



SURFACE VEHICLE RECOMMENDED PRACTICE

J3101™

FEB2020

Issued

2020-02

Hardware Protected Security for Ground Vehicles

RATIONALE

Automotive computer systems are required to establish trustworthiness through device identity, sealing, attestation, data integrity, and availability. These systems must be resilient to a wide range of attacks that cannot be thwarted through software-only security mechanisms. A hardware root of trust and the hardware-based security primitives are fundamentally necessary to satisfy demands of connected and highly or fully automated vehicles. This document provides a comprehensive view of security mechanisms supported in hardware for automotive use cases, along with best practices for using such mechanisms.

TABLE OF CONTENTS

1.	INTRODUCTION.....	4
1.1	Scope and Objective	4
1.2	Audience	4
2.	REFERENCES.....	5
2.1	Applicable Documents	5
2.1.1	SAE Publications.....	5
2.1.2	ISO Publications.....	5
2.1.3	NIST Publications.....	6
2.1.4	Other Publications.....	8
3.	DEFINITIONS	8
4.	HARDWARE PROTECTED SECURITY ENVIRONMENT	9
4.1	Defined	9
4.2	Design with a Hardware Protected Security Environment	10
4.3	Hardware Protected Security Abstraction Layers	10
4.4	Safety Determination of a System Secured by a Hardware Protected Security Environment	11
5.	LIFECYCLE OF HARDWARE PROTECTED SECURITY IN A VEHICLE	11
5.1	Engineering Development.....	11
5.2	Component Integration.....	11
5.3	Manufacture/Production	12
5.4	Distribution	12
5.5	Customer Use	12
5.6	Aftermarket Alteration	12
5.7	Resale/Reconditioning	12
5.8	Disposal/Retirement.....	12

SAE Technical Standards Board Rules provide that: "This report is published by SAE to advance the state of technical and engineering sciences. The use of this report is entirely voluntary, and its applicability and suitability for any particular use, including any patent infringement arising therefrom, is the sole responsibility of the user."

SAE reviews each technical report at least every five years at which time it may be revised, reaffirmed, stabilized, or cancelled. SAE invites your written comments and suggestions.

Copyright © 2020 SAE International

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of SAE.

TO PLACE A DOCUMENT ORDER: Tel: 877-606-7323 (inside USA and Canada)
Tel: +1 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
http://www.sae.org

SAE WEB ADDRESS:

Full document details: SAE J3101_202002

SAE values your input. To provide feedback on this Technical Report, please visit http://standards.sae.org/J3101_202002

Documents
Document Sharing Hub

6.	COMMON REQUIREMENTS	12
6.1	Mandatory versus Optional and Required versus Conditional Presentation of Requirements	13
6.2	Cryptographic Key Protection	13
6.2.1	Cryptographic Key Management Primer	13
6.2.2	Cryptographic Key Protection Overview	14
6.2.3	Requirements	15
6.2.4	Flow Examples (Informative)	26
6.3	Cryptographic Algorithms and Protocols	27
6.3.1	Cryptographic Algorithms Overview	27
6.3.2	Requirements	28
6.3.3	Implementation Considerations (Informative)	30
6.4	Random Number Generation	30
6.4.1	Random Number Generation Overview	30
6.4.2	Requirements	30
6.4.3	Implementation Considerations (Informative)	31
6.4.4	Understanding and Managing Entropy	32
6.5	Nonvolatile Critical Security Parameters	32
6.5.1	Nonvolatile Critical Security Parameters Overview	32
6.5.2	Requirements: Required	32
6.5.3	Implementation Considerations (Informative)	33
6.6	Cryptographic Algorithm Agility	33
6.6.1	Cryptographic Agility Overview	33
6.6.2	Requirements	33
6.7	Interface Control	34
6.7.1	Interface Control Overview	34
6.7.2	Requirements	34
6.8	Secure Execution Environment	36
6.8.1	Secure Execution Environment Basics: Required	36
6.8.2	Requirements	36
6.8.3	Implementation Considerations (Informative)	37
6.9	Self-Test	38
6.9.1	Operational States: Conditional	38
6.9.2	Self-Test Overview	39
6.9.3	Requirements	40
7.	PROFILES OF HARDWARE SECURITY COMMON REQUIREMENTS	41
7.1	Confidentiality Profile	42
7.2	Integrity Profile	43
7.3	Availability Profile	43
7.4	Access Control Profile	43
7.5	Non-Repudiation Profile	44
7.6	Limited Use	44
7.7	Assurance Level (Informative)	44
8.	VALIDATION AND VERIFICATION	45
9.	PRIMARY USE CASES	45
9.1	Authenticated Boot	45
9.1.1	Objectives	45
9.1.2	Scope and Definitions	45
9.1.3	Illustrative Process	46
9.1.4	Requirements	46
9.1.5	Implementation Considerations	47
9.1.6	Recommended Profiles	47
9.2	Authenticated Update	48
9.2.1	Objectives	48
9.2.2	Assumptions for Delivery of Firmware	48
9.2.3	Scope and Definitions	48
9.2.4	Illustrative Process	49
9.2.5	Requirements	50

9.2.6	Implementation Considerations	51
9.2.7	Recommended Profile(s)	52
9.3	Secure In-Vehicle Messaging	52
9.3.1	Objectives	52
9.3.2	Scope and Definitions	52
9.3.3	Requirements	52
9.3.4	Implementation Considerations	53
9.3.5	Recommended Profile(s)	54
9.4	Access Mechanisms	54
9.4.1	Objectives	54
9.4.2	Scope	54
9.4.3	Requirements	55
9.4.4	Illustrative Processes	56
9.4.5	Implementation Considerations	62
9.4.6	Recommended Profile(s)	65
9.5	Secure Storage	65
9.5.1	Objectives	65
9.5.2	Scope and Definitions	65
9.5.3	Requirements	65
9.5.4	Illustrative Process	66
9.5.5	Implementation Considerations	67
9.5.6	Recommended Profile(s)	68
10.	APPLICATION USE CASES	68
10.1	Intellectual Property Protection	68
10.1.1	Objectives	68
10.1.2	Scope and Definitions	68
10.1.3	Requirements	68
10.1.4	Implementation Considerations	69
10.1.5	Recommended Profile(s)	69
10.2	Secure Diagnosis at the ECU Level	69
10.2.1	Objectives	69
10.2.2	Scope and Definitions	69
10.2.3	Common Requirements	70
10.2.4	Illustrative Process	71
10.2.5	Recommended Profiles	73
10.3	Secure Logging	73
10.3.1	Objectives	73
10.3.2	Scope and Definitions	74
10.3.3	Common Requirements	74
10.3.4	Illustrative Process	75
10.3.5	Recommended Profiles	76
11.	NOTES	77
11.1	Revision Indicator	77
APPENDIX A	EXAMPLES OF ENTROPY AND DETERMINISTIC BIT GENERATORS	78
Figure 1	Hardware protected security environment abstraction layers	11
Figure 2	Key management and provisioning	14
Figure 3	Digital signature process	50
Figure 4	Example update flow by service technician	58
Figure 5	Access control - fully hardware protected security environment controlled	63
Figure 6	Access control - partial hardware protected security environment control	64
Figure 7	Authorization flow for secure diagnostics	71
Table 1	Common requirements of each profile	42

1. INTRODUCTION

Automotive computer systems are required to establish trustworthiness through device identity, sealing, attestation, data integrity, and availability. These systems must be resilient to a wide range of attacks that cannot be thwarted through software-only security mechanisms. A hardware root of trust and the hardware-based security primitives are fundamentally necessary to satisfy demands of connected and highly or fully automated vehicles. This document provides a comprehensive view of security mechanisms supported in hardware for automotive use cases, along with best practices for using such mechanisms. The goal of this document is to provide a common reference that facilitates communication among engineers across different parts of the automotive supply chain relevant to hardware-enabled security features. Silicon vendors will find this document useful in understanding the hardware security foundations and their corresponding use cases and applications that they should support to address vehicle security needs. This document should also bring more order into the diverse nature of hardware security features, so products are developed with the end use case in mind and with the right level of security. ECU suppliers and system integrators will benefit from the different security requirements and use cases outlined here as they assess the threats that affect their systems and the right hardware systems needed to address them.

This document represents a collection of characteristics of hardware mechanisms that are of use to, and address the needs of, the automotive industry in order to provide insight to the silicon industry and prevent fragmentation.

There exists a demand in the auto industry for a document to provide a baseline of due diligence in product development. This document aims to meet that need as a reference of industry best practices of minimum expectations of hardware protected cybersecurity.

1.1 Scope and Objective

This document presents a common set of requirements to be implemented in hardware-assisted functions to facilitate security-enhanced applications, to achieve an ideal system for hardware protection for ground vehicle applications.

This document will outline a common set of requirements to meet this goal and illustrate examples of the use of such requirements in various use cases which span the lifecycle of ground vehicle products, without explicitly detailing implementation requirements.

SAE J3101 has taken the approach of defining requirements through fundamental use cases. These requirements become building blocks for innovation, but are not development-process oriented. The presented building blocks are not attempting to encompass all future potential innovation; however, considerable future innovation should be possible through creative combinations of the presented requirements. It should be expected that some innovation may create future core requirements that themselves become new building blocks not captured within the scope of this revision of this document.

1.2 Audience

This standard is written from the point of view of automakers and suppliers to automakers addressed to embedded component suppliers such as microcontroller vendors. Although consumer automobile applications dominate the illustrated use cases the document is intended to apply to any ground vehicle application. Government applications suggested within this document are “non-tactical” characterized as otherwise civilian vehicles repurposed for government purposes. This document specifically does not include use cases of military applications

2. REFERENCES

2.1 Applicable Documents

The following publications form a part of this specification to the extent specified herein. Unless otherwise indicated, the latest issue of SAE publications shall apply.

2.1.1 SAE Publications

Available from SAE International, 400 Commonwealth Drive, Warrendale, PA 15096-0001, Tel: 877-606-7323 (inside USA and Canada) or +1 724-776-4970 (outside USA), www.sae.org.

SAE J3061 Cybersecurity Guidebook for Cyber-Physical Vehicle Systems

2.1.2 ISO Publications

Copies of these documents are available online at <http://webstore.ansi.org/>.

ISO/IEC 2382:2015	Information Technology - Vocabulary
ISO/IEC/IEEE DIS 8802-1	AE Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Part 1AE: Media Access Control (MAC) Security
ISO/IEC 9797-1:2011	Information Technology - Security Techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms Using A Block Cipher
ISO/IEC 9797-2:2011	Information Technology - Security Techniques - Message Authentication Codes (MACs) - Part 2: Mechanisms Using a Dedicated Hash-Function
ISO/IEC 10116:2017	Information Technology - Security Techniques - Modes of Operation for an N-Bit Block Cipheriso 15782-1:2009
ISO/IEC 17025:2017	General Requirements for the Competence of Testing and Calibration Laboratories
ISO 18031:2011	Information Technology - Security Techniques - Random Bit Generation
ISO/IEC 18033-3:2010	Information Technology - Security Techniques - Encryption Algorithms - Part 3: Block Ciphersiso 19772
ISO/IEC 19790:2012	Information Technology - Security Techniques - Security Requirements for Cryptographic Modulesiso/SAE DIS 21434
ISO 26262-1:2011	Road Vehicles - Functional Safety - Part 1: Vocabulary
ISO/IEC 27000:2016	Information Technology - Security Techniques - Information Security Management Systems - Overview and Vocabulary
ISO 29192-2:2012	Information Technology - Security Techniques - Lightweight Cryptography - Part 2: Block Ciphers

2.1.3 NIST Publications

Available from NIST, 100 Bureau Drive, Stop 1070, Gaithersburg, MD 20899-1070, Tel: 301-975-6478, www.nist.gov.

NIST SP 800-38A	Recommendation for Block Cipher Modes of Operation: Methods and Techniques (December 2001)
NIST SP 800-38B	Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication (May 2005)
NIST SP 800-38C	Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality (July 2007)
NIST SP 800-38D	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC (November 2007)
NIST SP 800-38E	Recommendation for Block Cipher Modes of Operation: the XTS-AES Mode for Confidentiality on Storage Devices (January 2010)
NIST SP 800-38F	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping (December 2012)
NIST SP 800-47	Security Guide for Interconnecting Information Technology Systems (August 2002)
NIST 800-53A	Assessing Security and Privacy Controls in Federal Information Systems and Organizations: Building Effective Assessment Plans (December 2014)
NIST SP 800-57	Part 1, Rev. 4: Recommendation for Key Management, Part 1: General (January 2016)
SP 800-90A	Rev. 1: Recommendation for Random Number Generation Using Deterministic Random Bit Generators (June 2015)
SP 800-90B	Recommendation for the Entropy Sources Used for Random Bit Generation (January 2018)
SP 800-90C	[draft] Recommendation for Random Bit Generator (RBG) Constructions (April 2016)
NIST SP 800-131A	Rev. 2: Transitioning the Use of Cryptographic Algorithms and Key Lengths (March 2019)
NIST IR 7316	Assessment of Access Control Systems (September 2006)
FIPS PUB 140-2	Security Requirements for Cryptographic Modules (May 2001)
FIPS PUB 140-3	Security Requirements for Cryptographic Modules (March 2019)
FIPS PUB 186-4	Digital Signature Standard (DSS) (July 2013)
FIPS PUB 180-4	Secure Hash Standard (SHS) (August 2015)
FIPS PUB 197	Advanced Encryption Standard (AES) (November 2001)
FIPS PUB 198-1	The Keyed-Hash Message Authentication Code (HMAC) (July 2008)
FIPS PUB 199	Standards for Security Categorization of Federal Information and Information Systems (February 2004)
FIPS PUB 202	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions (August 2015)

Cryptographic Algorithm Validation Program (CAVP)

<https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program>

Cryptographic Module Validation Program (CMVP)

<https://csrc.nist.gov/projects/cryptographic-module-validation-program>

Interoperable Randomness Beacons

<https://csrc.nist.gov/projects/interoperable-randomness-beacons>

Entropy as a Service

<https://csrc.nist.gov/projects/entropy-as-a-service>

NISTCAVP - Cryptographic Algorithm Validation Program

<https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/validation>

NISTDRBG - Cryptographic Algorithm Validation Program (DRBG)

<https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/validation/validation-list/drbg>

NISTRNG - Cryptographic Algorithm Validation Program (RNG)

<https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/validation/validation-list/rng>

2.1.3.1 IETF Publications

Copies of these documents are available online at <http://tools.ietf.org/>

RFC 7696	Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms
RFC 4949	Internet Security Glossary, Version 2 (August 2007)
RFC 6979	Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA) (August 2013)
RFC 8391	XMSS: eXtended Merkle Signature Scheme (May 2018)
RFC 7905	ChaCha20-Poly1305 Cipher Suites for Transport Layer Security (TLS) (June 2016)
RFC 8439	ChaCha20 and Poly1305 for IETF Protocols (June 2018)
RFC 7748	Elliptic Curves for Security (January 2016)
RFC 8032	Edwards-Curve Digital Signature Algorithm (EdDSA) (January 2017)
RFC 8554	Leighton-Micali Hash-Based Signatures (April 2019)
RFC 8125	Requirements for PAKE Schemes (April 2017)

2.1.3.2 IEEE Publications

Available from IEEE Operations Center, 445 and 501 Hoes Lane, Piscataway, NJ 08854-4141, Tel: 732-981-0060, www.ieee.org.

IEEE 1609.2-2016	Wireless Access in Vehicular Environments-Security Services for Applications and Management Messages
IEEE 1619-2018	Cryptographic Protection of Data on Block-Oriented Storage Devices

2.1.4 Other Publications

ANSI X9.62, "Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)," 2005.

IEEE-ISTO 6100.1.0.0, "Uptane Standard for Design and Implementation," 2019.

ITU-T X.509, "Information Technology - Open Systems Interconnection – The Directory: Public-Key and Attribute Certificate Frameworks," International Telecommunication Union, October 2016, <https://www.itu.int/rec/T-REC-X.509>.

BSI AIS 20/31, "Evaluation of Random Number Generators," Bundesamt für Sicherheit in der Informationstechnik, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_20_AIS_31_Evaluation_of_random_number_generators_e.pdf.

Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A., "Algorithm 9.43," Handbook of Applied Cryptography, August 2001, CRC Press, ISBN 0-8493-8523-7, <http://cacr.uwaterloo.ca/hac/>.

44 U.S.C., Sec. 3542: Title 44 United States Code Section 3542 - "Definitions."

Blake-Wilson, S., Menezes, A., "Unknown Key-Share Attacks on the Station-to-Station (STS) Protocol," Lecture Notes in Computer Science, 1560, Springer, 154-170, 1999.

Dan Kaminsky "Black Ops" Blog, <https://dankaminsky.com/2012/08/06/bo2012/> - Black Hat 2012.

[Shor's algorithm] Peter W. Shor. 1997. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," SIAM J. Comput. 26, 5, 1484-1509, 1997, doi:<http://dx.doi.org/10.1137/S0097539795293172>.

[PQC_NIST] Dustin Moody, The ship has sailed: The NIST Post-Quantum Crypto "Competition," NIST.

TCG TPM 2.0 Library r1.38, (February, 2017), <https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-1-Architecture-01.38.pdf>.

Weimerskirch and Schramm, "Using software flashing to secure embedded device updates," <http://www.embedded.com/design/configurable-systems/4008264/3/Using-software-flashing-to-secure-embedded-device-updates>.

3. DEFINITIONS

The following provides further refinement of definitions of terms for the purpose of SAE J3101.

ACCESS CONTROL: Obtaining the use of a resource (ISO/IEC 2382:2015). Access control is concerned with determining the allowed activities of legitimate users, mediating every attempt by a user to access a resource in the system, thus protecting system resources against inappropriate or undesired user access (NIST: Assessment of Access Control Systems).

ASSET: Anything that has value to the product's stakeholders. Assets can be tangible or intangible. Examples of tangible assets are ECUs, sensors, or actuators. Examples of intangible assets are reputation, or intellectual property (ISO 21434 (draft)).

AUTHENTICATION: A provision of assurance that a claimed characteristic of an entity is correct (ISO 27000:2016).

AUTHORIZATION: Granting of rights (ISO 15782-1:2009).

CRITICAL SECURITY PARAMETERS (CSP): Contain security-related information (e.g., counters, authentication data such as passwords and personal identification numbers (PINs)) whose disclosure or modification can compromise the security of a cryptographic operation (FIPS 140-2).

CRYPTO AGILITY: The ability to migrate from one implemented algorithm suite to another over time (BSI 7009).

CYBERSECURITY PROPERTIES:

- a. **AVAILABILITY:** Ensuring timely and reliable access to and use of information (data) (44 U.S.C., Sec. 3542). A property of being accessible and usable upon demand by an authorized entity (ISO/IEC 27000:2016). A loss of availability is the disruption of access to or use of information or an information system (FIPS 199).
- b. **CONFIDENTIALITY:** Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information (44 U.S.C., Sec. 3542). A property of information that is not made available or disclosed to unauthorized individuals, entities, or processes (ISO/IEC 27000:2016).
- c. **INTEGRITY:** Guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity (44 U.S.C., Sec. 3542). A property of accuracy and completeness (ISO/IEC 27000:2016).

FIRMWARE: Computer programs and data stored in hardware; for example, in programmable read-only memory such that typically the programs and data cannot be dynamically written or modified during execution of the firmware (RFC 4949). Firmware may be updated by an authorized controlled mechanism.

FRESHNESS: Data is recent (current), and ensures not to have been substituted by old (previously consumed) data.

IDENTIFICATION: The process of verifying the identity of a user, process, or device, usually as a prerequisite for granting access to resources in an information technology system (NIST SP 800-47).

KEYSTORE: A keystore is a facility for storing the keys of a cryptographic application which provides a security service (see 6.2).

RESOURCE/OBJECT: An entity in the system that either contains or receives information or can perform an action or a function. Access to a resource implies access to the information and/or functionality of that resource.

SECURITY MECHANISM: An element of a system that includes specific protection related functionality which provides cybersecurity to a system (NIST 800-53A).

SELF-TEST: Test procedures executed by the hardware protected security environment automatically and/or on-demand to ensure the module is functioning properly (FIPS 140-2).

ZEROIZATION: A method of erasing electronically stored data, cryptographic keys, and critical security parameters by altering or deleting the contents of the data storage to prevent recovery of the data (FIPS 140-2).

4. HARDWARE PROTECTED SECURITY ENVIRONMENT

4.1 Defined

The body of requirements outlined within SAE J3101 shall define an environment of security referred to as the hardware protected security environment.

Everything else outside the scope of requirements outlined within hardware protected security for ground vehicles would represent the “normal environment” (e.g., where applications execute), which represents a larger attack surface and is a less trusted domain compared to the hardware protected security environment.

A hardware protected security environment offers at least one mechanism of security implemented by hardware support, which may be leveraged or extended further by firmware.

Hardware protected security offers hardware mechanisms which may require physical presence to perform specific security functions beyond the reach of the normal environment.

Firmware written to extend the hardware protected security environment, potentially leveraging shared hardware resources, must operate in a way to extend the security of the available hardware elements in a chain of trust offering services of the hardware protected security environment in a secure manner rooted to the hardware security mechanisms of the microcontroller.

One of the primary purposes of the HW hardware protected security environment is to reduce the size, complexity, and attack surface of the trust base of the system as a whole, forming a minimal trust anchor which is implicitly trusted. The trust anchor then forms the basis by which all other functionality may be verified. Additionally, the hardware protected security environment includes a means to shield secrets from the normal environment and resist exposure of secrets.

Firmware within the hardware protected security environment can make use of the hardware to protect data at rest and at use.

Normal environment applications can make use of the security environment to secure data on behalf of such application; additionally, normal environment applications may make use of the hardware of a hardware protected security environment directly, however, only using it in a non-trusted manner.

4.2 Design with a Hardware Protected Security Environment

The hardware protected security environment is potentially a closed system where inputs are controlled, and requirements can be implemented with respect to RAM, key storage, etc. With this closed system, a general building block may be achieved. Those blocks could be used at a system level if appropriate and applicable. Leveraging this building block model, certain hardware protected security environment requirements may apply to any ECU within vehicle networks, while others may only apply to specialized ECUs such as gateways and outside vehicle communications ECUs.

For example, under the use case of a smart sensor where only a very small microcontroller is used, proposals are pushing security into the smart sensors or also relying on security of the receive side of the signal (upstream to the ECU). Best practice would be to put a hardware protected security environment even in constrained sensors that are downstream and not internal to another ECU (with a hardware protected security environment). Such would provide defenses to avoid internally originated attacks on an in-vehicle network.

If an ECU is designed with multiple chips within it, then it can be feasible to design the interaction of the chips to rely on only a single hardware protected security environment within that ECU.

One objective of the hardware protected security environment is to avoid the need for all vendors to be cryptography experts, and concentrate the implementation of cryptography within an isolated sub-system.

Design for security within normal environment applications is beyond the scope of SAE J3101.

4.3 Hardware Protected Security Abstraction Layers

SAE J3101 covers the physical layer and the first hardware abstraction layers and security mechanisms implemented within them. Normal environment middleware and applications are beyond the scope of SAE J3101 and shall not be leveraged to achieve any requirements outlined in SAE J3101.

Note that some other physical interfaces and hardware abstraction layer components may be shared with the normal environment. Any hardware component shared between the normal and protected environments shall not have any memory or hysteresis of any secret material, so that the normal application which uses a shared hardware component cannot determine any characteristics of the security mechanism through that HW component; e.g., hardware memory protection unit, timers, counters, peripheral interfaces.

Note some hardware components cannot be shared outside the security environment and maintain security, such components must be dedicated to the security environment to avoid providing elevated privilege to normal applications.

Hardware Protected Security Abstraction Layers

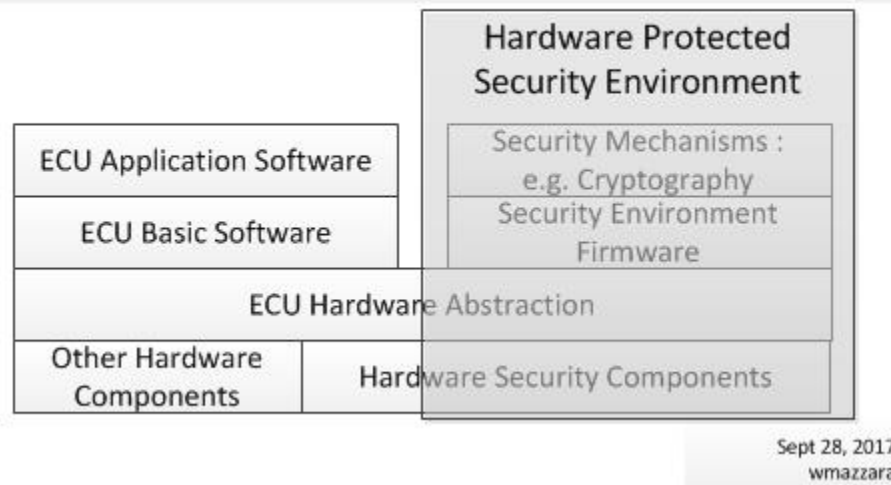


Figure 1 - Hardware protected security environment abstraction layers

4.4 Safety Determination of a System Secured by a Hardware Protected Security Environment

The use case of an application which utilizes an interface to the hardware protected security environment may be considered safety relevant depending on the target application. The system designer should refer to ISO 26262 for guidance on functional safety requirements. A hardware protected security environment may be used to comply with a safety goal or may be a potential cause of violation of a safety goal. In such cases, the hardware protected security environment is to be treated like any other hardware component that has safety-related requirements. Safety mechanisms designed to protect the system against failures should not violate the hardware protected security environment security requirements.

EXAMPLE: A common example where the hardware protected security environment may impact a safety goal is the usage of a cipher-based message authentication code (CMAC) as a safety mechanism for end-to-end protection of network communication. The safety of the system may be impacted if a false positive is provided by the hardware protected security environment as a result of a systematic or random failure. It is important to note that different system architectures may allow the safety goal to be met through system-level redundant safety mechanisms which would relieve the hardware protected security environment from having to meet the stringent safety requirements.

5. LIFECYCLE OF HARDWARE PROTECTED SECURITY IN A VEHICLE

Phases of lifecycle are to be used to distinguish hardware requirements based on individual or subsets of expected device lifecycles. A priority must be placed on ensuring that accommodation in hardware to meet usability in one phase of lifecycle does not compromise security, beyond tolerance, in another phase. SAE J3101 delineates the lifecycle of a vehicle into the following phases. Additional detail on the relevance of product lifecycle to cybersecurity of vehicle systems can be found in the SAE J3061 standard.

5.1 Engineering Development

Component development work done by the engineering team responsible for the component itself.

5.2 Component Integration

Sub-system integration and development work done up the chain of the vendor customer relationships to achieve the end customer product, i.e., vehicle.

5.3 Manufacture/Production

Series of assembly work done through a chain of vendor customer relationships to build the end customer product, i.e., vehicle.

Sub-phases to manufacture/production exists through the tiers of a supply chain from the silicon vendor through various tiers of suppliers to ultimately the end product manufacturer (OEM). Each sub-phase may have unique requirements of key management and authorization to certain privileges.

5.4 Distribution

Mode of operation required to ship and/or sell the end customer product, i.e., vehicle to the end customer.

5.5 Customer Use

Primary design intent mode of operation of the end customer product, i.e., vehicle. Must consider necessary service and anticipated updates of product as used by the actual customer, i.e., the whole intact vehicles.

5.6 Aftermarket Alteration

Product lifecycle phase which may or may not occur to a product under which the OEM and/or other third parties may revise the function or configuration of the product.

5.7 Resale/Reconditioning

Mode of operation and/or set of functionalities required to recondition a part of or the complete end customer product, i.e., vehicle for resale to another customer, or for transition of control in the case of shared assets, e.g., fleets.

5.8 Disposal/Retirement

Mode of operation and/or set of functionalities required to properly decommission a part of or the complete end customer product, i.e., vehicle. Lifecycle of hardware protected security in a vehicle.

6. COMMON REQUIREMENTS

The following common requirements have been derived from the use cases detailed later in the document, and represent categories of hardware protected security environment functionality.

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

6.1 Mandatory versus Optional and Required versus Conditional Presentation of Requirements

Section 7 will organize the common requirements into meaningful sets. The subsections presented as “required” are necessary to appropriately characterize the common requirement addressed by the subsection if it is necessary to achieve the profile. Within each common requirement, “conditional” subsections may be implemented if appropriate for the use case, but may still be omitted if the profile is applied to a different use case. Statements within each section are marked “mandatory” or “optional.” If the implementation claims to meet the characteristics of the common requirement, the “mandatory” statements must be supported. So subsections of common requirements marked as “required” may be omitted if the profile does not include that common requirement, and “mandatory” statements within “conditional” subsections of a common requirement are not necessary under appropriate use cases.

For example, the common requirements are presented using the following template headings:

X. Common Requirements

X.Y Heading

X.Y.Z Sub-Heading

X.Y.Z.1 Example Heading: Required

[MANDATORY] REQ_X.Y.Z.1_10: Requirement Text

[OPTIONAL] REQ_X.Y.Z.1_20: Requirement Text

X.Y.Z.2 Example Heading: Conditional

[MANDATORY] REQ_X.Y.Z.2_10: Requirement Text

[OPTIONAL] REQ_X.Y.Z.2_20: Requirement Text

NOTE: Some requirement numbers were associated with text which was deleted during development. The original numbers of requirements have been preserved for continuity and are not necessarily sequential.

6.2 Cryptographic Key Protection

6.2.1 Cryptographic Key Management Primer

A principal requirement of any hardware protected security environment is to provide a protected storage, management, and usage capability for cryptographic keys. All mechanisms and functionality associated with managing crypto keys are elements of secure cryptographic key management. Broadly, key management functionality encompasses the full lifecycle of cryptographic keys, from genesis (i.e., generation, establishment, derivation, or input), storage, and management, to destruction (zeroization). These key management facets may be described in terms of the hierarchy depicted in the following figure.

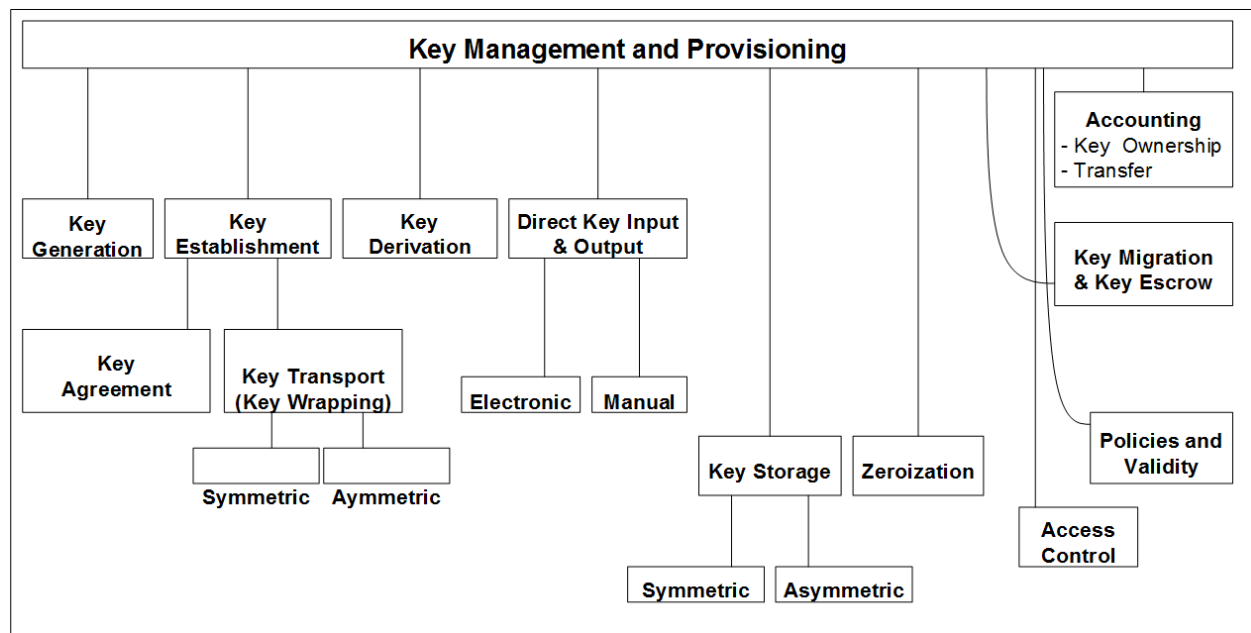


Figure 2 - Key management and provisioning

Hardware protected security environment key management functionality will encompass some or all of the key management facets, depending on supported key types, cryptographic algorithms and protocols that are employed, and client functionality demanded by larger system-level security services outside of the hardware protected security environment.

6.2.2 Cryptographic Key Protection Overview

A common requirement of any hardware protected security environment is a protected storage and usage environment for the purpose of storing cryptographic keys which is to be defined as a “keystore.”

Keystores: The component of the hardware protected security environment that directly manages the cryptographic keys within a larger key management context is defined as a “keystore.” The keystore is responsible for securely storing the keys to prevent leakage or any unauthorized disclosure of keys outside of the hardware protected security environment. Management and utilization of a keystore is a headline use case of the hardware protected security environment and is not optional. While general purpose and embedded operating systems typically include APIs to manage a “keystore,” API specifics are beyond the scope of SAE J3101.

Hardware protected keystores are necessary to support persistent storage of all keys symmetric (secret) and asymmetric (public and private key pairs).

Note that there is an overlap in function of the keystore with the core requirement of 6.5, and use cases of 9.5, and 10.1. The keystore is fundamentally tied to the keys which enable cryptographic functions of the hardware protected security environment. The keystore shall be specifically defined as distinct from more general nonvolatile data protection (within the hardware protected security environment).

Secure key usage is another distinct aspect of the common requirements of the hardware protected security environment. An API is assumed to be required through which the normal environment and the secure environment can distinctly manage and use the keystore without unauthorized disclosure of the keys in the keystore. In cryptography, secret and private keys (symmetric and asymmetric) must be protected, but chosen algorithms are known.

Algorithms which use the key must maintain the security of the key during its use. Secure key storage and secure key usage should be coupled. The exposure of the key can lead to the complete unravelling of the whole crypto scheme.

See 6.8.

6.2.3 Requirements

6.2.3.1 Types of Keys

[MANDATORY] REQ_6.2.3.1_10: The hardware protected security environment shall support digital certificates if public keys (asymmetric cryptography) are employed. The digital certificates should be X.509 or IEEE 1609.2 compatible formats.

[OPTIONAL] REQ_6.2.3.1_20: The hardware protected security environment shall support either ephemeral or long-term symmetric keys, or both.

6.2.3.2 Key Storage: Required

This aspect of key management defines the rules and processes associated with secure persistent and/or ephemeral storage of cryptographic keys. Proper key storage ensures protection of key material at rest.

[MANDATORY] REQ_6.2.3.2_10: A hardware protected security environment must securely store all cryptographic keys and explicitly control access to each.

[MANDATORY] REQ_6.2.3.2_20: A keystore may be direct storage of the keys within the hardware protected security environment, or use of external storage external to the hardware protected security environment that is protected by encryption and integrity mechanisms implemented within the hardware protected security environment.

[OPTIONAL] REQ_6.2.3.2_30: Key storage capacities should only be constrained by the physical limits of the underlying hardware. Allocation of storage between differing uses should be defined under each application specified for the hardware protected security environment, both in maximums and minimums. Denial of service due to exhaustion of available resource should be mitigated by a resource manager implemented in either hardware or firmware as a part of the hardware protected security environment.

NOTE: The hardware protected security environment is responsible for protecting both long-term and short-term asymmetric and symmetric keys. Short term keys are sometimes referred to as ephemeral keys.

[MANDATORY] REQ_6.2.3.2_40: The hardware protected security environment keystore and its cryptographic key contents shall be separately managed from any other nonvolatile data protection. This includes other non-volatile data both within the hardware protected security environment and in the normal environment.

[MANDATORY] REQ_6.2.3.2_50: The hardware protected security environment shall distinctly manage the cryptographic integrity of the keystore from other components of the hardware protected security environment.

[MANDATORY] REQ_6.2.3.2_60: The hardware protected security environment shall manage and maintain the integrity of the entire keystore during any updates to the keystore (addition, modification, or deletion of a key).

[MANDATORY] REQ_6.2.3.2_70: The hardware protected security environment shall check the integrity of the keystore before any transaction which makes use of a stored key, as well as in response to any conditions that warrant a check of the integrity of the keystore specified by the system design (e.g., normal or abnormal changes to power mode).

[MANDATORY] REQ_6.2.3.2_100: The hardware protected security environment keystore shall maintain the identification and association of each cryptographic key to its authorized calling application (called key-entity association), whether the application is located internal or external to the hardware protected security environment, and whether the key is stored within or outside the protected hardware protected security environment boundary.

- [MANDATORY] REQ_6.2.3.2_110: The hardware protected security environment shall maintain the hierarchy and/or association of all cryptographic keys within the keystore that are protected by other stored cryptographic keys.
- [MANDATORY] REQ_6.2.3.2_130: All cryptographic keys stored encrypted within the keystore (internal or external to the hardware protected security environment secure boundary) shall be encrypted by a cryptographic algorithm and key length combination that results in a computational resistance to attack equal to or greater than the stored key.
- [OPTIONAL] REQ_6.2.3.2_140: If the encryption mode used to encrypt any internally or externally stored key requires an initialization vector, such shall be distinct in each key encryption operation.
- [MANDATORY] REQ_6.2.3.2_150: The hardware protected security environment shall support binding of keys that are stored encrypted within the keystore to a specific device.

NOTE: This prevents transferring encrypted keys stored in external non-volatile memories from a given device to prevent copying the stored keys to multiple other devices.

6.2.3.2.1 Long-Term Keying Material Protection: Required

- [MANDATORY] REQ_6.2.3.2.1_10: Long-term keys must be protected while at rest for confidentiality, integrity, and against replay attacks. These attacks could be mounted on the nonvolatile storage device containing these keys or in transit from the nonvolatile storage device to the security module if the security module does not contain on-board nonvolatile storage.

6.2.3.2.2 Short-Term Keying Material Protection: Required

- [MANDATORY] REQ_6.2.3.2.2_20: The hardware protected security environment shall protect the confidentiality and integrity of short-term keys used on a session basis while in transit (during key exchange operations) and against replay attacks.
- [MANDATORY] REQ_6.2.3.2.2_30: The hardware protected security environment shall ensure that the exposure of any short-term keys used on a session basis do not compromise the integrity and confidentiality of any other short-term keys used in past or future sessions.
- [OPTIONAL] REQ_6.2.3.2.2_10: The hardware protected security environment shall protect short-term keys used on a session basis such that at the termination of that session their use is also terminated.

6.2.3.2.3 Symmetric Key Protected Storage: Required

- [MANDATORY] REQ_6.2.3.2.3_10: Symmetric keys (also referred to as shared secret keys) shall be preserved for integrity, confidentiality, and freshness.
- [OPTIONAL] REQ_6.2.3.2.3_20: The hardware protected security environment shall encrypt internally stored symmetric keys using an internally stored key encryption key.
- [MANDATORY] REQ_6.2.3.2.3_30: The hardware protected security environment shall output externally stored symmetric keys by a method that meets the key output requirements of this document.
- [OPTIONAL] REQ_6.2.3.2.3_50: The hardware protected security environment shall support a defined lifetime and a means of rekeying for symmetric keys.

NOTE: There are multiple ways to ensure defined symmetric key lifetimes which are not necessarily date derived, such as usage and/or boot cycle and/or ignition cycle counters. Hardware protected security environment enforcement of key lifetime is essential. Recommendations from various standards bodies on key lifetimes can be found at <http://www.keylength.com/en/4/>.

6.2.3.2.4 Asymmetric Key Protected Storage: Required

Asymmetric keys are comprised of public and private keys.

[MANDATORY] REQ_6.2.3.2.4_10: The hardware protected security environment shall protect public keys by using the hardware protected security environment's facilities for integrity to prevent unauthorized modification or substitution or rollback.

[MANDATORY] REQ_6.2.3.2.4_15: For confidentiality, the hardware protected security environment shall store and use private keys strictly inside the hardware protected security environment for confidentiality, i.e., to prevent any unauthorized disclosure of the private key outside the environment.

[MANDATORY] REQ_6.2.3.2.4_20: The hardware protected security environment shall maintain the associations of all private keys with their respective public keys.

6.2.3.3 Protection at Use: Required

Secure Key Usage: Secure key usage pertains to a hardware protected security environment controlling the use (execute privileges) of a cryptographic key by a specific entity (application) for a specific purpose. An API is assumed to be required through which applications outside of the hardware protected security environment can distinctly manage and use the keystore without unauthorized disclosure of its keys. Secure key usage pertains to the key store APIs and the authorization to access the keys and control their usage.

[MANDATORY] REQ_6.2.3.3_10: If all verifications/authorizations passed successfully, the hardware protected security environment shall perform all cryptographic operations within the hardware protected security environment and return only the resulting data to the normal environment.

[MANDATORY] REQ_6.2.3.3_20: The hardware protected security environment shall maintain the security of all cryptographic algorithm keys during their use.

6.2.3.4 Key Ownership and Accessibility: Required

[MANDATORY] REQ_6.2.3.4_10: The hardware protected security environment shall not allow access to any keys stored internally outside of the hardware protected security environment, except in an encrypted form, with explicit authorization from the hardware protected security environment.

NOTE: Examples are:

- Using JTAG, debugging, and viewing logs does not reveal keys in the clear.
- If the product specification requires, key generation and migration of keys to an entity external to the system is accomplished without knowledge of the key being disclosed to any additional entities.

6.2.3.4.1 Key Ownership: Required

Key ownership is a type of key-entity association (applications or entities to which a key is associated) that addresses the organization(s) or user(s) that own or control the key and its use within the hardware protected security environment.

[OPTIONAL] REQ_6.2.3.4.1_10: The hardware protected security environment shall manage key ownership by independently managing key usage authorization and key change authorization.

[MANDATORY] REQ_6.2.3.4.1_20: The hardware protected security environment, if a keystore is specified to manage keys from multiple owners (either end product owners or supply chain entities), shall check the authorization of any entity which requests to install or invalidate a key within the keystore as a part of the requesting transaction.

NOTE: Requirements to define ownership of keys are beyond the scope of this document. Ownership transition between OEM, component manufacturer, and/or customer upon sale of a vehicle/component from one party to another may necessitate a system procedure to zeroize all keys of the prior owner and provision keys of the new owner. This could also be a part of the end of life of a vehicle. However, such system procedures must make use of the hardware protected security environment functions presented herein.

6.2.3.4.2 Key Access Control: Required

- [MANDATORY] REQ_6.2.3.4.2_10: The hardware protected security environment shall be maintain and perform all keystore management and access logic within the boundary of a hardware protected security environment.
- [MANDATORY] REQ_6.2.3.4.2_20: The hardware protected security environment shall implement access control for each key stored within its hardware-secured boundary.
- [MANDATORY] REQ_6.2.3.4.2_30: The hardware protected security environment shall authenticate and authorize all changes to access policies as a part of the transaction which requests the changes.
- [MANDATORY] REQ_6.2.3.4.2_40: If the hardware protected security environment host device requires any key management function to be performed by the hardware protected security environment, then the hardware protected security environment shall authenticate the authorization of the requesting entity as a pre-condition to performing the action.
- [MANDATORY] REQ_6.2.3.4.2_50: The hardware protected security environment shall prohibit all cryptographic key access and invocation of cryptography until it has completed its own power-on integrity test (refer to NIST FIPS 140-2).
- [MANDATORY] REQ_6.2.3.4.2_60: The hardware protected security environment shall protect plain text secret and private keys from unauthorized read, write, and execute access via all physical and logical interfaces to the hardware protected security environment and its keystore.

NOTE: Examples include: JTAG, UART, debugging interfaces, log access interfaces, APIs.

- [OPTIONAL] REQ_6.2.3.4.2_70: The hardware protected security environment shall require explicit authorization to export any keys to an authenticated and authorized user.
- [MANDATORY] REQ_6.2.3.4.2_80: The hardware protected security environment, if product specifications require export of keys to an entity external to the hardware protected security environment, shall perform that action entirely within the hardware protected security environment without disclosing any cryptographic key attributes to any entity other than the intended recipient.
- [MANDATORY] REQ_6.2.3.4.2_110: The hardware protected security environment shall manage the specified usage policies for each key it holds (example usage policies may be based on purpose, application, time, or context).

6.2.3.4.3 Key Authorized Use: Required

- [MANDATORY] REQ_6.2.3.4.3_10: The hardware protected security environment shall verify usage or validity rules (e.g., validity periods, geo-fence constraints, frequency of use, etc.) as required by internal or external applications according to the key management plan.
- [MANDATORY] REQ_6.2.3.4.3_20: The hardware protected security environment shall ensure the integrity and freshness and association of cryptographic key usage rules/policies to their associated keys

[MANDATORY] REQ_6.2.3.4.3_30: If all verifications/authorizations for a requested key use have been passed, then the hardware protected security environment shall perform the requested cryptographic operation within or protected by the hardware protected security environment. [MANDATORY] REQ_6.2.3.4.3_40: The hardware protected security environment shall manage key attributes to protect a key from use by an inappropriate function or algorithm. (e.g., signature key use in an encryption algorithm).

6.2.3.4.4 Key Access Policies: Required

[MANDATORY] REQ_6.2.3.4.4_10: During use, the keystore shall keep private asymmetric keys and shared secret (symmetric) keys confidential through use of either encryption or access control.

[MANDATORY] REQ_6.2.3.4.4_20: During use, the keystore shall maintain the integrity of public asymmetric keys through Authentication, encryption or access control.

[MANDATORY] REQ_6.2.3.4.4_30: The hardware protected security environment shall implement access control for each key stored within its hardware protected security environment.

[MANDATORY] REQ_6.2.3.4.4_40: Changes to access policies shall be authenticated within the hardware protected security environment as a part of the transaction which requests the changes.

[MANDATORY] REQ_6.2.3.4.4_50: The hardware protected security environment shall authenticate the authority of the requesting entity before performing a key access as a single transaction with any cryptographic action using that key.

6.2.3.5 Key Validity: Required

[MANDATORY] REQ_6.2.3.5_10: The hardware protected security environment shall manage the validity of all keys according to policies established in the hardware protected security environment's key management plan.

NOTE: For example, validity may be determined by time, product lifecycle, market region, etc.

[MANDATORY] REQ_6.2.3.5_20: The hardware protected security environment shall not reuse any key revoked or otherwise marked invalid thereafter (however, re-provisioning of a new key is an acceptable recovery strategy).

NOTE: The concept may exist of invalid keys within a key space (e.g., all ones). Hardware protected security environment enforcement of specified validity rules may be required.

NOTE: The hardware protected security environment ensures that the cryptographic key validity lifetime can be dictated by date, or time, attributes, or other attributes (such as frequency of usage, boot cycle counts, ignition cycle counters, and, in some cases, even the estimated longevity of the cryptographic algorithm and the key's length).

NOTE: The hardware protected security environment can also tie key validity to a successful authentication of the normal environment firmware/software. In this case, the keys are not used for any operation until the authentication is completed

NOTE: The use of one time programmable (OTP) memory devices to hold keys can present a challenge to managing the validity of these keys. Considerations in design should be implemented to allow appropriate management of the validity of all keys.

6.2.3.6 Time-Bound Keys and Algorithms

[NIST SP 800-57 Part 1, Rev. 4]

A cryptographic key period is the time span during which a specific key is authorized for use by legitimate entities on the keys for a given system will remain in effect.

NOTE: A suitable cryptographic key period:

1. Limits the amount of information protected by a given key that is available for cryptanalysis.
2. Limits the amount of exposure if a single key is compromised.
3. Limits the use of a particular algorithm (e.g., to its estimated effective lifetime).
4. Limits the time available for attempts to penetrate physical, procedural, and logical access mechanisms that protect a key from unauthorized disclosure.
5. Limits the period within which information may be compromised by inadvertent disclosure of keying material to unauthorized entities.
6. Limits the time available for computationally intensive cryptanalytic attacks (in applications where long-term key protection is not required).

NOTE: Risk factors affecting the length of a crypto period include, but are not limited to:

1. The strength of the cryptographic mechanisms (e.g., the algorithm, key length, block size, and mode of operation).
2. The embodiment of the mechanisms (implementation strength).
3. The operating environment.
4. The volume of information flow or the number of transactions.
5. The security life of the data.
6. The security function (e.g., data encryption, digital signature, key derivation, or key protection).
7. The re-keying method (e.g., keyboard entry, re-keying using a key loading device where humans have no direct access to key information, or remote re-keying within a PKI).
8. The key update or key-derivation process.
9. The number of nodes in a network that share a common key.
10. The number of copies of a key and the distribution of those copies.
11. Personnel turnover (e.g., root certificate authority personnel).
12. The threat to the information from adversaries (e.g., whom the information is protected from, and what are their perceived technical capabilities and financial resources to mount an attack).
13. The threat to the information from new and disruptive technologies (e.g., quantum computers).

NOTE: Algorithms required or forbidden by specific nation's regulations are not referenced in this document; however, such regulations demonstrate another need for crypto agility for the purpose of producing a product sellable on a global basis in accordance with trade laws.

NOTE: Proprietary or customized cryptographic algorithms always increase the risk of unintended security vulnerabilities because they do not benefit from formal analysis and testing in the professional cryptography and security communities (as the design documents and source code are typically not publicly available).

NOTE: Guidance on deprecated or obsolete cryptographic algorithms can be found in NIST SP 800-131A.

6.2.3.7 Key Management System: Required

Key Accounting: Key accounting refers to the recording, audit, and reporting of any data elements, events, or other security-relevant attributes associated with a key during its entire lifecycle. Only hardware protected security environment-specific key accounting (i.e., logging-related) requirements are specified by this document. System-specific accounting and reporting requirements are not addressed within this specification.

Key Escrow: Key escrow refers to the process of exporting a cryptographic key, typically for backup and recovery purposes needed to meet system availability requirements. System-level key escrow requirements are not addressed by this document, only the requirements pertaining to secure key transport (export) from the hardware protected security environment.

Key Migration: Key migration is the export, typically of all operational keys, from one device to another device; the operation may include elements of key escrow if the migration is being performed to or from an escrowed collection of keys. This type of activity may be necessary in automotive ECUs; for example, if an ECU hardware needs to be replaced and other vehicular components are configured only to communicate with keys installed on the outgoing device. In this case, migrated keys would require re-association to the new ECU's root of trust. Key migration is a very sensitive operation and may not be a permitted operation in an operational customer use lifecycle state. Key migration should require the strictest rule/policy.

[MANDATORY] REQ_6.2.3.7_10: The hardware protected security environment shall satisfy the functional requirements of whatever key provisioning scheme is specified for the product and must execute each such function in a seamless single API call to the hardware security environment, so that the validity of any key within the keystore affected by the function is maintained as a part of the function. This may include provisioning, revocation, replacement, migration, injection, or negotiation.

[MANDATORY] REQ_6.2.3.7_20: The hardware protected security environment initial key material shall be provisioned in the hardware protected security environment securely at some point during initial product manufacture.

NOTE: Subsequent key management can then be authenticated by this initial key material.

[MANDATORY] REQ_6.2.3.7_30: The hardware protected security environment shall perform secure key management in accordance with the principles outlined in NIST SP 800-57. Hardware protected security environment-centric key management objectives should satisfy the requirements provided in FIPS 140-2.

[MANDATORY] REQ_6.2.3.7_60: The hardware protected security environment shall maintain the integrity of at least one root of trust at any time throughout the life of the hardware protected security environment, i.e., it should not allow deletion of all sources of trust during normal operation of the product.

[MANDATORY] REQ_6.2.3.7_70: The hardware protected security environment shall not use unchecked raw asymmetric keys.

[OPTIONAL] REQ_6.2.3.7_80: The hardware protected security environment shall maintain a cryptographically enforced chain of trust throughout the hardware protected security environment's provisioning supply chain according to distinct lifecycle states.

[OPTIONAL] REQ_6.2.3.7_100: The hardware protected security environment shall ensure that each transition between lifecycle states is cryptographically authenticated by the allowed, designated party.

[MANDATORY] REQ_6.2.3.7_110: The hardware protected security environment shall support a key management interface (example interfaces include key provisioning, revocation, replacement, migration, input/output, and negotiation) that controls the function and use of all cryptographic keys within the hardware protected security environment. Example interfaces include key provisioning, revocation, replacement, migration, input/output, and negotiation.

[MANDATORY] REQ_6.2.3.7_120: If the hardware protected security environment requires a common time as an input as a pre-condition to the security of any key management operation, the authenticity of the time signal and the authorization of its source must be confirmed within the hardware protected security environment before it is valid for use.

[MANDATORY] REQ_6.2.3.7_130: The hardware protected security environment shall prevent replay and rollback schemes which attempt to perform an unauthorized update of a key (for example, countermeasures may include a secure counter or a secure time source).

6.2.3.7.1 Key Management and Provisioning: Required

The secure key management and provisioning scheme is a system mechanism that is external to the function to load or install a key in the device without any knowledge of the key being disclosed. Use of the keystore within the hardware protected security environment inherently requires a secure key provisioning algorithm. The firmware of the hardware protected security environment needs to implement all client mechanisms that are specified to take place within this procedure. These may include the secure provisioning, management, generation, derivation, migration, and zeroizing of keys, all of which must be accomplished without disclosing the keys.

Key refresh cycles (specific validity periods and frequency of renewal of keys) are beyond the scope of this document and can be determined by the application design requirements. However, the hardware protected security environment needs to meet a performance specification for useful frequency of generation of keys according to the key management plan to prevent alternate generation schemes in the normal environment.

[OPTIONAL] REQ_6.2.3.7.1_10: The hardware protected security environment shall authenticate the key provisioning entity(e.g., by use of a PKI).

NOTE: The hardware protected security environment therefore may need to have the public key of the accepted certificate authority(s) pre-provisioned.

NOTE: If knowledge of any existing root keys is discovered, an attacker can change those keys and recovery may not be possible. The compromise of the existing root key may be beyond the scope of hardware protected security; however, the hardware protected security environment should provide resources to make such a compromise difficult through defense in depth (reputation, multi-factor authentication, resistance to side channel attacks, etc.).

NOTE: It is recommended to avoid creating a system where if the root of trust stored on one instance of a part is discovered, then control of the keys to all other parts is compromised.

NOTE: An asymmetric scheme keeps the root of trust outside the part storing only non-confidential public keys on the part making it impossible to determine any secrets in regards to a part from another part of the same product line. For example, a keyfob is particularly vulnerable because one sample may be thoroughly attacked. Any secrets revealed during the attack should not enable the theft of other vehicles.

NOTE: The choice of the use of long term keys versus temporary keys, use of primary keys versus derived keys, definitions of rolling codes, counters, or usage windows are beyond the scope of this specification. However, the implementation of session management must leverage access controls and validity controls implemented within the hardware protected security environment.

[MANDATORY] REQ_6.2.3.7.1_20: The hardware protected security environment shall prevent replay and rollback schemes which target an update of a key. For example, such support may include a secure counter or a secure time source, as well as key revocation schemes.

Some key provisioning algorithms use a common time as an input, if such is included the authenticity of the time signal must be confirmed within the hardware protected security environment before it is valid for use.

[MANDATORY] REQ_6.2.3.7.1_40: The hardware protected security environment shall check the authorization of the external entity within the hardware protected security environment prior to the execution of the requested function.

NOTE: Certain schemes may necessitate provisioning keys received through a diagnostic (aka, vehicle service) session without exposing any secrets, as the technician may not be a trusted entity.

NOTE: The challenge of key provisioning is to also maintain a chain of trust through the delivery scheme. The end-to-end protocol is beyond the scope of SAE J3101. The hardware protected security environment must define requirements to verify the trusted source when provisioning keys.

[MANDATORY] REQ_6.2.3.7.1_50: The hardware protected security environment shall protect all key exchange operations from unknown key-share (UKS) attacks. [Blake-Wilson; Menezes]

[MANDATORY] REQ_6.2.3.7.1_60: If the hardware protected security environment performs key derivation and supports a policy restricting the number of updates (derivations) that may be performed with a key without rekeying, then the hardware protected security environment shall track and associate the number of updates with the key.

6.2.3.7.2 Key Generation: Conditional

Key Generation: Key generation is the generation of a symmetric key or an asymmetric key pair using a deterministic random bit generator (DRBG) or random number generator (RNG). Key generation and key migration should only be performed within the hardware protected security environment using entropy sources within the hardware protected security environment.

[MANDATORY] REQ_6.2.3.7.2_10: The hardware protected security environment shall have a certified RNG for key generation.

NOTE: Vendor selection and adherence to specific key generation algorithms is beyond the scope of this document, but may be dictated by other industry or regulatory requirements.

[MANDATORY] REQ_6.2.3.7.2_20: The hardware protected security environment shall perform all key generation functions within the boundary of the hardware protected security environment.

NOTE: Key generation function is seeded by a random number, and the key is generated locally.

[MANDATORY] REQ_6.2.3.7.2_50: RNG seeds of sufficient minimum entropy shall be used that match generated key's expected computational resistance to attack (i.e., a 128 bit minimum entropy seed is needed to seed a RNG generating a 128 bit key).

[OPTIONAL] REQ_6.2.3.7.2_90: Key generation shall be defined as a local operation to then be distinct from key provisioning or key derivation.

6.2.3.7.3 Key Establishment: Conditional

Key establishment may include key agreements, key negotiation, key import, or other means of determining cryptographic keys.

1. Key Agreement: The hardware protected security environment agrees on a cryptographic key with another entity based on pre-shared secrets, symmetric keys typically through use of a key derivation function.
2. Key Negotiation: The hardware protected security environment calculates a cryptographic key using asymmetric public keys of key group members. Diffie-Hellman is an example of a key negotiation protocol.
3. Key Import: The hardware protected security environment receives a key from outside the hardware protected security environment boundary. Key import of secret and private keys needs to utilize encryption, integrity, and authentication controls over the imported key.

[MANDATORY] REQ_6.2.3.7.3_10: While this document does not mandate allowed key agreement protocols, the hardware protected security environment shall employ authentication in all supported key agreement protocols.

[MANDATORY] REQ_6.2.3.7.3_20: The hardware protected security environment shall ensure that all key encryption (“key wrapping”) algorithms (which can be symmetric or asymmetric) are always at least as strong as the computational resistance to attack of the imported key.

NOTE: Example key wrap algorithms include AES-WRAP (symmetric) and RSA public key encryption (asymmetric).

NOTE: Vendor selection and adherence to specific key establishment protocols is beyond the scope of this document, but could be dictated by other industry or regulatory requirements.

6.2.3.7.4 Key Derivation: Conditional

Key Derivation: Key derivation refers to the process of deriving one cryptographic key by inputting another cryptographic key or secret parameter into a key derivation algorithm (function). Key derivation is used in a variety of cryptographic protocols and schemes. It is frequently used in key rollover schemes or in key agreement protocols that need to derive a usable key from an established master secret. Key derivation is a symmetric process typically used by two different parties that need to derive the same key from a common known secret. The hardware protected security environment protects the inputs to the key derivation function just as it protects any other secret key. A key derivation function is used when two entities share a common secret and derive the same key, particularly for symmetric algorithms. The hardware protected security environment derives the key from that shared secret without exposing either the shared secret or the derived key.

NOTE: Adherence to specific key derivation algorithms is beyond the scope of this document, but may be dictated by other industry or regulatory requirements.

[OPTIONAL] REQ_6.2.3.7.4_10: The hardware protected security environment shall manage and protect the inputs to a cryptographic key derivation algorithm as a secret key.

[OPTIONAL] REQ_6.2.3.7.4_20: The hardware protected security environment shall reset all inputs to the key derivation function as soon as they are no longer needed.

NOTE: One method of key reset is to zeroize the key, but this is not possible in some memory technologies. [FIPS140-3]

[MANDATORY] REQ_6.2.3.7.4_30: The hardware protected security environment shall support update of key derivation algorithms during the lifetime of the hardware protected security environment, unless the product is stated to be of limited use (see 7.6).

NOTE: Given long automotive lifetime requirements, key derivation functions could become obsolete.

6.2.3.7.5 Key Migration: Conditional

Key Migration: Key migration refers to the process and rules associated with directly importing or exporting a cryptographic key to/from a keystore within a hardware protected security environment. If the key is not locally migrated (e.g., via a direct, locally attached key loading device and/or human user), it is referred to as key transport (subset of key establishment). When secret or private keys are input or output from a secure hardware protected security environment boundary, they need to be encrypted (wrapped). Key encryption (wrapping) mechanisms provide a method for protecting the confidentiality of the key.

Secure key migration schemes can be necessary in order to replace hardware.

Example use case of secure key migration would be keyless entry ECU replacement, while allowing existing key fobs to be maintained.

Another example use case is general key distribution to migrate the public key to install a different ECU.

Key migration allows one entity to generate a key and migrate it to a different entity. This is of particular interest with a shared secret key (symmetric) that is migrated for validation purposes only (e.g., HMAC).