modifiable operational environment*.'

The HSM shall meet the requirements for an operating system given in FIPS 140-2, Section 4.6 Level 2 except for the audit requirements and certain additional exceptions. The baseline requirements are the following:

- All cryptographic software and firmware shall be developed and installed in a form that protects the software and firmware source and executable code from unauthorized disclosure and modification.
- A cryptographic mechanism using an approved integrity technique (e.g., an approved message authentication code or digital signature algorithm) shall be applied to all cryptographic software and firmware components within the HSM.
    - The message authentication code may be used in the following circumstances only:
        - If the HSM itself calculates the MAC when the software is installed using a secret key known only to the HSM and uses this secret key to verify the software on boot.
        - If the software or firmware provider has a unique shared key with each distinct device and uses this to authenticate the software.

A Message Authentication Code (MAC) shall not be used to protect the software unless the MAC key is unique to the HSM.

- All cryptographic software and firmware, cryptographic keys, and control and status information shall be under the control of an operating system that meets the functional requirements specified and is capable of evaluation at the CC evaluation assurance level EAL2, or an equivalent trusted operating system.
- To protect plaintext data, cryptographic software and firmware, cryptographic keys, and authentication data, the discretionary access control mechanisms of the operating system shall be configured to:

    - Specify the set of roles that can execute stored cryptographic software and firmware.
    - Specify the set of roles that can modify (i.e., write, replace, and delete) the following cryptographic module software or firmware components stored within the cryptographic boundary: cryptographic programs, cryptographic data (e.g., cryptographic keys and audit data), and plaintext data.
    - Specify the set of roles that can read the following cryptographic software components stored within the cryptographic boundary: cryptographic data (e.g., cryptographic keys and audit data), and plaintext data.
    - Specify the set of roles that can enter cryptographic keys.
- The discretionary access control mechanisms may allow a role without explicit authorization to create a new cryptographic key by combining an existing key with new input if the device follows the Integrated or Connected Architectures. The discretionary access control mechanisms shall require explicit authorization to create a new cryptographic key by combining an existing key with new input if the device follows the Networked Architecture.
- The discretionary access control mechanisms may allow a role without explicit authorization to execute stored cryptographic software and firmware if the device follows the Integrated or Connected Architectures. The discretionary access control mechanisms shall require explicit authorization to execute stored cryptographic software and firmware if the device follows the Networked Architecture.
- The discretionary access control mechanisms of the OS may allow automated software and firmware update if that update is carried out by a process that includes cryptographic checks to ensure the validity of the update prior to installation.
- The operating system shall prevent all operators and executing processes from modifying executing cryptographic processes (i.e., loaded and executing cryptographic program images). In this case, executing processes refer to all non-operating system processes (i.e., operator-initiated), cryptographic or not.
- The operating system shall prevent operators and executing processes from reading cryptographic software stored within the cryptographic boundary.

5.4.2 HARDWARE PROTECTION

All HSMs shall store keys within a physical boundary meeting the requirements defined in FIPS 140-2, Section 4.5 for a Level 3 cryptographic module.

5.4.3 ADDITIONAL REQUIREMENT FOR NETWORKED ARCHITECTURE

If the HSM is connected to its Host Processor over a shared bus or external network as described in "networked architecture" in section 1.1 above, then it shall provide a mutual authentication mechanism between itself and the Host Processor such as a mutually authenticated TLS connection.

Mutual authentication shall be successful prior to the HSM providing the Host Processor any cryptographic or other security-related services.

5.5 OTHER PROCESSORS

This section stipulates requirements for to other processors resident in the EE, if any.

Other processors: ‘Other processors’ shall be defined as processors that are within the physical boundary of the End-Entity AND consist only of excluded non-privileged applications.

Other processors may be under the control of the Host Processor Operating System or may be under the control of another Operating System or virtualization layer within the End-Entity boundary.

Software running on other processors shall be subject to the secure boot requirements of Section 5.3.2, as follows:

- Secure boot on other processors may be managed by the Host Processor, or may be managed by a separate process governing the other processor
- If secure boot of the other processor is managed separately from the Host Processor, privileged applications running on the Host Processor EE shall not be permitted to perform IEEE 1609.2 signing operations if the secure boot of the other processor fails

If the Operating System of the other processor is under the control of a *modifiable operational environment* – as defined in FIPS 140-2, Section 4.6 – it shall meet the requirements of Section 5.3.3.

5.6 REQUIREMENTS FOR A SECURE ENVIRONMENT FOR DEVICE ENROLLMENT

All end-entities that participate in the SCMS need to be enrolled with an Enrollment Certificate before they can be provisioned with application EE certificates. The enrollment process is the point where an initial trust relationship is established between a new End-Entity and the SCMS infrastructure. The integrity of the system requires that only trusted devices are allowed to enroll, and that each End-Entity receives a valid Enrollment Certificate for interacting with the SCMS.

The enrollment process shall be performed in a secure environment using an approved process and equipment. Delegation of these processes to a supplier is acceptable provided that the security requirements are met.

5.6.1 ARCHITECTURE

The secure environment used for device enrollment shall require the following elements:

1. A documented procedure for performing the enrollment process.
2. A physically secure location where the enrollment will take place.
3. One or more authorized devices (computers) for managing the enrollment process.
4. An activity log or recording of the enrollment operations that were performed.

5.6.1.1 DOCUMENTED PROCEDURE

The procedure used to enroll devices shall be documented and followed consistently. It is recommended that a checklist or automated procedure be used to ensure consistency and compliance. The procedure shall include documentation for handling the following cases:

1. List of Authorized Enrollment Technicians and Equipment
    a. An SCMS Enrollment responsible party shall ensure that only authorized personnel and equipment may participate in the enrollment and provisioning process.
    b. The means of authorizing, onboarding, and removing personnel and systems shall be specified.
    c. The list of authorized personnel and equipment shall be auditable.
    d. If enrollment procedures are performed by a secondary (e.g., contracted) entity, the entity shall conform to all the requirements above.
2. Acceptance of a new End-Entity
    a. Authorized Enrollment Technicians shall be able to validate that the End-Entity, that is to be enrolled, is an authentic device. For example, this may be done by checking the device serial number against a manifest or by inspecting key features of the devices.
    b. The Authorized Enrollment Technicians shall ensure that the EE is in a known good condition. For example, they may ensure that the device was manufactured in a good state and has not been modified since manufacture. This may include the use of tamper evident packaging on the device, in which case the operators shall inspect the tamper seals to ensure they have not been compromised. The software or firmware installed in the End-Entity shall be checked to confirm that it is running an allowed version. It is recommended that a secure hash be performed over the installed software be checked against a trusted reference to validate that it has not been modified.
    c. Authorized Enrolment Technicians shall confirm that the End-Entity has successfully completed its self-test.
    d. Refer to PCI HSM Security Requirements version 3.0 (June 2016), Section I (Device Security Requirements During Manufacturing) for additional guidance on validating the End-Entity to be provisioned.
3. Connection to the End-Entity
    a. During the bootstrapping process, certain information needs to be communicated with strict integrity controls. The procedure shall describe how an operator (or automated process) validates that a secure connection has been established to the new End-Entity. For example, a physical cable connection that can be visually inspected is acceptable.
    b. If a wireless connection is to be used, the procedures shall describe how the connection to the End-Entity will be secured. This connection shall provide authenticity and secrecy and prevent against replay of old, valid messages. Standard protocols may be used if their authentication and encryption mechanisms meet these requirements.
4. Key Generation or Injection

    a. The enrollment process requires that each End-Entity generate or receive a private key and the corresponding public key. This procedure shall be initiated and completed in the secure environment and follow the 'level 2' requirements defined in FIPS PUB 140-2 Section 4.7 for key generation and secure key management.
    b. The association of the device public key to the End-Entity shall be verified. It is recommended that the private keys for the Certificate Signing Request (CSR) be generated on the target End-Entity and exported using the secure connection established in 3 above. If alternative approaches are used, they shall be procedurally documented to ensure that the private key used to generate the CSR is correctly associated with the End-Entity and that no copies of the private key are retained outside the End-Entity.
5. Certificate and Parameter Installation
    a. The enrollment process requires the installation of one or more root CA certificate and elector certificates into the End-Entity's secure storage. This installation shall be performed in the secure environment using the high-integrity communications channel established in Item 3 above.
6. Creation of an Activity Log
    a. The documented procedure shall describe the steps that shall be taken to log or record the enrollment process. Note that the log may not include any private keys or seeding material used to initialize any device.
7. Exceptions and Changes
    a. The procedures shall define what steps are to be taken in case of an error or failure. This should include guidelines for repair or secure decommissioning of failed equipment.
    b. Changes or exceptions to the enrollment procedure shall be recorded.

5.6.1.2 SECURE ENVIRONMENT

The enrollment process shall take place within a physically secure location with restricted access control. Alternatively, the procedures may be carried out in an open area with active monitoring or surveillance to ensure that only authorized individuals and equipment are involved. Refer to the PCI Physical Security Requirements version 2 (Nov 2016) Section 3 for guidelines for establishing a physically secure area for secure provisioning.

- Only authorized personnel shall be able to initiate the enrollment process or have access to the equipment used for enrollment.
- Only authorized equipment shall be connected (wired or wireless) to any network, system, RSU, or OBU involved in the enrollment process.
- The access control mechanism (or area monitoring) shall include a log wherein a record is kept and protected of who is present in the area during enrollment processes.

5.6.1.3 AUTHORIZED EQUIPMENT

Only specific, authorized equipment shall be used in the enrollment process. This equipment may include one or more general-purpose computers.

- The equipment shall not be used for any purpose other than End-Entity enrollment or related logging, testing, or quality control procedures.
- This equipment shall operate on a network segment that is protected from other general-purpose systems used for any other purpose.
- Only authorized personnel may access the equipment or install software, updates, or patches to the equipment. All approved and validated security patches shall be applied to all authorized systems.
- The operating system and application software shall be specified in the documents described in 5.6.1.1.

5.6.1.4 AUDIT AND ACTIVITY LOG

The ability for independent auditors to observe a secure process in real-time as well as logs that can be used to reconcile events or audit procedures later are both required to ensure accountability and to recover from newly emerging threats. The secure environment shall support process oversight in the following ways:

1. Each enrollment location shall maintain a log that records the results of the steps defined in section 5.6.1.1. It shall be possible to reconcile enrollment activity against a list of authorized, operational end-entities along with any securely scrapped or in-repair units to account for the final destination of all successfully enrolled devices.
2. Authorized and identified independent auditors shall have access to the secure environment in order to periodically supervise and inspect the ongoing procedures. Auditors shall not directly view or record any secret information such as private keys or random number seed values.

5.7 RSU SPECIFIC REQUIREMENTS

RSUs shall meet the requirements of the RSU Standard 1.0 and NEMA TS-10 as described below.

5.7.1 RSU STANDARD 1.0 REQUIREMENTS

The RSU shall meet the security requirements of section 3.3.5 and the design details of 4.4.5 of the ITE RSU Standard 1.0.

5.7.2 NEMA TS 10-2019 REQUIREMENTS

With the exception of the items below the TS 10-2019 security requirements are consistent with those of the ITE RSU Standard 1.0 and those of this document.

In [RS1], [RS2], [RS12], [DS97], and [DH26] reference is made to an HSM. In [RS1] and [DH26] it specifically states that the HSM shall be FIPS 140-2 level 3 compliant. Full FIPS 140-2 level 3 compliance is not required, only the specific sections of the FIPS 140-2 standard identified in this document. Furthermore, FIPS validation by a NIST-accredited Cryptographic Test Laboratory (CTL) is not required. It is likely that this text intended to state that the HSM must comply with the physical security requirements of FIPS 140-2. And this is consistent with this document and the ITE RSU Standard 1.0. [This specification recommends at this time that vendors obtain a letter from CTL that FIPS 140-2 physical security requirements are met. In the future, this may become a mandatory requirement for EEs to be provisioned IEEE 1609.2 certificates].

[S7] and [RS19] mentions detecting and reporting OBU misbehavior but the document does not provide a design. While this is important, this document levees no specific requirements in this area. SCMS Manager plans to provide separate requirements for misbehavior detection and reporting which will be published at Publications - SCMS Manager when available.

5.8 SOFTWARE SIGNING REQUIREMENTS

Previously requirements were listed for end-entities to validate digital signatures on software before installing it. This means that, at the back end, there must be a system for applying those digital signatures.

Back-End systems for software signing shall use a FIPS140-2 level 3 validated device for the protection of keys used for software signing. Signing keys must be of at least 128-bit strength (3072-bit for RSA, 256-bit for ECDSA). If the hardware only supports 2048-bit RSA, that is acceptable at this time. NIST guidelines state that 2048-bit RSA is acceptable through the year 2030. This specification requires that the applicable FIPS validation certificate number be presented as evidence for the software signing HSM(s).

In addition, policies and procedures shall be in place to assure that only authorized software is signed. Vendor specific methods are permitted for providing this protection. For those seeking guidance the Uptane secure software update system (https://uptane.github.io) is one such approach.

5.9 TRAFFIC MANAGEMENT CENTER (TMC) SPECIFIC REQUIREMENTS

If the TMC (or other similar back-end systems) signs messages using IEEE 1609.2 certificates (such as MAP, RTCM, or RSM messages), then it needs to provide protection for its signing keys. The TMC shall use a FIPS 140-2 level 3 validated HSM. The HSM shall require authentication for each signature performed. This specification requires that the applicable FIPS validation certificate number be presented as evidence for the TMC signing HSM(s) [component of a TMC Appliance]. Note that an HSM utilized in a TMC Appliance need not be operated in FIPS mode due to certain disparities between IEEE 1609.2 and FIPS requirements.

In addition, policies and procedures shall be in place to assure that only authorized message data is signed. Given that there are a variety of ways that a TMC may set up back-end message signing (on site, at a vendor site, etc.), no specific requirements beyond the FIPS140 device are leveed here. The operator of the message signing device shall provide documentation to SCMS Manager of its procedures and policies that are in place to protect against misuse of the back-end message signing keys.

6 DESIGN GUIDANCE

6.1 SINGLE APPLICATION HOST PROCESSOR

The requirements in 5.3 on the Host Processor assume that the processor is running an operating system with multiple applications. Those requirements are designed to protect the applications from interfering with each other or making improper use of another application's keys. Those requirements can be substantially simplified if there is only one application on the End-Entity because protections are implicit.

6.2 IMPLEMENTING AN INTEGRATED ARCHITECTURE HSM

SCMS Manager recommends using SAE J3101 to meet the HSM requirements of the integrated architecture described in section 2 above. What follows is a profile of J3101 to modify and/or clarify some of the requirements of the standard. SAE J3101 is available for purchase from sae.org.

As listed in J3101, section 6, requirements are written for the following: 1. Cryptographic key protection 2. Crypto algorithm 3. Random number generator 4. Secure nonvolatile data 5. Algorithm agility 6. Interface control 7. Secure execution environment 8. Self-tests

Unless otherwise noted below, the requirements of J3101 are recommended as written. The section numbering and naming used in the following sections correspond to those sections in J3101.

Section by Section Modifications/Clarifications

6.2.3.2.2 Short-Term Keying Material Protection: Required. While it is preferred for all short-term keying material to be protected by the hardware protected security environment, an exception will be permitted for TLS session keys. This applies to all three requirements in 6.2.3.2.2 (_20, _30, _10).

6.2.3.2.4 Asymmetric Key Protected Storage: Required. While it is preferred for all private keys to be protected by the hardware protected security environment, an exception will be permitted for TLS private keys. If TLS is implemented using 1609.2 certificates, the private keys for these certificates must be stored in the HPSE.

6.2.3.4.2_50 Clarification. If the processor has the ability to perform the software integrity check in the background, that is an acceptable approach. Some mechanism shall be employed to ensure that the integrity check is performed at least once per day. If the processor is capable of performing the integrity check in the background, that would satisfy this requirement.

6.2.3.5 Clarification. Regarding the last note on the use of OTP memory. Operational secret/private keys (such as 1609.2 keys, TLS keys, etc.) shall not be stored in OTP (One time programmable) memory. It must be possible to remove those keys when they are no longer in use. If the processor uses OTP to store a memory storage (or similar) key that is acceptable as long as that key is internal to the chip and not externally readable. Otherwise, OTP should not be used for storing private/secret keys.

6.2.3.7.6 Clarification. Section 5.3.1 above, defines the requirement for how the device can zeroized – through either an authenticated command or by a user this is physically present at the device. When an EE is being removed from service, any enrollment certificates shall be erased, so that the device can no longer request certificates using its old identity. The device will need to go through a new enrollment procedure appropriate to its new service location. For example, an OBU in an ambulance equipped with the ability request signal preemption shall have those associated materials removed before being placed in service elsewhere. Similarly, an RSU taken outside of its enrollment/application certificate region it shall have its application and enrollment certificates erased before being placed in service elsewhere.

6.2.3.7_60 Clarification. In the V2X ecosystem, the root of trust is distributed among five electors instead of being a single root of trust. Therefore, in places where J3101 refers to protecting the 'root of trust' which, in some systems, may be a single public key, in the V2X ecosystem it is five public keys. The protocol for managing those public keys is described in IEEE 1609.2.1. The hardware protected security environment shall maintain the integrity of the five elector roots of trust throughout the life of the hardware protected security environment. The integrity of these roots shall be checked as part of the processor integrity check (see 6.2.3.4.2_50 and its clarification for more details about integrity checking). See clarification on 6.2.4.2 for requirements for updating those elector roots.

6.2.4.2 Clarification. The hardware protected security environment shall have the ability to securely replace elector roots based on a properly signed CTL as specified in SCMS Manager Elector Technical Specifications published at Publications – SCMS Manager.

6.3.2.5 In this section there is a reference for ANSI X9.62 for the implementation of ECDSA. Note that this standard includes checks that must be performed as part of a signature validation. Those checks are required. Specifically: 5.4.2 requires that r' and s' be checked to be in the proper range. 5.4.3 requires that a particular computation not be the point at infinity. 5.4 mentions the need for a valid public key. If the public key has not already been validated, then the procedure in 5.2.2 must be used to validate the public key before use.

6.3.2.1 Secure Hash Function: Shall support SHA-256. Shall support SHA-384.

6.3.2.2 Encryption Algorithms: Shall support encryption algorithms specified by IEEE 1609.2, TLS, SSH, SCP, and SNMP. Protocols shall use AES-128 (or greater if desired or required for interoperability) and encryption modes as specified in those protocols.

6.3.2.4 Signature algorithms: RSA and DSA are not required but permitted. The supported minimum key size shall be 3072. (If the hardware only supports 2048-bit RSA or DSA, that is acceptable at this time. NIST guidelines state that 2048-bit RSA is acceptable through the year 2030.) Both 256 and 384 ECDSA are required. NIST curves required.

6.6.2.2._10: Clarification. This levies no additional requirements. The requirement that all software loaded onto the End-Entity device have a valid digital signature covers this.

6.6.2.2_40: Clarification – The firmware update public key shall not be updated until it has been authenticated as part of the new firmware update. There have been security failures in the past where, for example, the public key gets replaced as part of the firmware update even if the update fails. In some ways this seems straightforward, but it can a bit complex to protect against attack strategies like interrupting power in middle of an update which would leave the hardware security environment in a compromised state.

6.8.2.6 Resistance to Timing Attacks is Required. Private key operation timing shall not be correlated with the value of the private key. Secret key operation timing shall not be correlated with the value of the secret key. Resistance to other side channel attacks (power, radiation, etc. are optional. The principle at work here is that any attack that can be implemented remotely should be defended against. Side channel attacks that require physical access to the device are less of a concern.

6.3 DESIGN GUIDANCE FOR OPERATING SYSTEMS

In section 2.2.3 requirements were established for the handing roles and the privileges of applications. This section provides guidance on meeting those requirements.

Examples of roles (which generally correspond to applications with the listed privileges):

- A role that signs messages – it shall have access to execute (sign) with the private key but shall not have access to read/write/delete access to the private key.
- A role that is responsible for installing the public key roots of the electors. This role shall have read/write/delete access. All other roles in the device shall only have read access.
- A role that is responsible for firmware/software update. This role shall have read/write/delete access to program storage. All other roles shall have read only access.

The primary principle is that each role (as embodied in an application) shall have access only to those resources that are necessary to accomplish the tasks assigned to that role. The operating system shall be responsible for ensuring that the role cannot access resources beyond those required for its task(s).

The preferred way to support those restrictions is for the processor to have an MMU (memory management unit) that can enforce those limits. In the absence of an MMU other means of enforcing the roles may be necessary (such as more detailed code reviews, for example).

7 VALIDATION APPROACH

The primary goal of the SCMS Manager validation of compliance with the requirements in the sections above is to provide assurance to all users of the V2X ecosystem that the many system components on which they rely are operating in a secure manner. The SCMS Manager is focused on validating that a minimum level of security is achieved by all system components. As a prerequisite to being provisioned certificates, devices will need to be able to communicate with an SCMS using TLS as defined in this specification. The correctness of the TLS implementation needs to be checked as part of the SCMS Manager validation.

SCMS Manager will specify the validation activities that are required. SCMS operators will be responsible for completing those validation activities with their certificate customers. SCMS operators will not be expected to perform validation activities themselves. Their responsibilities will be to receive and confirm documentation as specified by SCMS Manager. SCMS Manager will identify, and as necessary, establish agreements with third-party validation organizations. Member SCMS operators will use those services as required by SCMS Manager. Initially, recipients of IEEE 1609.2 certificates will provide self-declaration that the requirements of this document are met. SCMS Manager will provide a checklist for SCMS operators to use for this purpose.

As third-party testing facilities become available with appropriate means of validating these requirements, SCMS Manager will require validation by those third-party organizations. Over time, this will reduce the reliance on self-declaration. This is consistent with the approach being developed by the Connected Intersections Project at ITE.

SCMS operators shall retain and provide to the SCMS Manager, upon request, any documents related to the validation process for each certificate recipient.

7.1 VALIDATION REVIEW AND UPDATES

Devices validated under the procedures described above will be subject to re-validation based on requirements to be developed by SCMS Manager.

- Full or partial re-validation will be required when any changes are made to the product that impact the security of the device. Initially this determination will be made by self-declaration by the vendor. As will other aspects of validation, this may evolve over time to an external review.
- Re-validation may be done periodically to confirm that devices being delivered are the same as the device originally validated.
- Re-validation may be required when Misbehavior Detection or other analysis indicate a potential security problem with a validated device.

7.2 GRANDFATHERING / PHASE OUTS

SCMS Manager recognizes that some devices may be fielded before this validation process is fully implemented. SCMS Manager will establish guidelines for grandfathering and phasing out of devices as necessary.