Draft ETSI EN 304 623 V0.1.3 (2026-06)

HARMONISED EUROPEAN STANDARD

Cyber Security (CYBER);
CRA;
Cybersecurity requirements for boot managers

Reference
DEN/CYBER-EUS-008

Keywords
CRA, cybersecurity

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This draft Harmonised European Standard (EN) has been produced by ETSI Technical Committee Cyber Security (CYBER), and is now submitted for the combined Public Enquiry and Vote phase of the ETSI Standardisation Request deliverable Approval Procedure (SRdAP).

The present document has been prepared under the Commission's standardisation request C(2025) 618 [i.3] final to provide one voluntary means of conforming to the requirements of Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements and amending Regulations (EU) No 168/2013 and (EU) No 2019/1020 and Directive (EU) 2020/1828 (Cyber Resilience Act) (CRA) [i.1].

Once the present document is cited in the Official Journal of the European Union under that Regulation, compliance with the normative clauses of the present document given in Table A.1 confers, within the limits of the scope of the present document, a presumption of conformity with the corresponding requirements of that Regulation and associated EFTA regulations.

Proposed national transposition dates	
Date of latest announcement of this EN (doa):	3 months after ETSI publication
Date of latest publication of new National Standard or endorsement of this EN (dop/e):	6 months after doa
Date of withdrawal of any conflicting National Standard (dow):	18 months after doa

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

The present document is a European harmonised standard that defines technical cybersecurity requirements for boot managers as products with digital elements according to Regulation (EU) 2024/2847 [i.1], the Cyber Resilience Act (CRA).

How to use the present document

Clause 4 describes the product context: architecture, capabilities, operational environment, and use cases. This clause is informative and contains no requirements.

Clause 4.6 defines three use cases for boot manager products (UC-IMM, UC-VER and UC-HW), each associated with a security profile (LOW, MEDIUM or HIGH).

The manufacturer declares the use case matching the product's capabilities as specified in Table 4.6.4-1.

Clause 5 contains the normative technical requirements. Requirements apply based on the declared security profile and on the functional capabilities the product implements. Clause 5.1 specifies the applicability rules.

Clause 6 defines assessment criteria for each requirement.

Annex A maps requirements in the present document to CRA essential requirements.

Annex B documents the security analysis underlying the requirements.

Annex C contains information on the product documentation.

Annex D describes the relation to physical attack resistance.

Annex E provides an informative mapping to NIST SP 800-193 [i.10]

Annex K defines generic requirements and assessment criteria for the use of state-of-the-art cryptography. This Annex is normative.

1 Scope

1.1 General

The present document specifies technical cybersecurity product requirements and corresponding assessment criteria for boot managers. The products with digital elements in scope, there after "the products":

- are specified within the technical description of the category of product number 8 by the Commission Implementing Regulation (EU) 2025/2392 [i.2] as:
  - "Software products with digital elements that manage the process of initial system startup after power on/restart by initialising hardware, loading or transferring control to the operating system environment or system resources, and selecting boot options. This category includes but is not limited to UEFI firmware, single-stage and multi-stage boot loaders."
- are only covered within the product context described in clause 4.

The scope covers software and firmware components that manage the boot process from power-on through establishment of the chain of trust to handoff to the boot target. Products in scope include boot management software and firmware regardless of distribution model or integration level. These are:

- **System firmware** that performs hardware initialisation and boot management.
- **Bootloaders** that manage boot target selection, verification, and loading.
- **Embedded boot firmware** in IoT and embedded devices.
- **Network boot implementations** enabling remote boot capabilities.
- Boot managers that **integrate with hardware security components** for chain of trust establishment.

NOTE 1: Boot managers may be single-stage (direct loading) or multi-stage (staged verification).

NOTE 2: For microcontrollers (MCUs) and microprocessors (MPUs):

- **Silicon-integrated immutable firmware:** Mask ROM, fused code, or boot firmware integrated during chip manufacturing is assessed as part of MCU/MPU hardware under semiconductor standards.
- **Updateable boot managers:** Boot software in flash storage (including OTP programmed post-manufacture) is assessed using the present document when distinctly identifiable or independently updatable.

NOTE 3: Runtime services executing after boot target handoff (such as secure monitor mode handlers or attestation services) are in scope only if they provide verification or attestation services to the boot process itself, not to the boot target.

The present document covers those products to demonstrate compliance with essential cybersecurity requirements in the Regulation (EU) 2024/2847 [i.1], Annex I Part I under the conditions identified in Annex A.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found in the ETSI docbox.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] ENISA Report 1747792503 (version 2 - April 2025): "European Cybersecurity Certification Group Sub-group on Cryptography Agreed Cryptographic Mechanisms".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents may be useful in implementing an ETSI deliverable or add to the reader"s understanding but are not required for conformance to the present document.

[i.1] Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements and amending Regulations (EU) No 168/2013 and (EU) 2019/1020 and Directive (EU) 2020/1828 (Cyber Resilience Act).

[i.2] Commission Implementing Regulation (EU) 2025/2392 of 28 November 2025 on the technical description of the categories of important and critical products with digital elements pursuant to Regulation (EU) 2024/2847 of the European Parliament and of the Council.

[i.3] Standardisation request M/606 - C(2025)618: Commission Implementing decision of 3.2.2025 on a standardisation request to the European Committee for Standardisation (CEN), the European Committee for Electrotechnical Standardisation (Cenelec) and the European Telecommunications Standards Institute (ETSI) as regards products with digital elements in support of Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements and amending Regulations (EU) No 168/2013 and (EU) 2019/1020 and Directive (EU) 2020/1828 (Cyber Resilience Act).

[i.4] prEN 40000-1-3: "Cybersecurity requirements for products with digital elements - Part 1-3: Vulnerability handling' (produced by CEN CENELEC).

NOTE: Version and date to be added upon its publication by CEN CENELEC.

[i.5] prEN 40000-1-2: "Cybersecurity requirements for products with digital elements - Part 1-2: Generic Security Framework ' (produced by CEN CENELEC).

NOTE: Version and date to be added upon its publication by CEN CENELEC.

[i.6] EN 17927:2023: "Security Evaluation Standard for IoT Platforms (SESIP)".

[i.7] prEN 50765:2026: "Cybersecurity requirements for microprocessors and microcontrollers with security-related functionalities".

[i.8] prEN 50766:2026: "Cybersecurity requirements for tamper-resistant microprocessors and microcontrollers".

[i.9] ISO/IEC 15408: "Common Criteria for Information Technology Security Evaluation".

[i.10] NIST SP 800-193: "Platform Firmware Resiliency Guidelines", National Institute of Standards and Technology.

[i.11] CAPEC™: "Common Attack Pattern Enumeration and Classification", MITRE Corporation.

[i.12] MITRE EMB3D™: "Embedded Device Security Threat Model", MITRE Corporation.

[i.13] CWE™: "Common Weakness Enumeration", MITRE Corporation..

[i.14] UEFI Forum: "UEFI Specification Version 2.11 (December 2024)".

[i.15] Trusted Computing Group®: "TPM 2.0 Keys for Device Identity and Attestation v1.10".

[i.16] Trusted Computing Group®: "DICE Layering Architecture Version 1.0".

[i.17] Trusted Computing Group®: "DICE Attestation Architecture v1.2".

[i.18] Federal Office for Information Security, Germany: "BSI TR-02102 Cryptographic Mechanisms".

[i.19] National Cybersecurity Agency, France: "ANSSI RGS, Annex B".

[i.20] Canadian Centre for Cyber Security: "CCCS ITSP.40.111".

[i.21] ETSI TS 119 312 (V1.5.1): "Electronic Signatures and Trust Infrastructures (ESI); Cryptographic Suites".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in Regulation (EU) 2024/2847 [i.1] and the following apply:

attestation: process of providing cryptographic evidence about boot state, configuration, and measurements to a local or remote verifier

NOTE: Attestation enables trust decisions based on demonstrated boot integrity without necessarily preventing execution of unverified code.

EXAMPLE: Attestation may include measurement logs, signed attestation statements, and cryptographic proofs of boot component identity

authorised entity: person, system, or process permitted to perform security-relevant operations on the boot manager, where the permission depends on the lifecycle phase and the configured access control policy

NOTE 1: An authorised entity may be a manufacturer, integrator, administrator, or end user depending on the lifecycle phase.

NOTE 2: Security-relevant operations may include configuration changes, trust anchor modification, update installation, and recovery mode activation.

boot component: any executable code, firmware module, configuration data, or cryptographic material loaded and used during the boot process

boot device: any storage medium or interface from which boot code can be loaded and executed, including internal storage, removable media, or network sources

boot manager: software or firmware component that controls the boot process and allows management of multiple boot targets or boot configurations after power-on or reset

NOTE: This term includes both single-purpose bootloaders and multi-function firmware with boot management capabilities.

boot sequence: ordered series of operations from power-on through hardware initialisation, boot manager execution, and handoff to the boot target

boot source: logical origin from which boot code can be loaded and executed

EXAMPLE: Boot devices, network interfaces, and debug interfaces.

NOTE: Each source presents different security boundaries and threat exposures.

boot target: software component to which the boot manager transfers control upon successful completion of its designated function, such as an operating system kernel, hypervisor, another bootloader, or an embedded application

certificate: data structure that associates a public key with an identity, optionally including usage constraints, validity period, and other attributes

NOTE: A certificate may be signed by a certification authority or may be self-signed. In the context of the present document, "certificate" refers to both ITU-T X.509-compliant certificates and minimal public key serialisations, as specified in the relevant requirements.

chain of trust: sequential verification where each boot component validates the next, also known as transitive trust or trust chain, establishing security from hardware root to the boot target

critical data: data whose corruption would prevent successful boot completion or compromise security verification, including boot code, verification keys, anti-rollback counters, and security policy settings

cryptographic mechanism: security-related procedure using cryptography

NOTE 1: The term "cryptographic mechanism" is used in this annex as an umbrella term covering the categories used in the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [reference], including cryptographic algorithms, primitives, schemes, protocols, protocol profiles, cipher suites, modes of operation, constructions and parameter sets.

NOTE 2: The terms "cryptographic primitive", "cryptographic construction", "cryptographic scheme" and "cryptographic protocol" are used consistently with the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [reference].

NOTE 3: A cryptographic mechanism can be specified at different levels of abstraction. For example, AES is a cryptographic primitive, AES-GCM is a mode of operation / authenticated encryption construction, TLS 1.3 is a protocol, a TLS 1.3 restricted cipher-suite profile is a protocol profile, and TLS_AES_128_GCM_SHA256 is a cipher suite.

cryptographic property: property provided or supported by a cryptographic mechanism

NOTE 1: Cryptographic properties include, for example, confidentiality, integrity, authenticity, entity authentication, message authentication, key establishment, key confirmation, replay protection, collision resistance, second-preimage resistance, preimage resistance, signature unforgeability, non-repudiation, forward secrecy and password-verifier protection. Where relevant, a cryptographic property can be expressed using a formal cryptographic notion, for example IND-CPA, IND-CCA, IND-CCA2, EUF-CMA, SUF-CMA or authenticated key exchange security.

NOTE 2: A cryptographic property is distinct from a product-level technical requirement, although it can support such a requirement. For example, a secure communication requirement can rely on cryptographic properties such as confidentiality, integrity and authentication.

defence-in-depth: security strategy employing multiple, overlapping, and diverse protection mechanisms so that compromise of any single mechanism does not result in complete security failure

NOTE: In the present document, "threat" is used synonymously with "cyber threat" as defined in Regulation (EU) 2024/2847 [i.1].

enforcement: active prevention of unauthorised actions through cryptographic verification and blocking of execution when verification fails, as distinct from detection-only approaches that record but do not prevent execution

essential operation: operation required for the boot manager to complete its primary function of loading and transferring control to a boot target

evaluation activity: assessment procedure addressing a group of related requirements

firmware: low-level software stored in non-volatile memory on the device/product that provides hardware initialisation and runtime services, including boot managers

firmware runtime services: functions provided by boot manager firmware that remain accessible to the operating system after boot handoff, typically including variable storage, time services, and system management functions

handoff: transfer of execution control from the boot manager to the boot target, marking the completion of the boot manager's function and the beginning of boot target execution

hardware security component: generic term for hardware-based security modules that provide cryptographic services, secure storage, or isolated execution environments

NOTE 1: TPM and HSM are examples of hardware security components. The generic term 'hardware security component' is used throughout the present document unless referring to specific implementation examples.

NOTE 2: The term also includes immutable storage and protection provided by code-signed firmware (for example trust anchors embedded in mask ROM or in write-protected firmware regions whose authenticity is verifiable), where the integrity and where applicable confidentiality properties of such storage are equivalent to those of dedicated hardware security modules.

immutable component: component whose code cannot be modified after manufacture or initial programming, typically implemented in Read-Only Memory (ROM), One-Time Programmable (OTP) storage, e-fuses, or write-protected flash memory

measured boot: recording cryptographic measurements of boot components before execution for attestation without blocking execution

monotonic counter: counter value that can only increase, never decrease, used for anti-rollback protection to make version downgrade detectable

network boot: loading boot components from network sources using protocols such as PXE or HTTPS Boot

physical presence: requirement for a user to be physically at the device location with direct physical access to the device to perform certain security-sensitive operations, typically verified through local keyboard input or button presses

NOTE: Physical presence can also encompass operations conducted in a controlled environment such as a manufacturing plant.

recovery boot: specialised boot process designed to restore system functionality after failures

rollback protection: security mechanism preventing unauthenticated installation of older firmware versions that may contain known vulnerabilities

root of trust: component providing foundational trust for the system through hardware-protected execution and storage

NOTE: Root of Trust is typically implemented in immutable ROM, hardware security modules, or physically isolated processors. Serves as the anchor for chain of trust verification.

secure boot: specific UEFI implementation of verified boot as defined in the UEFI Specification [i.14]

security-critical configuration: configuration settings whose modification would reduce security posture, including verification enable/disable settings, trust anchor databases, certificates, user-enrolled keys, boot order, and recovery mode settings

sensitive data: data requiring protection from disclosure

EXAMPLE: Sensitive data include cryptographic private keys and symmetric keys.

NOTE: Unauthenticated access to sensitive data enables impersonation, forgery, or decryption of protected content.

trust anchor: public key, certificate, or cryptographic hash that serves as the starting point for verification and is trusted without requiring further validation

update capability: ability to modify boot manager code, configuration, or security policies after initial deployment, ranging from configuration-only updates to full firmware replacement

verified boot: cryptographic verification of boot components before execution to ensure only authorised code runs during system initialisation

NOTE: Implementations verify authenticity using cryptographic signatures or authenticated hashes bound to cryptographic keys or hardware roots of trust.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ACM	Agreed Cryptographic Mechanisms
ACPI	Advanced Configuration and Power Interface
AES	Advance Encyption Standard
ASLR	Address Space Layout Randomisation
ATM	Automated Teller Machine
BIOS	Basic Input/Output System
CDI	Compound Device Identity
CMAC	Cipher-based Message Authentication Code
CPU	Central Process Unit
CRC	Cyclic Redundancy Check
CRL	Certificate Revocation List
CSA	Cyber Security Act
CVE	Common Vulnerabilities and Exposures
DHCP	Dynamic Host Configuation Protocol
DICE	Device Identifier Composition Engine
DMA	Direct Memory Access
DPA	Differential Power Analysis
ECC	Error-Correcting Code
ECCG	European Cyber Security Certification Group
ECDSA	Elliptic Curve Digital Signature Algorithm
EdDSA	Edwards-curve Digital Signature Algorithm
eMMC	embedded Multi-Media Card
EUVD	EU Vulnerability Database
FEC	Forward Error Correction
GCM	Galois Counter Mode
GHASH	Galois Hash
HKDF	HMAC-based Extract-and-Expand Key Derivation Function
HMAC	Hash-based Message Authentication Code
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IoT	Internet of Things
IPMI	Intelligent Platform Management Interface
iSCSI	internet Small Computer System Interface
ISO	International Organization for Standardisation
JTAG	Joint Test Action Group
KEV	Known Exploited Vulnerabilities
KMAC	Keccak Message Authentication Code
LAN	Local Area Network
MAC	Message Authentication Code
MCU	Microcontroller Unit
MPU	Microprocessor Unit
NVMe	Non-Volatile Memory express™
NVRAM	Non-Volatile Random Access Memory
OCSP	Online Certificate Status Protocol
OEM	Original Equipment Manufacturer

OS	Operating System
OTP	One-Time Programmable
PCR	Platform Configuration Register
PDR	Protection, Detection, Recovery
PIN	Personal Identification Number
PXE	Preboot Execution Environment
RAM	Ramdom Access Memory
ROM	Read-Only Memory
RSA	Rivest, Adleman and Shamir

NOTE: Algorithm invented by Rivest, Adleman and Shamir.

RTD	Root of Trust for Detection
RTU	Root of Trust for Update
SATA	Serial Advanced Technology Attachment
SBOM	Software Bill of Materials
SESIP	Security Evaluation Standard for IoT Platforms
SMM	Security Monitoring and Management
SPI	Serial Peripheral Interface
SWD	Serial Wire Debug
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TOCTOU	Time-Of-Check-Time-Of-Use
TPM	Trusted Platform Module
UART	Universal Asynchronous Receiver and Transmitter
UEFI	Unified Extensible Firmware Interface
USB	Universal Serial Bus
VPN	Virtual Private Network
WiFi	Wireless Fidelity (synonym for WLAN)

4 Product context

4.1 Product Functions

4.1.1 Intended purpose

Boot managers establish initial system trust and manage the boot process from power-on through handoff to the boot target. They serve as the foundational trust component that executes before operating systems or security services are available.

The intended purpose includes:

- Establishing the initial chain of trust for the system.
- Loading and transferring control to boot targets (such as another bootloader, operating systems, hypervisors, or embedded applications).
- Managing boot configuration and security policy.

Where capable it also includes:

- Verifying integrity of boot components.
- Authentication of boot components.
- Recording measurements for attestation.
- Providing recovery mechanisms.

4.1.2 Reasonably foreseeable use

Boot managers are components that may be:

- Integrated into diverse hardware platforms across all industry verticals.
- Deployed in security contexts ranging from consumer devices to critical infrastructure.
- Configured by users with varying levels of technical expertise.
- Operated in environments ranging from physically secure to hostile and untrusted.
- Transferred to secondary markets without factory reset.
- Operated beyond the manufacturers active support lifecycle.
- Combined with other components in composite products.
- Subjected to configuration changes by downstream integrators.
- Used in safety-critical systems where availability requirements may conflict with security update procedures.

These reasonably foreseeable uses inform the requirements in clause 5 and risk analysis methodology in Annex B.

Reasonably foreseeable deployment contexts determine risk level determination as described in clause B.2. Boot managers are commonly composed of components from multiple sources including silicon vendors, independent BIOS vendors, OEMs, and open-source projects. Each component may have distinct update mechanisms, vulnerability response capabilities, and maintenance lifecycles.

4.1.3 Reasonably foreseeable misuse

Reasonably foreseeable misuse includes use of the boot manager in ways not intended but predictable from human behaviour or system interactions. Examples include:

- Disabling security features for convenience
- Using development/debug modes in production
- Operating without applying available security updates
- Connecting to untrusted networks or storage

Requirements in clause 5 address these scenarios through secure defaults and fail-safe behaviour.

4.1.4 General functions

Boot managers establish system trust and load the boot target. This process includes verifying or measuring boot components, managing boot configuration, and transferring execution control to the selected target.

Security functions vary by design. Verified boot implementations block execution of unverified code through cryptographic signature verification. Measured boot implementations record cryptographic measurements of boot components for attestation purposes. Both approaches protect boot configuration from unauthorised modification and may include rollback protection against version downgrade attacks.

Configuration capabilities determine how boot behaviour can be modified. Some implementations have fixed behaviour determined at manufacturing. Others allow modification of security policies during subsequent boots, cryptographic credentials, and boot target selection. Boot managers with configuration capability implement access controls to prevent unauthorised changes.

Lifecycle management includes update, recovery, and reset functions. Updates modify boot manager firmware or configuration post-deployment. Recovery functions provide alternative boot paths when primary operations fail. Reset capabilities support secure decommissioning by erasing cryptographic material and restoring default configurations, where technically feasible. However, when cryptographic material or configuration data are stored in immutable components such as e-fuses or ROM, complete erasure or restoration may not be possible.

4.2 Product Architecture

4.2.1 Boot architecture

4.2.1.1 Unverified boot

Unverified Boot implements sequential stage execution without cryptographic verification mechanisms.

The system begins with power-on, proceeds through hardware initialisation, and continues with sequential stage execution where each stage directly loads and executes the next until the boot target is reached.

Unverified boot implements no cryptographic authentication mechanisms, but includes simple integrity checks such as CRC, checksums or hashes. These mechanisms may detect accidental corruption but do not provide protection against deliberate modification or substitution attacks.

Unverified boot is appropriate for environments where physical security controls provide the primary protection, or for device categories (e.g. sensors) where risk assessment determines that the assets stored or transmitted do not warrant verified boot implementation.

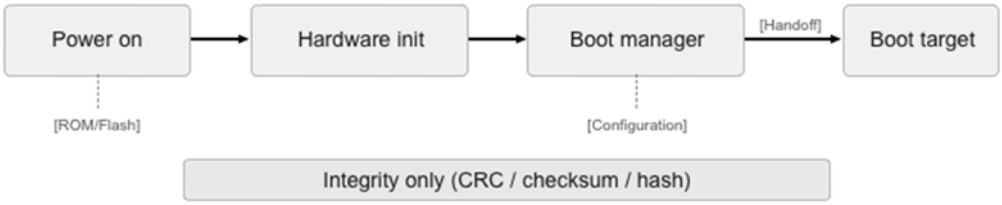

Figure 4.2.1.1-1

4.2.1.2 Verified boot

Verified boot implements cryptographic verification with enforcement throughout the boot process. Components are cryptographically verified before execution, creating a verification chain from power-on through boot target execution.

NOTE 1: Verification may be performed either immediately before execution (on-boot verification) or prior to installation with integrity-protected storage retained between updates (on-update verification, also referred to as capsule update). Both patterns establish that the code executed on the device has been cryptographically verified at some point in time before its execution. A boot manager may implement either pattern or a combination of both.

Components are verified against stored cryptographic signatures or authenticated hash values and if verification fails the system halts the boot process and potentially triggers recovery mode. Initial verification material (keys or hashes) may be stored in hardware-protected memory, fuses, or one-time programmable storage. Systems implementing verified boot need cryptographic verification capabilities and secure storage for verification material. Verified boot employs cryptographic authentication mechanisms using either:

- Asymmetric cryptography: Digital signatures verified using public keys stored in the trust root (e.g. RSA, ECDSA, EdDSA signatures)
- Symmetric cryptography: Message Authentication Codes (MACs) or authenticated encryption verified against pre-shared secrets (e.g. HMAC, CMAC, AES-GCM, KMAC)
- One-way functions, such as cryptographic hash functions, which are a necessary component in signature systems, and map data and public keys to protected trust anchors

NOTE 2: See Annex K for approved cryptographic mechanisms and deprecation schedules.

When correctly applied, either together or separately, these mechanisms provide cryptographic assurance that boot components have not been modified and originate from a trusted source. When verification fails, the system halts execution and may transition to a degraded secure state, trigger recovery mode, or perform a power-reset to prevent execution of unauthorised code.

The choice between asymmetric and symmetric authentication schemes, including the use of authenticated hashes and digital signatures, depends on the threat model, key management capabilities, available cryptographic hardware support, and performance requirements particularly in safety-critical contexts where boot time is constrained.

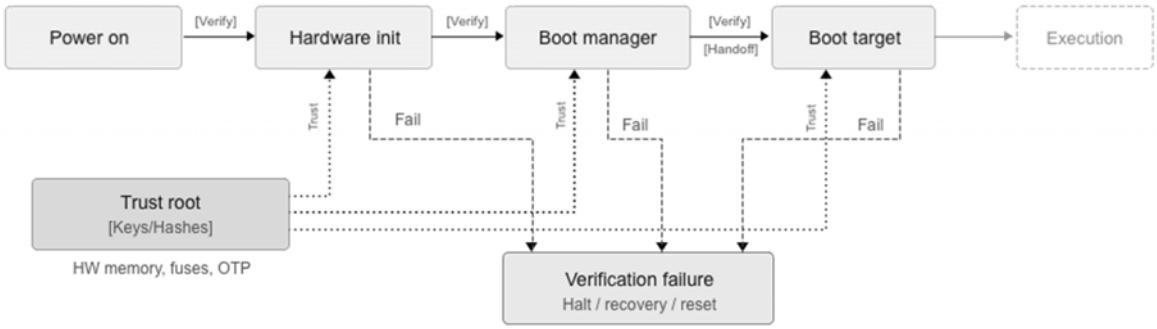

Figure 4.2.1.2-1

4.2.1.3 Measured boot

Measured boot records cryptographic measurements of each boot stage without preventing execution. During initialisation, the system activates measurement capabilities and records a cryptographic hash of each subsequent stage into secure storage.

Each stage computes a cryptographic hash of the next component and stores or extends it into designated registers, and the boot process continues regardless of measurement values. The resulting measurement log provides an attestable record of the boot sequence that can be verified against known-good values or reference measurements.

This architecture relies on hardware or firmware capable of secure measurement process, storage and reporting. The fundamental difference from verified boot is in the enforcement policy. The system records measurements in the log but continues the boot process regardless of values. This allows:

- Attestation of the actual boot state to remote verifiers.
- Post-boot analysis and remediation decisions.
- Operation in degraded mode with logged anomalies.
- Flexibility for development and recovery scenarios.
- Execution of parts of the system software conditional on the measured stages being valid.

The measurement log provides a cryptographically protected, auditable record that can be verified locally or remotely against known-good reference values or security policies.

Measured boot employs the same cryptographic authentication mechanisms as verified boot.

EXAMPLE: The diagram below illustrates the general measured boot architecture.

Each boot stage computes a cryptographic hash and records it into hardware-protected storage (hardware-based Root of Trust), which may be implemented using a variety of technologies including PCRs in a TPM [i.15], CDI derivation in DICE [i.16] and [i.17], or other secure measurement storage mechanisms.

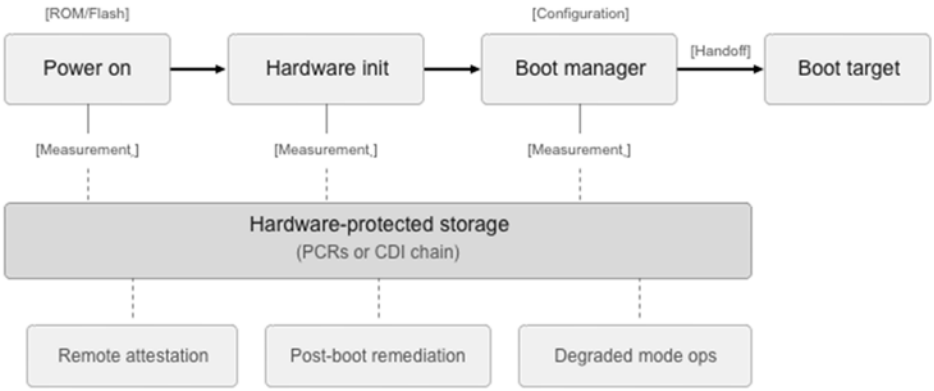

Figure 4.2.1.3-1

4.2.2 Functional extensions

Boot managers may provide additional functions beyond core boot architecture. These functions are classified as either security-enhancing or functional (see clause 4.2.3).

Security-enhancing capabilities improve the security posture of the boot manager and characterise the use cases defined in clause 4.6.

Functional capabilities provide features that may be required by the product's intended purpose but introduce additional attack surface. Where a functional capability is implemented, the requirements in clause 5 that reference that capability apply. Both types of capability can be combined with any boot architecture (unverified, verified, or measured).

4.2.3 Product capabilities

4.2.3.1 Introduction

The following clauses describe each capability. Table 4.2.3.1-1 summarises the classification defined in clause 4.2.2. Where applicability statements in clause 5 refer to boot managers with a particular capability, they apply to boot managers implementing any option listed in the corresponding clause other than the "no [capability]" option, unless the clause defines a different threshold.

Table 4.2.3.1-1: Capability classification

Security-enhancing	Functional
Verified boot (clause 4.2.1.2)	Network boot (clause 4.2.3.3)
Update capability (clause 4.2.3.2)	Configuration management (clause 4.2.3.4)
Logging capability (clause 4.2.3.7)	Measured boot (clause 4.2.1.3)
Hardware security (clause 4.2.3.5)	Recovery capability (clause 4.2.3.6)
Key provisioning (clause 4.2.3.10)	Authentication capability (clause 4.2.3.8)
	Debug interface (clause 4.2.3.9)

4.2.3.2 Update capability

Update capability describes whether and how the boot manager can be modified after initial deployment.

- **No updates:** Boot manager code and configuration cannot be modified after manufacture.
- **Configuration updates:** Only boot configuration, security policies, and trusted keys can be updated. Boot manager code remains immutable.
- **Partial updates:** Specific components can be updated while core components remain immutable. Root of trust and initial boot stages cannot be modified.

- **Full updates:** Complete boot manager firmware can be updated including all boot stages.
- **Runtime updates:** Boot manager can be updated while the system is running without requiring reboot until the next boot cycle.
- **Recovery-mode updates:** Updates can only be applied through a separate recovery mode or alternate boot path.
- **Network updates:** Boot manager can fetch and apply updates directly from network sources.
- **Enclave-managed updates:** Updates are managed by a separate security processor or trusted execution environment.

Products may combine multiple update capabilities (e.g. configuration updates via network, full updates via recovery mode), except products with no update capability. For the purposes of the present document, references to "update capability" in clause 5 apply to boot managers implementing partial updates, full updates, runtime updates, recovery-mode updates, network updates, or enclave-managed updates. Boot managers implementing only "no updates" or "configuration updates" are not considered to have update capability.

4.2.3.3 Boot source

Boot source identifies media types from which the boot manager can load and execute code.

- **Internal storage:** Boot exclusively from non-volatile storage integrated within the same package or die as the processor.
- **Fixed storage:** Boot exclusively from non-removable storage physically attached to the system but in a separate package.
- **Removable storage:** Can boot from removable storage devices such as USB, SD cards, or optical media.
- **Network boot:** Can boot from network sources including PXE, HTTP, iSCSI, or cloud services.
- **External expansion:** Can boot from external expansion buses such as Thunderbolt or ExpressCard.
- **Debug interfaces:** Can boot or load code via debug or programming interfaces such as JTAG, SWD, or UART.

Multiple boot sources are common. Each additional source increases the attack surface.

NOTE: Requirements referencing "Network boot" apply when the boot manager includes Network boot capability. Requirements referencing "Network update" apply when the boot manager includes Network updates under update capability. These may overlap but are distinct: Network boot loads the entire boot image from network while network update modifies persistent boot manager components.

4.2.3.4 Configuration management

Configuration capability describes whether boot behaviour can be modified through settings.

- **No configuration:** Fixed behaviour with no configurable settings, or configuration set once via one-time programmable mechanisms.
- **Static configuration:** Configuration can be set but involves physical access or special tools to modify.
- **Local configuration:** Configuration modifiable through local interfaces requiring physical presence.
- **Protected configuration:** Configuration modifiable with authentication but without physical presence requirement.
- **Remote configuration:** Configuration modifiable through network interfaces.
- **Runtime configuration:** Configuration modifiable by the operating system through runtime services.

For the purposes of the present document, references to "configuration capability" in clause 5 apply to boot managers implementing local, protected, remote, or runtime configuration. Boot managers implementing only "no configuration" or "static configuration" are not considered to have configuration capability.

4.2.3.5 Hardware security

Hardware security describes whether the boot manager utilises dedicated security hardware.

Software only: Boot manager operates entirely in software without dedicated security hardware.

Hardware storage: Utilises hardware components for protected key storage and cryptographic operations.

Hardware enforcement: Hardware actively enforces security policies during boot.

Hardware enforcement typically includes hardware storage.

4.2.3.6 Recovery capability

Recovery capability describes how the system can recover from boot failures, corruption, or attacks.

- **No recovery:** No separate recovery path or fallback boot mechanism.
- **Basic recovery:** Simple recovery mode or fallback boot option such as safe mode or previous known-good configuration.
- **Full recovery:** Comprehensive recovery environment with diagnostics, repair tools, and restore capabilities.

Recovery mechanisms provide resilience but introduce additional attack surface.

4.2.3.7 Logging capability

Logging capability describes whether and how the boot manager can record, store, and communicate security-relevant events and operational states during the boot process. The presence, type, and durability of logging functions depend on the boot managers architecture, resource constraints, and deployment context.

- **No logging:** The boot manager does not implement any logging functionality. No events or states are recorded during boot in persistent storage.

NOTE 1: This variant is typical for resource-constrained devices or early boot stages where logging infrastructure is not available.

- **Volatile logging:** The boot manager records events in volatile memory during boot. Logs are lost upon power-cycle or reset and are accessible only during the current boot session.

NOTE 2: Requirements referring to persistent log retrieval cannot be fulfilled when only volatile logging is implemented.

- **Persistent logging:** The boot manager records events in persistent storage (e.g. flash, NVRAM, dedicated log partition). Logs survive power cycles and can be accessed after reboot.
- **Hardware-protected logging:** The boot manager authorises hardware security components (e.g. TPM, secure element, write-once memory) to protect log integrity and prevent tampering.
- **External logging:** The boot manager exports log events to external interfaces (e.g. serial console, debug ports) for real-time monitoring or remote analysis.

Logging capability enables security monitoring and forensic analysis but uses storage resources.

4.2.3.8 Authentication capability

Authentication capability describes whether the boot manager involves identity verification for privileged operations:

- **No authentication:** No identity verification for any operations.

- **Physical presence:** Operations require physical presence detection (button press, jumper) but no credential verification.
- **Password authentication:** Operations require password or PIN verification.
- **Cryptographic authentication:** Operations require cryptographic credential verification (smart card, certificate, hardware token).
- **Multi-factor authentication:** Operations require multiple authentication mechanisms.

Authentication capability protects configuration and security policy changes from unauthorised modification.

4.2.3.9 Debug interface

Debug interface describes whether the boot manager exposes diagnostic or programming interfaces.

- **No debug interface:** No debug, diagnostic, or programming interfaces present in production.
- **Disabled debug:** Debug interfaces physically present but permanently disabled or fused off.
- **Authenticated debug:** Debug interfaces require authentication before activation.
- **Conditional debug:** Debug interfaces available only in specific modes (e.g. manufacturing, recovery).
- **Open debug:** Debug interfaces accessible without restriction.

Debug interfaces enable development and diagnostics but significantly increase attack surface when accessible. For the purposes of the present document, references to "debug capability" in clause 5 apply to boot managers implementing authenticated, conditional, or open debug interfaces. Boot managers with no debug interface, or with permanently disabled debug interfaces, are not considered to have debug capability.

4.2.3.10 Key provisioning

Key provisioning describes whether and how trust anchors used for boot verification can be provisioned by the integrator of the boot manager.

- **No key provisioning:** Trust anchors are fixed at manufacture and cannot be modified.
- **Manufacturing provisioning:** Trust anchors are provisioned during manufacturing or integration using dedicated tools or interfaces.
- **Updateable provisioning:** Trust anchors can be modified post-deployment through authenticated mechanisms.

Key provisioning does not imply that end users can modify trust anchors. It describes a mechanism available to the integrator. Whether the integrator exposes trust anchor management to downstream entities is outside the scope of the present document.

4.2.4 Resource constraints

Boot managers operate under inherent resource limitations determined by their implementation and hardware platform:

- **Storage:** Restricted firmware storage capacity shared with other components.
- **Computation:** No OS services, drivers, or shared libraries during early stages.
- **Time:** Boot performance requirements limiting security operations.
- **Memory:** Limited RAM available during early boot stages.

Resource constraints influence implementation choices and inform requirement applicability.

4.3 Operational environment

4.3.1 Introduction

The operational environment describes deployment context affecting risk factor assessment described in clause B.2. Environmental characteristics are independent of product characteristics and describe where and how the boot manager is deployed.

4.3.2 Physical security context

- **Secure:** Physically secure locations (data centres, secure facilities).
- **Controlled:** Controlled access environments (offices, industrial sites).
- **Restricted:** Limited public access with some monitoring (hospitals, retail).
- **Public:** Publicly accessible with minimal security (kiosks, ATMs, digital signage).
- **Hostile:** Uncontrolled, adversarial environments (consumer devices, field deployment).
- **Mobile:** Travels through varying physical environments (laptops, smartphones, tablets - should assume hostile).

4.3.3 Network exposure

Boot managers may be deployed with varying levels of network exposure:

- **None (air-gapped):** No network connectivity.
- **Isolated:** Private/isolated networks (LAN, industrial networks).
- **Managed:** Internet-connected within a managed environment (VPN, enterprise).
- **Internet:** Direct internet exposure.
- **Cellular:** Mobile/cellular networks.

4.3.4 Data sensitivity and value

The sensitivity and value of data accessible through or protected by the boot manager affects attacker motivation and consequence severity:

- **Minimal:** No significant sensitive data (basic sensors, simple IoT).
- **Personal:** Personal user data (consumer devices, personal files).
- **Financial:** Financial/payment data (payment terminals).
- **Business:** Business confidential data (enterprise systems, intellectual property).
- **Health:** Protected health information (medical devices, health records).
- **Critical:** Critical infrastructure control (power grid, water systems).
- **Classified:** Government classified information.

4.3.5 System criticality

Impact severity if the boot manager is compromised, considering availability, integrity, and safety:

- **Non-critical:** Low business impact, non-critical functions, general purpose devices.

- **Business impact:** Moderate business impact, internal systems, productivity loss, reputation damage.
- **Safety-critical:** Systems where compromise causes physical harm, critical infrastructure disruption, major financial loss or widespread service disruption.

4.3.6 Lifecycle expectations

- **Short:** < 5 years.
- **Medium:** 5 to 15 years.
- **Long:** > 15 years.

4.3.7 Administrative control

The model for managing configuration and updates:

- **Centrally-managed:** IT department or security operations centre controls all configuration and updates, security policies are centrally enforced, compliance monitoring is in place.
- **User-managed:** Individual end-users control configuration and updates. Varying security expertise, inconsistent update adoption.
- **Vendor-managed:** Manufacturer retains control over updates and configuration. Users cannot modify boot security settings.
- **Hybrid-managed:** Shared responsibility between multiple parties, needs clearly documented responsibility delineation.
- **Autonomous:** No administrative interface, device operates independently. Updates require physical access or hardware replacement.

4.3.8 Availability requirements

- **Flexible:** Downtime acceptable for updates/maintenance.
- **Standard:** Normal business availability expectations.
- **High:** High uptime required.
- **Critical:** Continuous operation required for safety or critical infrastructure.

4.3.9 Additional environmental characteristics

4.3.9.1 Environmental stress

Harsh environmental conditions affect boot manager reliability and security:

- **Temperature extremes:** Harsh thermal environments affecting non-volatile memory reliability and boot component integrity.
- **Vibration and shock:** Mechanical stress causing storage connection failures and data corruption risks.
- **Radiation:** Environments with ionising radiation causing storage bit flips requiring error correction and redundancy.
- **Humidity and corrosion:** Outdoor deployment, marine environments affecting hardware integrity.
- **Aging effects:** Flash wear (limited write cycles), bit rot, electromigration degradation over extended lifecycles.

These stresses increase probability of boot failures, requiring enhanced error detection, redundant storage, and robust recovery mechanisms. See REQ-BM-AP-006 and REQ-BM-AP-007.

4.3.9.2 Update frequency expectations

Boot manager update patterns vary by deployment:

- **Never:** ROM-based, OTP firmware, no post-manufacture updates possible.
- **Rare:** Major version updates only (e.g. every 2 to 5 years) for long-lifecycle industrial equipment.
- **Periodic:** Scheduled maintenance windows (quarterly, annually) for enterprise/infrastructure.
- **Responsive:** Updates deployed when vulnerabilities are discovered, following a testing-dependent timeline.
- **Continuous:** Rapid update capability for internet-connected consumer devices.

Update frequency affects vulnerability exposure window and influences defence-in-depth requirements and may also affect the lifetime of memory used. See clause B.2.2 (RF-SURFACE).

4.3.9.3 Interfaces

Boot managers interact through various technical interfaces:

- **Storage:** SPI flash, USB, eMMC, NVMe, SATA with different protection mechanisms.
- **Network:** Ethernet, WiFi supporting PXE, HTTP Boot, iSCSI protocols.
- **Debug:** JTAG, SWD, UART, USB, SPI programmers requiring access control.
- **User:** Serial console, display output, keyboard, remote management (IPMI/BMC).
- **Hardware:** Hardware security component connections (SPI/I2C or other protocols).
- **Runtime:** Firmware runtime services, ACPI tables, SMM/SMI interfaces.

Boot managers receive inputs such as hardware reset signals, configuration data, boot media, network packets, or user input, and produce outputs including system state, measurement logs, or error codes. Control interfaces include configuration utilities, update mechanisms, recovery modes, and remote management protocols.

The interface types listed above are comprehensive. The specific protocols and technologies listed under each type are representative examples; additional interfaces assignable to these types are subject to the same security considerations described in clause 5.

4.4 Distribution of security functions

4.4.1 Introduction

As a component within a larger system, boot managers both provide and depend upon security functions. This clause clarifies the security boundaries and responsibilities.

4.4.2 Security functions provided by the boot manager

Depending on capabilities, boot managers provide:

- Initial trust establishment before any other software executes.
- Boot chain integrity verification.
- Cryptographic measurement recording for attestation.
- Secure handoff to boot target.
- Protection of boot configuration and policy.
- Rollback protection for boot components.

- Root of trust for measurement.
- Root of trust for verification.
- Root of trust for storage.
- Root of trust for reporting.

NOTE: Verification, measurement, and rollback capabilities are implementation dependent.

4.4.3 Security functions required from the platform

The boot manager may depend on the following security functions from the hardware platform. These dependencies are documented as assumptions in clause B.3.2. Where a platform does not provide a listed function, the corresponding threats identified in clause B.3.2 are not mitigated by the present document.

- Hardware root of trust (immutable boot ROM or equivalent).
- Secure storage for cryptographic keys and configuration.
- Protected execution environment (where available).
- Hardware security components for key storage (where available).
- Physical security mechanisms (where available).

NOTE: Hardware security mechanisms may include various implementations such as TPM, HSM, secure elements, or write protection mechanisms that prevent tampering of boot manager code by the operating system or other software executing after boot handoff.

4.4.4 Security functions delegated to boot target

The following security functions are outside the scope of the boot manager and are the responsibility of the loaded boot target:

- Runtime security monitoring and threat detection.
- User authentication and authorisation (post-boot).
- Network security and firewall functions.
- Application security and sandboxing.
- Ongoing vulnerability management and patching.
- Security event logging (post-handoff).

NOTE: Some boot managers provide runtime services that remain accessible after boot target handoff (e.g. UEFI runtime services for variable storage, time, capsule updates or attestation). These runtime services are within scope when they affect boot security, integrity, or configuration.

4.4.5 Trust boundaries

Boot managers operate across multiple trust boundaries where different security domains interface.

Boot managers interface with multiple trust domains, each requiring different verification approaches, for example root of trust by hardware vendor, security hashes for different stages by firmware or board OEM, or certificates by operating system vendors.

- **Hardware:** Components trusted by boot manager without verification capability, including CPU microcode, platform security processors providing opaque services, immutable boot ROM and/or e-fuses.

- **Storage:** Components requiring verification before use, including all data read from storage media, configuration data, and boot components that may have been tampered with or corrupted.
- **Network:** Untrusted input requiring complete verification, including all network-sourced boot components, protocol data, and remote configuration.
- **Operating system:** Components to which the boot manager transfers control, including operating systems, hypervisors, and runtime services. This boundary is bidirectional when the operating system performs updates to the boot manager.

These boundaries define cases boot managers can enforce security properties or cases where they rely instead on trust assumptions or external security mechanisms.

4.5 Users

4.5.1 Introduction

Boot managers operate mostly without any user interaction during normal operation, with security decisions predetermined by configuration rather than runtime user input. In the requirements of clause 5, "authorised entity" is used in place of "user" to avoid ambiguity between the CRA definition of user and technical access roles. See clause 3.1.

4.5.2 User categories

Boot managers interact with different user types across their lifecycle, including:

- Manufacturers, during production and initial provisioning.
- System integrators, during deployment.
- System administrators, for enterprise configuration.
- End users, for boot selection.
- Service technicians, during maintenance.

4.5.3 User interaction patterns

- Manufacturing phase: Fuse programming, key provisioning, initial firmware installation, testing.
- Configuration phase: Use of setup and configuration tools.

NOTE: In some business models, operations such as fuse programming or key provisioning may be deferred from the manufacturing phase to the configuration phase, allowing OEMs to customise or finalise device configuration post-manufacture.

- Boot phase: Physical presence detection, boot target selection, recovery mode activation.
- Maintenance phase: Updates, rollbacks, resets.
- Decommissioning phase: Secure erasure, memory clean-up, key destruction, decommissioning verification.

These interactions represent points where boot manager security can be modified, compromised or bypassed, requiring appropriate access controls and authentication mechanisms.

Interactions requiring special attention:

- Device resale or transfer.
- Return for service/warranty.
- Employee termination (enterprise devices).

- End-of-life disposal.

These scenarios present opportunities for security policy changes, credential transfer, or secure erasure failures that could compromise subsequent users or violate data protection obligations.

User interactions requiring authentication are addressed in clause 5.4. Physical presence verification provides protection for security-critical configuration changes.

4.6 Use cases

4.6.1 Purpose

This clause describes use cases that characterise boot manager products by the type of security-enhancing capabilities provided.

A boot manager manufacturer declares the use case that corresponds to the capabilities of their product as placed on the market. Each use case is associated with a security profile (LOW, MEDIUM, or HIGH); each security profile determines which requirements in clause 5 apply.

Separately, boot manager products may implement functional capabilities (see clause 4.6.4) that are not associated with any specific use case but that, when present, trigger additional requirements in clause 5, serving to mitigate the risks introduced by those capabilities. The normative requirements for use case declaration and requirement application are specified in clause 5.1.

4.6.2 Capabilities as risk exposures

Most capabilities described in clause 4.2.3 do not inherently improve the security of the boot manager. They provide however functionality that may be required by the product's intended purpose. Each capability introduces additional attack surface and associated risks. A boot manager that does not implement a given capability is not subject to the risks introduced by the specific capability.

Where a boot manager implements a functional capability, the requirements in clause 5 that reference that capability apply. These requirements mitigate the risks introduced by the capability. Their applicability is conditional on the capability being present, not on the use case or security profile.

A limited set of capabilities genuinely enhance the security posture of the system. These capabilities are associated with the use cases defined in clause 4.6.3 because they establish the security foundation of the product. See Table 4.2.3.1-1 for the classification of capabilities.

4.6.3 Use case definitions

4.6.3.0 Use cases and security profiles

Three use cases are defined. Each use case is characterised by the security-enhancing capabilities the boot manager implements and is associated with a security profile. Use cases are cumulative: each higher use case includes all requirements of the use cases below it.

The rationale for the association between use cases and security profiles is described in the present document (Annex B).

Table 4.6.3.0-1: Use cases and security profiles

Use case	Security profile	Characterised by
UC-IMM	LOW	Immutable code + trust anchors
UC-VER	MEDIUM	Verified boot + updatable + logging + key provisioning
UC-HW	HIGH	UC-VER + hardware-assisted security

4.6.3.1 UC-IMM: Immutable

UC-IMM describes a boot manager where the executable code and trust anchors are stored in hardware-enforced immutable storage and cannot be modified after deployment.

The associated security profile: for UC-IMM is: LOW. All requirements for security profile LOW apply.

NOTE 1: The security posture of UC-IMM relies on immutability as the primary security property. The absence of update, configuration, and network boot capabilities eliminates entire classes of attack vectors.

NOTE 2: A UC-IMM boot manager cannot be remediated if a vulnerability is discovered post-deployment.

NOTE 3: UC-IMM addresses immutability of the boot manager only. Properties of the post-handoff stage are an environmental assumption (see clause B.3.4, AS-TARGET-2). Where this assumption does not hold, UC-IMM's security posture does not extend to the running system.

4.6.3.2 UC-VER: Verified and updateable

UC-VER describes a boot manager that implements verified boot, update capability, logging, and key provisioning as its security foundation.

The associated security profile: for UC-VER is: MEDIUM. All requirements for security profiles LOW and MEDIUM apply.

NOTE 1: The association of verified boot and update capability is deliberate: verified boot without update capability prevents remediation of compromised keys or signing infrastructure; update capability without verified boot provides no integrity assurance for updates.

NOTE 2: Key provisioning is a capability of the boot manager product. Whether the device manufacturer in which the boot manager is integrated exposes key management to the device owner is a decision for the device manufacturer. The boot manager provides the mechanism; it does not determine downstream key management policy.

4.6.3.3 UC-HW: Hardware-assisted security

UC-HW describes a boot manager that implements all UC-VER capabilities plus hardware-assisted security mechanisms.

The associated security profile: for UC-HW is: HIGH. All requirements for security profiles LOW, MEDIUM and HIGH apply.

NOTE: UC-HW does not imply resistance to advanced physical attacks (e.g. side-channel analysis, fault injection, chip decapsulation).

4.6.4 Functional capabilities and additional requirements

Functional capabilities (see Table 4.2.3.1-1) are not associated with any specific use case. They may be implemented based on the product's intended purpose. Where a functional capability is implemented, all requirements in clause 5 that reference that capability become applicable, regardless of the declared use case. These requirements mitigate the risks introduced by the capability.

Table 4.6.4-1: Capability requirements by use case

Capability	UC-IMM	UC-VER	UC-HW
Verified boot (see note 1)	Not expected	Required	Required
Update capability	Not permitted	Required	Required
Logging	Optional	Required	Required
Key provisioning	Not expected (see note 2)	Required	Required
Hardware security	Optional	Optional	Required
Network boot	Not permitted (see note 3)	Optional	Optional
Configuration management	Limited (see note 4)	Optional	Optional
Measured boot	Not expected	Optional	Optional
Recovery	Not expected	Optional	Optional
Authentication	Optional	Optional	Optional
Debug interface	Optional	Optional	Optional
NOTE 1: "Verified boot" covers both on-boot verification and on-update verification (capsule update) as defined in clause 4.2.1.2. Individual requirements in clause 5 may apply specifically to on-boot or on-update verification. Such scope is stated in the applicability statement of the requirement. NOTE 2: "Not expected" means the capability serves no purpose given the use case characterisation but is not prohibited. If nevertheless implemented, the corresponding requirements apply. NOTE 3: "Not permitted" means the capability is incompatible with the use case characterisation. A boot manager implementing that capability cannot declare that use case. NOTE 4: Under UC-IMM, configuration capability is limited to operational parameters that do not affect the security properties of the boot manager.			

4.6.5 Use case selection

Use case selection is based on the security-enhancing capabilities the boot manager implements as placed on the market, as follows:

a) UC-IMM, if the boot manager code, configuration, and trust anchors are immutable.

b) UC-VER, if the boot manager implements verified boot, update capability, logging, and key provisioning: UC-VER.

c) IC-HW, if the boot manager additionally uses hardware-backed security mechanisms: UC-HW.

Where a boot manager implements some but not all capabilities of a higher use case, the manufacturer declares the highest use case for which all required capabilities are met.

NOTE: Use case selection is based on the capabilities of the boot manager product. The deployment context of the device in which the boot manager is integrated is the responsibility of the device manufacturer's product conformity assessment, not of the boot manager manufacturer.

5 Technical requirements for products

5.1 Introduction: Applicability of the requirements

5.1.1 General

The technical requirements of the present document apply under the product context described in clause 4, which shall be in accordance with its intended use. The product shall comply with all applicable technical requirements of the present document at all times when operating in such a product context.

This clause establishes cybersecurity requirements implementing the essential cybersecurity requirements of Annex I, Part I of Regulation (EU) 2024/2847 [i.1].

The risk analysis underlying the requirements in clause 5 is documented in Annex B and capability-based conditions defined in clause 4.2.3. Annex B identifies assets requiring protection (clause B.1), risk factors (clause B.2), assumptions (clause B.3), threats (clause B.4), and maps risk factors to use cases (clause B.5) and security profiles (clause B.6).

Not all requirements are universally applicable: The applicability of requirements may be based on use cases or specific capabilities of the product. Requirements apply based on:

a) the security profile associated with the declared use case (clause 4.8); and

b) implemented functional capabilities: requirements that reference a functional capability apply when that capability is present, regardless of the declared use case.

Both conditions are evaluated independently. A requirement applies when the security profile condition and the capability condition (if any) are both met.

5.1.2 Use case declaration

5.1.2.0 Introduction

The manufacturer shall declare the use case (UC-IMM, UC-VER, or UC-HW) applicable to the boot manager product. The declared use case shall be consistent with the capability requirements defined in Table 4.6.4-1.

5.1.2.1 [UC-IMM]

Where the boot manager is declared as UC-IMM, the boot manager shall meet all of the following conditions:

a) the boot manager executable code and trust anchors reside in hardware-enforced immutable storage;

b) the boot manager does not implement update capability;

c) the boot manager does not implement network boot.

5.1.2.2 [UC-VER]

Where the boot manager is declared as UC-VER, the boot manager shall implement:

a) verified boot as defined in clause 4.2.1.2;

b) update capability as defined in clause 4.2.3.2;

c) logging capability as defined in clause 4.2.3.7;

d) key provisioning as defined in clause 4.2.3.10.

5.1.2.3 [UC-HW]

Where the boot manager is declared as UC-HW, the boot manager shall, in addition to meeting [UC-VER], implement hardware-backed security mechanisms for key storage and cryptographic operations that protect the root of trust against remote software attacks.

5.1.3 Not applicable verdict

A requirement verdict of NOT APPLICABLE shall be justified only when all of the following conditions are met:

a) the capability referenced in the requirement's applicability statement is not implemented by the product;

b) the declared use case shown in Table 4.6.4-1 does not assume that capability;

c) residual risks arising from the absence of that capability are documented in the product technical documentation.

5.2 Appropriate level of cybersecurity

5.2.0 Introduction appropriate level of cybersecurity requirements

This clause addresses the requirements in the CRA [i.1] Annex I, Part I, point (1). The present clause specifies the cybersecurity risk assessment recorded with the boot manager product and the relationship between the risk assessment and the requirements in clauses 5.3 to 5.15.

NOTE: Guidance on the generic security framework underlying "appropriate level of cybersecurity" is given in prEN 40000-1-2 [i.5]. prEN 40000-1-2 is not a normative reference of the present document.

5.2.1 [REQ-BM-RRM-001]

Requirement: The boot manager shall be placed on the market with a cybersecurity risk assessment recorded in the product technical documentation. The risk assessment shall identify the boot manager product and version assessed, the methodology used, the scope of the assessment, the assumptions from clause B.3 applicable to the product, the identified risks, and the conclusions.

Applicability: Applies to all boot managers (all security profiles).

5.2.2 [REQ-BM-RRM-002]

Requirement: The cybersecurity risk assessment shall consider the threats catalogued in Annex B as applicable to the boot manager's declared use case (clause 4.8) and operational environment (clause 4.3). The identified risks shall be consistent with the declared use case.

Applicability: Applies to all boot managers (all security profiles).

5.2.3 [REQ-BM-RRM-003]

Requirement: Where the cybersecurity risk assessment identifies risks not addressed by the requirements in clauses 5.3 to 5.15 applicable to the declared use case, additional product-level measures addressing those risks shall be implemented and documented in the product technical documentation.

Applicability: Applies to all boot managers (all security profiles).

5.2.4 [REQ-BM-RRM-004]

Requirement: Where the boot manager relies on security functions provided by integrated hardware or software components (see clause 4.4.3), the product technical documentation shall identify the security needs the boot manager places on those components and shall record documentary or test evidence that the components meet those needs.

Applicability: Applies to boot managers that rely on security functions provided by integrated components.

NOTE: Examples of security functions provided by integrated components include hardware-backed key storage, cryptographic primitives provided by a secure element, secure boot anchors provided by hardware ROM, and attack-resistance properties provided by a secure microcontroller or tamper-resistant security processor.

5.2.5 [REQ-BM-RRM-005]

Requirement: The cybersecurity risk assessment, the mapping of identified risks to the requirements applicable under clauses 5.3 to 5.15, any additional measures implemented under REQ-BM-RRM-003, and the supporting evidence for components under REQ-BM-RRM-004, shall be recorded in the product technical documentation.

Applicability: Applies to all boot managers (all security profiles).

5.3 No known exploitable vulnerabilities

5.3.0 Introduction to no known exploitable vulnerabilities requirements

This clause addresses the requirements in the CRA [i.1] Annex I, Part I, point (2)(a). Boot managers shall be placed on the market without known exploitable vulnerabilities. Due to the pre-OS execution environment, boot manager vulnerability management relies on verified update mechanisms rather than runtime scanning or automatic updates triggered on first use.

5.3.1 [REQ-BM-KEV-001]

Requirement: The boot manager shall not be placed on the market with known exploitable vulnerabilities in its code, configuration, or cryptographic mechanisms.

Applicability: Applies to all boot managers (all security profiles).

5.3.2 [REQ-BM-KEV-002]

Requirement: Where the boot manager implements update capability, security updates addressing known exploitable vulnerabilities shall be available for the duration of the support period.

Applicability: Applies to boot managers with update capability (MEDIUM or HIGH security profile).

5.3.3 [REQ-BM-KEV-003]

Requirement: Where the boot manager does not implement update capability, the product technical documentation shall state the absence of update capability as a residual risk, the date of the last vulnerability assessment performed prior to placing the product on the market, and the defence-in-depth measures implemented to mitigate the inability to remediate post-deployment vulnerabilities.

Applicability: Applies to boot managers without update capability (LOW security profile).

NOTE: For a boot manager without update capability, vulnerabilities disclosed between the date of the pre-market vulnerability assessment and the date of placing on the market cannot be remediated through post-deployment updates. The currency of the assessment is therefore a factor in the residual risk that the defence-in-depth measures required by this clause are intended to address.

5.4 Secure by default configuration

5.4.0 Introduction to secure by default configuration requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(b). Boot managers shall be put on the market with security features enabled, no default credentials, and protection matching available hardware and software resources.

NOTE: The technical implementation of security features (i.e. digital signature validation, or authenticated hash verification, logging capabilities, rollback protection) and the responsibility for enabling them by default may vary depending on the deployment model, device capabilities, and security policy. In many cases, the OEM or system integrator is responsible for activating these features during device configuration, rather than the silicon supplier, that is in charge to provide those features.

5.4.1 [REQ-BM-SBD-001]

Requirement: The boot manager shall enable cryptographic validation by default.

Applicability: Applies to boot managers with verified boot (MEDIUM or HIGH security profile).

NOTE: See clause 5.4.0 for details on implementation and activation details responsibilities.

5.4.2 [REQ-BM-SBD-002]

Requirement: The boot manager shall not implement unauthorised or undocumented mechanisms that allow bypassing authentication or security controls.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of such mechanisms include:

a) default or hardcoded passwords;

b) maintenance or service backdoors;

c) undocumented access interfaces or commands.

5.4.3 [REQ-BM-SBD-003]

Requirement: Where the boot manager supports password authentication to protect interfaces that can compromise core security guarantees of the system, the boot manager shall support the establishment of such passwords during initial deployment.

Applicability: Applies to boot managers with configuration capability using password authentication (all security profiles).

NOTE: The boot manager provides the mechanism. Whether a password is set at initial deployment depends on the integrator's configuration of the product.

5.4.4 [REQ-BM-SBD-004]

Requirement: The boot manager shall prevent automatic fallback to a boot mode that disables or weakens verification or rollback protection without explicit authorisation.

Applicability: Applies to boot managers with verified or measured boot (MEDIUM or HIGH security profile).

5.4.5 [REQ-BM-SBD-005]

Requirement: The boot manager shall provide a prominent indication appropriate for the device's interface capabilities (e.g. user-visible message, audio notification, LED signal, or logged event) when a security-relevant protection is reduced or disabled.

Applicability: Applies to boot managers with configuration capability (all profiles).

5.4.6 [REQ-BM-SBD-006]

Requirement: The boot manager shall support restoring the system to secure default configuration settings.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE 1: Where configuration or keys are immutable (e.g. ROM/fuses), restoration may be limited. Such limitations should be documented.

NOTE 2: This requirement applies to boot manager configuration settings. Firmware versioning is governed separately by the anti-rollback requirements in clause 5.8 (REQ-BM-INT-016, REQ-BM-INT-017); device-level reset is outside the scope of this requirement.

5.5 Secure updates

5.5.0 Introduction to secure updates requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(c). Where update capability is implemented, boot managers authenticate update sources, protect update-related keys, and isolate update logic from the normal boot path.

A boot manager assures the integrity and authenticity of an update in one of a small number of ways. The requirements in this clause apply across all of them. The following implementation scenarios, referred to in the present document as scenarios A to E, characterise these approaches:

- Scenario A: signature verification on the product: the boot manager verifies a digital signature over the update package, or over signed metadata covering it, before installation. This is the primary approach for boot managers able to perform public-key verification against a protected trust anchor.
- Scenario B: integrity-only verification: the boot manager verifies only an integrity value such as a hash or checksum and does not itself enforce source authenticity. This approach is confined to constrained boot managers that cannot perform signature verification, and relies on compensating controls such as a restricted update interface or an authenticated delivery channel.
- Scenario C: authenticated delivery channel: integrity and source authenticity are assured by an authenticated channel between the boot manager and an authorised update source. This approach applies where updates are delivered over a network and the boot manager does not perform package signature verification itself.
- Scenario D: restricted local or offline update: updates are applied through controlled local mechanisms, for example by authorised personnel using trusted tooling and controlled media. This is a primary approach where network update is not available or not permitted.
- Scenario E: platform-managed verification: signature verification and installation control are performed by a trusted underlying platform component rather than by the boot manager. This applies in the narrow case where the boot manager is updated through such a platform. A boot manager may combine more than one scenario, for example signed packages delivered over an authenticated channel.

NOTE: Protection against version downgrade and unauthorised rollback of updates is addressed by the anti-rollback requirements in clause 5.8 (REQ-BM-INT-016, REQ-BM-INT-017).

5.5.1 [REQ-BM-SU-001]

Requirement: Where a user interface is available, the boot manager shall provide status indicators for available updates or rely on authorised components to present such indicators.

Applicability: Applies to boot managers with update and configuration capabilities (MEDIUM or HIGH security profile).

NOTE: Authorised components may include the operating system, firmware management services, or other trusted update-management mechanisms.

5.5.2 [REQ-BM-SU-002]

Requirement: When network capability is available, the boot manager shall support mechanisms to check for the availability of updates or rely on authorised components to do so.

Applicability: Applies to boot managers with update capability (MEDIUM or HIGH security profile).

NOTE: Authorised components may include the operating system, firmware management services, or other trusted update-management mechanisms.

5.5.3 [REQ-BM-SU-003]

Requirement: The boot manager shall enable security policy updates through configuration, allow disabling features to address vulnerabilities, and maintain configuration update capability throughout the support period.

Applicability: Applies to boot managers with configuration capability (all profiles).

5.5.4 [REQ-BM-SU-004]

Requirement: The boot manager shall accept updates only from authenticated and authorised sources.

Applicability: Applies to boot managers with update capability where the boot manager performs on-update verification (MEDIUM or HIGH security profile).

5.5.5 [REQ-BM-SU-005]

Requirement: When hardware security components are available, the boot manager shall store update-verification keys using those components.

Applicability: Applies to boot managers with update capability where the boot manager performs on-update verification (MEDIUM or HIGH security profile).

5.5.6 [REQ-BM-SU-006]

Requirement: The boot manager shall isolate update logic from normal boot path.

Applicability: Applies to boot managers with update capability where the boot manager performs on-update verification (MEDIUM or HIGH security profile).

5.5.7 [REQ-BM-SU-007]

Requirement: The boot manager shall apply updates atomically, such that an interrupted or failed update does not leave the boot manager in a partially updated state. If an update cannot be completed successfully, the boot manager shall revert to the previously operational state and remain bootable.

Applicability: Applies to boot managers that apply updates themselves (MEDIUM or HIGH security profile).

NOTE: The previously operational state is the last boot manager image and configuration known to boot successfully. The mechanism for retaining it (for example a redundant image or a recovery image) is an implementation choice.

5.6 Authentication and access control

5.6.0 Introduction to authentication and access control requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(d). Boot managers control code execution authorisation and configuration protection through cryptographic verification, physical presence detection, and hardware-based protections.

5.6.1 [REQ-BM-AC-001]

Requirement: The boot manager shall require that any configuration changes to trusted keys, certificates, or trust anchor databases are authorised only through one of the following mechanisms:

a) direct physical presence at the device; or

b) submission of a signed artefact whose authenticity and integrity are cryptographically verified by the boot manager prior to application.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE: For boot managers deployed in critical systems or environments with elevated security requirements, manufacturers may consider requiring a combination of independent authentication mechanisms (e.g. physical presence combined with cryptographic credential verification) for trust anchor modifications. This defence-in-depth approach mitigates risks from compromise of a single authentication factor.

5.6.2 [REQ-BM-AC-002]

Requirement: The boot manager shall allow protection of configuration settings related to the boot process against unauthorised modification if configured.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE 1: In embedded system scenarios, replacing or revoking trusted keys or certificates is often performed via authenticated software updates, without requiring physical presence.

NOTE 2: Examples of configuration settings related to the boot process include:

a) boot order;

b) boot parameters;

c) selection of boot targets or modes.

5.6.3 [REQ-BM-AC-003]

Requirement: The boot manager shall require explicit authentication before any modification of security-critical configuration settings.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE: Examples of such actions include:

a) weakening or disabling secure default settings;

b) downgrading security-related configuration parameters;

c) modifying security-critical options such as verification policies or trust anchors.

5.6.4 [REQ-BM-AC-004]

Requirement: When passwords are used for authentication, the boot manager shall throttle authentication attempts, for instance, by limiting attempts, increasing delays, or using cryptographic methods designed to be slow key derivation functions that are deliberately slow.

Applicability: Applies to boot managers with configuration capability using password authentication (all security profiles).

5.6.5 [REQ-BM-AC-005]

Requirement: Where security configuration policies are provided as separate signed artefacts, the boot manager shall verify their authenticity and integrity before importing or enforcing them.

Applicability: Applies to boot managers with verified or measured boot and configuration capability (MEDIUM or HIGH security profile).

5.6.6 [REQ-BM-AC-006]

Requirement: The boot manager shall indicate when running with non-default security-critical configuration through persistent visual indication or, where logging capability is supported, through logged events.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE 1: The indication may be visual (when a user interface is available) or may consist of recorded events in secure logs.

NOTE 2: Examples of indications include: a) a persistent on-screen message or icon; b) a status LED or display element; c) a logged event recorded in a secure audit log.

5.6.7 [REQ-BM-AC-007]

Requirement: The boot manager shall protect trusted certificate stores from unauthorised modification.

Applicability: Applies to boot managers with verified or measured boot (MEDIUM or HIGH security profile).

5.6.8 [REQ-BM-AC-008]

Requirement: The boot manager shall be accompanied with documentation declaring its code execution authority model, including the mechanisms and entities that constitute it, their scope and relationships, the default state as placed on the market, any external dependencies of the mechanisms, and the processes by which the model can be changed where any aspect is mutable.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Where no entity or mechanism has authority to decide which code can be executed, this is itself a declaration of the model and satisfies the requirement provided the documentation states it explicitly.

5.6.9 [REQ-BM-AC-009]

Requirement: For each authority transfer mechanism provided by the boot manager, the boot manager shall be accompanied with documentation declaring the entities that can initiate or approve the transfer, the components covered, and the permanence of the transfer.

Applicability: Applies to boot managers that provide one or more authority transfer mechanisms.

5.6.10 [REQ-BM-AC-010]

Requirement: The boot manager shall be accompanied with documentation declaring how the integrator or a downstream party can verify whether a given piece of software is authorised to run on the boot manager, including the tools, public keys, or mechanisms required for that verification.

Applicability: Applies to all boot managers (all security profiles).

5.6.11 [REQ-BM-AC-011]

Requirement: Any authority transfer initiated through a mechanism provided by the boot manager shall require explicit authorisation by the entity declared in the authority model as competent to authorise the transfer and shall not be performed as part of an automatic update or as a condition for continued operation.

Applicability: Applies to boot managers that provide one or more authority transfer mechanisms.

5.7 Confidentiality protection

5.7.0 Introduction to confidentiality protection requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(e). Boot managers shall protect cryptographic key material, credentials, and security policies in resource-constrained environments against persistent physical access threats.

5.7.1 [REQ-BM-CP-001]

Requirement: The boot manager shall restrict access to cryptographic key material to authorised boot components as defined by the boot manager's security policy.

Applicability: Applies to boot managers using cryptographic keys (all security profiles).

5.7.2 [REQ-BM-CP-002]

Requirement: Device-specific keys shall be used for symmetric verified boot operations. The use of global secret keys for verified boot purposes is explicitly forbidden.

Applicability: Applies to boot managers with verified boot (MEDIUM or HIGH security profile).

NOTE: Use of symmetric device-unique keys mitigates the risk of large-scale attacks resulting from compromise of a single global key.

5.7.3 [REQ-BM-CP-003]

Requirement: The boot manager shall prevent key material from being used for cryptographic purposes other than those for which it was designated.

Applicability: Applies to boot managers with verified or measured boot (MEDIUM or HIGH security profile).

5.7.4 [REQ-BM-CP-004]

Requirement: The boot manager shall not place sensitive data in crash dumps or log.

Applicability: Applies to boot managers with logging capability (MEDIUM or HIGH security profile).

NOTE: Examples of sensitive data include:

a) cryptographic key material;

b) authentication credentials;

c) security-critical configuration parameters;

d) personal or otherwise confidential information managed during the boot process.

5.7.5 [REQ-BM-CP-005]

Requirement: The boot manager shall protect stored credentials using approved mechanisms that prevent recovery of the credential values. This does not include tokenised values or credentials.

Applicability: Applies to boot managers with configuration and recovery capability (all security profiles).

NOTE 1: Examples of protected credential types include user passwords, recovery keys, and other authentication secrets.

NOTE 2: Examples of protection mechanisms include:

a) key-derivation functions (e.g. salted password hashing);

b) hardware-bound storage or sealed storage;

c) encryption using keys not accessible to unauthorised entities;

d) key-wrapping or secret-derivation schemes.

5.7.6 [REQ-BM-CP-006]

Requirement: The boot manager shall protect network boot parameters containing authentication information using approved mechanisms that prevent unauthorised disclosure.

Applicability: Applies to boot managers with configuration capability and network boot (MEDIUM or HIGH security profile).

5.7.7 [REQ-BM-CP-007]

Requirement: The boot manager shall cryptographically erase all data possessed by boot manager and clear all security-critical configuration during secure disposal, where technically feasible (see note).

Applicability: Applies to boot managers with verified or measured boot and configuration capability that implement secure disposal (MEDIUM or HIGH security profile).

NOTE: When cryptographic material or sensitive data are stored in immutable components such as e-fuses or ROM, complete erasure may not be possible.

5.7.8 [REQ-BM-CP-008]

Requirement: When secure disposal is supported, the boot manager shall provide a secure response upon successful completion of the sanitisation process.

Applicability: Applies to boot managers that implement secure disposal (all security profiles).

5.7.9 [REQ-BM-CP-009]

Requirement: The boot manager shall support protecting the confidentiality and integrity of boot configuration data transmitted over the network using secure methods or secure alternatives.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

5.7.10 [REQ-BM-CP-010]

Requirement: The boot manager shall clear confidential or unprotected sensitive data after use, clear from memory credentials and temporary data structures containing such before boot target handoff and not persist authentication credentials or confidential cryptographic material beyond the operating system handoff. The measurements and credentials used for attestation shall persist.

Applicability: Applies to all boot managers (all security profiles).

NOTE 1: Examples of unprotected sensitive data include authentication credentials, unsalted passwords, cryptographic key material, and temporary data structures created during the boot process.

NOTE 2: Examples of mechanisms for removing sensitive data include: a) overwriting sensitive data after use; b) clearing temporary data structures and credentials before handing off control to the boot target; c) avoiding persistence of authentication credentials or cryptographic material beyond the operating system handoff.

5.8 Integrity protection

5.8.0 Introduction to integrity protection requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(f). Boot managers establish trust chains through cryptographic verification, measurement, and protection mechanisms that resist component substitution, rollback, and TOCTOU attacks.

5.8.1 [REQ-BM-INT-001]

Requirement: The boot manager shall enforce privilege boundaries so that code executed in one boot stage cannot obtain privileges assigned to another boot stage or cross established trust boundaries.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of privilege boundaries include:

a) restrictions on access to memory regions;

b) restrictions on execution of privileged instructions;

c) limitations on access to security-sensitive hardware resources.

5.8.2 [REQ-BM-INT-002]

Requirement: The boot manager shall verify integrity and authenticity of each boot stage using approved cryptographic mechanisms as defined in Annex K and shall establish a chain of trust to a root trust anchor before transferring control.

Applicability: Applies to boot managers with on-boot verification (MEDIUM or HIGH security profile).

5.8.3 [REQ-BM-INT-003]

Requirement: The boot manager shall prevent boot continuation with components that fail integrity or authenticity verification.

Applicability: Applies to boot managers with on-boot verification (MEDIUM or HIGH security profile).

NOTE 1: Examples of verification failures include:

a) invalid, expired, or revoked cryptographic credentials;

b) substituted or tampered components;

c) missing or untrusted verification data.

NOTE 2: Cryptographic verification mechanisms may include digital signatures, message authentication codes (MACs), or other approved methods providing equivalent assurance.

5.8.4 [REQ-BM-INT-004]

Requirement: The boot manager shall compute and record a cryptographic measurement of each boot stage into hardware-protected storage before transferring control, establishing a measurement chain rooted in a hardware-based trust anchor.

Applicability: Applies to boot managers with measured boot (all security profiles).

NOTE 1: Unlike verified boot, measured boot does not prevent execution of unverified components. Instead, it provides an attestable record that enables a remote or local verifier to determine whether the boot sequence matches expected values.

NOTE 2: Measurement mechanisms may include:

a) extending hash values into hardware-protected registers (e.g. TPM PCRs);

b) deriving compound device identifiers (e.g. DICE CDI);

c) recording measurements in other tamper-evident storage.

NOTE 3: The integrity of the measurement chain depends on the trustworthiness of the initial measurement by the hardware root of trust and the correct extension of measurements at each subsequent stage

5.8.5 [REQ-BM-INT-005]

Requirement: The boot manager shall verify authenticity and integrity of update packages before installation, and additionally after the receipt of content where updates are written to protected storage.

Applicability: Applies to boot managers with update capability where the boot manager performs on-update verification (MEDIUM or HIGH security profile).

5.8.6 [REQ-BM-INT-006]

Requirement: The boot manager shall implement software-based cryptographic verification with keys loaded from protected storage.

Applicability: Applies to boot managers with verified boot when hardware security components are unavailable (MEDIUM or HIGH security profile).

5.8.7 [REQ-BM-INT-007]

Requirement: The boot manager shall support configuration among a set of approved signing algorithms.

Applicability: Applies to boot managers with verified boot and with configuration capability, where the trust anchor is not immutable (MEDIUM or HIGH security profile).

NOTE: A set of length one is acceptable. The requirement establishes the capability to substitute one approved algorithm for another, e.g. when an algorithm becomes deprecated.

5.8.8 [REQ-BM-INT-008]

Requirement: The boot manager shall support combined verification using more than one approved signing algorithm.

Applicability: Applies to boot managers with verified boot and with configuration capability, where the trust anchor is not immutable (MEDIUM or HIGH security profile).

5.8.9 [REQ-BM-INT-009]

Requirement: The boot manager shall use a design that prevents TOCTOU attacks within its documented threat model.

Applicability: Applies to boot managers with verified boot capability (MEDIUM or HIGH security profile).

NOTE: Threats addressed by this requirement are those identified in the security analysis (Annex B) for the declared use case. Physical-attack TOCTOU on the boot medium (T-PHYS-5, T-PHYS-6) is addressed under Annex D and is not within the scope of this requirement.

5.8.10 [REQ-BM-INT-010]

Requirement: The boot manager shall load components into protected memory before verification, when supported by the underlying hardware platform.

Applicability: Applies to boot managers with verified boot (MEDIUM or HIGH security profile).

5.8.11 [REQ-BM-INT-011]

Requirement: The boot manager shall protect the integrity and authenticity of sensitive configuration settings.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE: Mechanisms that protect the integrity and authenticity of sensitive configuration settings may include hardware-protected storage, or authenticated encryption with freshness protection (e.g. nonces or monotonic counters) and non-deterministic encryption.

5.8.12 [REQ-BM-INT-012]

Requirement: The boot manager shall restore secure default configuration settings when configuration corruption is detected, where technically feasible.

Applicability: Applies to boot managers with configuration capability (all profiles).

NOTE: When cryptographic material or configuration data are stored in immutable components such as e-fuses or ROM, complete erasure or restoration may not be possible.

5.8.13 [REQ-BM-INT-013]

Requirement: The boot manager shall authenticate network boot servers using cryptographic certificates before accepting actions or configuration data from the network boot server.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

5.8.14 [REQ-BM-INT-014]

Requirement: The boot manager shall reject network boot connections with invalid, or revoked server certificates.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: Expiration checking of cryptographic credentials is conditional on a dependable source of time being available to the boot manager (see assumption in clause B.3.2). Where no dependable source of time is available, the boot manager cannot perform expiration checking; revocation checking remains applicable.

5.8.15 [REQ-BM-INT-015]

Requirement: The boot manager shall verify that network configuration and boot-related responses originate from authorised infrastructure.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: Examples of network configuration and boot-related responses include:

a) DHCP responses;

b) PXE boot offers;

c) network-provided boot configuration data.

5.8.16 [REQ-BM-INT-016]

Requirement: The boot manager shall enforce rollback protection when verifying boot stages and configuration data.

Applicability: Applies to boot managers with update or configuration capability (MEDIUM or HIGH security profile).

NOTE: In some business models, anti-rollback protection can be configured by the OEM. Refer to note in clause 5.3.

5.8.17 [REQ-BM-INT-017]

Requirement: The boot manager shall store anti-rollback counters in hardware-backed or tamper-evident storage and verify signed version metadata before accepting updates.

Applicability: Applies to boot managers with HIGH security profile and update capability.

5.8.18 [REQ-BM-INT-018]

Requirement: The boot manager shall implement anti-rollback mechanisms and verify signed version metadata before accepting updates.

Applicability: Applies to boot managers with verified boot and measured boot, where measured boot includes remote attestation (MEDIUM or HIGH security profile).

NOTE: Certificate-chain verification may include checking:

a) the full chain of trust from the end-entity certificate to the trusted root;

b) certificate validity periods;

c) revocation status through mechanisms such as CRLs or OCSP;

d) compliance with certificate policies or usage constraint.

5.8.19 [REQ-BM-INT-019]

Requirement: The boot manager shall support revocation of mutable compromised keys and certificates.

Applicability: Applies to boot managers with verified boot and measured boot, where measured boot includes remote attestation (MEDIUM or HIGH security profile).

NOTE 1: In boot managers without connectivity capability, revocation can either be managed manually (e.g. via dedicated tools or physical programming) or through authenticated software updates.

NOTE 2: In many embedded systems, the number of revocation slots (especially for verified boot) is limited by available storage or hardware design. For this reason the number of possible key revocations may be constrained.

NOTE 3: Revocation mechanisms may include:

a) a revocation database;

b) configuration or policy updates;

c) certificate revocation lists (CRLs);

d) online or offline status information (e.g. OCSP responses);

e) manufacturer- or owner-provided revocation metadata.

5.8.20 [REQ-BM-INT-020]

Requirement: The boot manager shall use cryptographic algorithms, key sizes, and parameters in accordance with Annex K.

Applicability: Applies to all boot managers (all security profiles).

5.8.21 [REQ-BM-INT-021]

Requirement: The boot manager shall support migration between approved cryptographic algorithms.

Applicability: Applies to boot managers with update capability, where the trust anchor is not immutable (MEDIUM or HIGH security profile).

NOTE: Migration may be achieved through configuration metadata, controlled algorithm selection mechanisms, or replacement of the boot manager itself, in alignment with the update capability declared in clause 4.2.3.2.

5.8.22 [REQ-BM-INT-022]

Requirement: The boot manager shall use collision- and preimage-resistant one-way functions for origin authentication and verification of boot code provenance.

Applicability: Applies to boot managers with verified or measured boot capability (MEDIUM or HIGH security profile).

NOTE 1: Examples of collision- and preimage-resistant one-way functions include SHA-2 and SHA-3.

NOTE 2: Non-collision-resistant hashes (e.g. GHASH in AES-GCM) may be used for integrity protection only when combined with device-specific keys.

5.8.23 [REQ-BM-INT-023]

Requirement: The boot manager shall support replacement of trust anchors used for boot verification.

Applicability: Applies to boot managers with update capability, where the trust anchor is not immutable (MEDIUM or HIGH security profile).

NOTE: Trust anchor replacement supports recovery from key compromise and aligns with the key provisioning capability declared in clause 4.2.3.10.

5.8.24 [REQ-BM-INT-024]

Requirement: The boot manager shall protect the storage in which trust anchors, key enrolment records, and policy database contents used by the code execution authority are held against modification by unauthorised entities.

Applicability: Applies to boot managers that implement one or more code execution authority mechanisms requiring stored authority data.

5.9 Data minimisation

5.9.0 Introduction to data minimisation requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(g). Boot managers shall process only data necessary for boot operations, security verification, and error indication, given memory and storage constraints.

5.9.1 [REQ-BM-DM-001]

Requirement: The boot manager shall minimise disclosure of information to network infrastructure by limiting transmitted data to what is necessary for boot operations.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: Examples of measures to minimise network disclosure include:

a) requesting only the files required for the network boot process;

b) negotiating only protocol parameters essential for establishing the boot session;

c) restricting identifiers exposed on the network (e.g. to MAC address and device class).

5.9.2 [REQ-BM-DM-002]

Requirement: The boot manager shall prevent disclosure of sensitive device information on the network and shall ensure that temporary network credentials are not retained beyond their required use.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: Examples of sensitive device information that should not be disclosed include:

a) hardware serial numbers;

b) internal configuration or diagnostic data.

5.10 Availability protection

5.10.0 Introduction to availability protection requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(h). Boot managers shall maintain availability during failures or attacks through recovery mechanisms, failsafe operations, and resistance to denial-of-service conditions. Where this clause specifies limits without numeric values, the manufacturer shall document specific thresholds in security documentation per Annex C.

5.10.1 [REQ-BM-AP-001]

Requirement: The boot manager shall support fallback to previous known-good configuration.

Applicability: Applies to boot managers with configuration capability (all profiles).

5.10.2 [REQ-BM-AP-002]

Requirement: The boot manager shall be able to automatically recover from any interruptions during the update process.

Applicability: Applies to boot managers with HIGH security profile.

5.10.3 [REQ-BM-AP-003]

Requirement: The boot manager shall enforce timeouts to prevent indefinite blocking during boot-related operations.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of operations that may require timeouts include:

a) parsing or validation of configuration data;

b) network discovery, negotiation, or file retrieval;

c) loading or verifying boot components.

5.10.4 [REQ-BM-AP-004]

Requirement: The boot manager shall limit retry attempts and resource consumption for signature verification and network operations.

Applicability: Applies to boot managers with verified boot or network boot (MEDIUM or HIGH security profile).

5.10.5 [REQ-BM-AP-005]

Requirement: The boot manager shall prevent unauthorised bypass of security verification steps during normal operation.

Applicability: Applies to all boot managers (all security profiles).

NOTE: In specific, controlled scenarios such as in-field debug mode, disabling security verification steps may be permitted for issue analysis, provided that such actions are strictly authenticated.

5.10.6 [REQ-BM-AP-006]

Requirement: The boot manager shall detect errors in critical data (e.g. cryptographic keys, configuration data) and mitigate the impact of detected errors. Where the underlying platform supports error correction for critical data, the boot manager shall additionally use those mechanisms to correct errors.

Applicability: Applies to boot managers with MEDIUM or HIGH security profile.

NOTE 1: Mitigation may include refusing to boot, falling back to a known-good configuration, or initiating a recovery path. The objective is to prevent execution in a corrupted state.

NOTE 2: Where critical data is stored in immutable elements such as e-fuses, error correction can be implemented using Forward Error Correction (FEC) codes or similar techniques.

5.10.7 [REQ-BM-AP-007]

Requirement: The boot manager shall implement a mechanism to maintain availability of essential boot code in the presence of partial storage corruption subject to having enough data available for recovering the correct content of the essential boot code or failure of stages following the initial boot code.

Applicability: Applies to boot managers with MEDIUM or HIGH security profile, where the boot manager is not stored in immutable memory.

NOTE 1: The mechanisms may include redundant storage with selection of uncorrupted copies, fallback boot partitions, or integrity-driven repair where sufficient information is available. The implementation should be documented in the product description.

NOTE 2: Essential boot code includes components required to initialise the platform and load subsequent boot stages. The initial boot code (e.g. reset vector and immediate successor instructions) is excluded from this requirement, as recovery from corruption at that stage requires hardware mechanisms outside the boot manager's control.

5.10.8 [REQ-BM-AP-008]

Requirement: The boot manager shall require authentication or physical presence before performing sensitive actions in recovery mode.

Applicability: Applies to boot managers with recovery capability (all security profiles).

NOTE 1: Recovery mode may be triggered automatically upon boot failures; however, any critical or security-relevant operations performed during recovery mode require authentication or physical presence.

NOTE 2: The boot manager may implement authentication, physical presence, or both as the authorisation mechanism for sensitive actions in recovery mode. Where physical presence is not a feature of the product (for example headless infrastructure equipment or communications devices managed exclusively through network interfaces), authentication is the authorisation mechanism.

5.10.9 [REQ-BM-AP-009]

Requirement: When network boot is supported, the boot manager shall handle network boot failures in a manner that maintains boot availability.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: The mechanisms for handling network boot failures may include, for example:

a) enforcing timeouts;

b) falling back to a local boot mechanism;

c) performing exponential backoff retries;

d) attempting multiple authorised network boot servers.

5.10.10 [REQ-BM-AP-010]

Requirement: The boot manager shall generate logs for security-relevant failure conditions.

Applicability: Applies to boot managers with verified boot or measured boot and logging capability (MEDIUM or HIGH security profile).

NOTE: Security-relevant failure conditions may include verification failures, integrity violations, authentication failures, or unexpected modification of boot components

5.10.11 [REQ-BM-AP-011]

Requirement: The boot manager shall use time representations that remain valid beyond 2038.

Applicability: Applies to all boot managers (all security profiles).

NOTE 1: This requirement applies to all security-relevant uses of time, including:

a) certificate validity periods;

b) internal system time used for boot decisions;

c) timestamps recorded in logs or audit records.

NOTE 2: This requirement helps prevent failures associated with 32-bit time representations (commonly known as the Year-2038 problem).

5.10.12 [REQ-BM-AP-012]

Requirement: The boot manager shall protect recovery mechanisms from unauthorised modification or disablement.

Applicability: Applies to boot managers with recovery capability (all security profiles).

5.10.13 [REQ-BM-AP-013]

Requirement: The boot manager shall prevent partial execution of boot components when verification has not completed successfully.

Applicability: Applies to boot managers with verified boot (MEDIUM or HIGH security profile).

5.10.14 [REQ-BM-AP-014]

Requirement: The boot manager shall not perform security-sensitive operations when required cryptographic components or services are unavailable.

Applicability: Applies to all boot managers (all security profiles).

5.10.15 [REQ-BM-AP-015]

Requirement: The boot manager shall not bypass security controls when handling security violations or verification failures.

Applicability: Applies to boot managers with recovery capability (all security profiles).

5.10.16 [REQ-BM-AP-016]

Requirement: The boot manager shall preserve recovery capability across firmware or configuration updates.

Applicability: Applies to boot managers with update and recovery capability (MEDIUM or HIGH security profile).

5.11 Non-interference

5.11.0 Introduction to impact minimisation requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(i). Boot managers shall limit resource consumption, release hardware properly, and prevent negative impact on services provided by other devices or networks.

5.11.1 [REQ-BM-IM-001]

Requirement: The boot manager shall avoid generating harmful or disruptive network behaviour.

Applicability: Applies to boot managers with network boot (MEDIUM or HIGH security profile).

NOTE: Examples of harmful or disruptive network behaviour include:

a) broadcast storms;

b) network loops.

5.12 Attack surface minimisation

5.12.0 Introduction to attack surfaces minimisation requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(j). Boot managers shall minimise the attack surface by limiting functionality, eliminating debug interfaces, and restricting all non-essential capabilities in production builds.

5.12.1 [REQ-BM-AMS-001]

Requirement: The boot manager shall, before handing off control to the boot target, ensure that only the resources necessary for the boot target remain enabled.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of resources that may need to be disabled or released include, but are not limited to:

a) debug or diagnostic interfaces;

b) DMA-capable devices or peripherals;

c) hardware resources not required by the boot target;

d) allocated volatile memory;

e) network or communication buffers; and

f) memory regions that are not required for the operation of the boot target.

5.12.2 [REQ-BM-AMS-002]

Requirement: The boot manager shall exclude non-essential code from production builds.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of non-essential code falling under this requirement may include a) test code; b) debug instrumentation; c) coverage tools; d) experimental features; e) unsupported platform code.

5.12.3 [REQ-BM-AMS-003]

Requirement: The boot manager shall disable or protect interfaces and functions that are not intended for operational use.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of interfaces and functions that should be disabled in production configuration include:

a) debug interfaces;

b) diagnostic consoles;

c) test or manufacturing modes;

d) verbose or developer-oriented logging.

5.12.4 [REQ-BM-AMS-004]

Requirement: The boot manager shall remove unused or non-essential interfaces and protocols to reduce the attack surface.

Applicability: Applies to all boot managers (all security profiles).

NOTE: Examples of interfaces and protocols that should be disabled include:

a) unused hardware interfaces;

b) non-essential network services or discovery mechanisms.

5.12.5 [REQ-BM-AMS-005]

Requirement: The boot manager shall validate all inputs to ensure they match the expected formats.

Applicability: Applies to all boot managers (all security profiles).

NOTE 1: Input validation mechanisms may include:

a) bounds checking;

b) enforcement of size limits;

c) rejection of malformed, truncated, or oversized data.

NOTE 2: Inputs in this context may include:

a) boot images;

b) certificates;

c) configuration;

d) cryptographic data before processing.

5.12.6 [REQ-BM-AMS-006]

Requirement: The boot manager shall detect errors in critical operations and reject non-compliant inputs; transition to error handling on faults.

Applicability: Applies to all boot managers (all security profiles).

5.12.7 [REQ-BM-AMS-007]

Requirement: The boot manager shall provide a mechanism to disable configuration options that are not required by the deployment integrator.

Applicability: Applies to boot managers with configuration capability (all profiles).

5.13 Exploit mitigation

5.13.0 Introduction to exploit mitigation requirements

This clause addresses the essential requirement in CRA [i.1], Annex I, Part I, point (2)(k) on mitigating the impact of an incident. The mechanisms available to boot managers are constrained by the pre-OS execution environment.

5.13.1 [REQ-BM-EM-001]

Requirement: The boot manager shall implement exploitation mitigation mechanisms appropriate to the documented threat model to prevent unauthorised code execution, privilege escalation between boot stages, and uncontrolled propagation of compromise.

Applicability: Applies to boot managers where the documented threat model includes runtime code-execution attacks or stage-to-stage privilege escalation (typically MEDIUM or HIGH security profile).

NOTE 1: Annex D provides informative guidance on exploitation mitigation techniques. The techniques listed are non-exhaustive; selection is appropriate to the toolchain, platform, and threat model. This requirement does not mandate a specific subset.

NOTE 2: Mitigations addressing hardware-side physical attacks (T-PHYS-5, T-PHYS-6, T-PHYS-9, T-PHYS-11) are addressed under Annex D component-standard deferrals and are not within the scope of this requirement.

5.14 Monitoring

5.14.0 Introduction to monitoring requirements

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(l). Boot managers provide visibility into boot processes through logs and attestation within pre-OS constraints of limited storage and no file systems.

5.14.1 [REQ-BM-MON-001]

Requirement: The boot manager shall record measurements of boot components and security-critical configuration in a tamper-evident way (e.g. tamper-proof storage, or extended into a platform configuration register) before handoff.

Applicability: Applies to boot managers with measured boot and logging capability (MEDIUM or HIGH security profile).

NOTE: Tamper-evident retention does not require persistent storage. A platform configuration register or equivalent volatile mechanism provides the required tamper-evident property. Write-endurance constraints apply where persistent storage is used.

5.14.2 [REQ-BM-MON-002]

Requirement: The boot manager shall provide measurement records in a format that supports remote attestation mechanisms to ensure freshness against replay.

Applicability: Applies to boot managers with measured boot and logging capability (MEDIUM or HIGH security profile).

NOTE 1: Freshness mechanisms may include nonces, monotonic counters, timestamps, or challenge-response protocols.

NOTE 2: Measurement records may include information such as:

a) integrity digests of boot components;

b) indices or identifiers of measured registers or structures;

c) event types or measurement categories;

d) version or revision information of components.

5.14.3 [REQ-BM-MON-003]

Requirement: The boot manager shall indicate security-relevant failures and state changes.

Applicability: Applies to boot managers with logging capability (MEDIUM or HIGH security profile).

NOTE: Examples of security-relevant failures and state changes includes:

a) verification failures;

b) authentication failures;

c) security policy violations;

d) recovery mode activation; and

e) execution of unsigned code.

5.14.4 [REQ-BM-MON-004]

Requirement: The boot manager shall provide version information accessible to operating systems, management systems or authorised entity.

Applicability: Applies to all boot managers (all security profiles).

5.14.5 [REQ-BM-MON-005]

Requirement: Where the boot manager implements logging capability, the boot manager shall record each authority transfer as a security-relevant event in the security log, including the mechanism invoked, the entity that initiated the transfer, the entity that approved the transfer, and the components affected.

Applicability: Applies to boot managers that implement logging capability and provide one or more authority transfer mechanisms.

5.15 Factory reset and data portability

5.15.0 Introduction

This clause addresses the requirements in the CRA [i.1], Annex I, Part I, point (2)(m).

The present clause specifies obligations for the restoration of the boot manager to its default state as placed on the market, and for the secure transfer of data and settings to other products or systems where the boot manager provides a transfer mechanism.

Transparency toward the user on data processed by the boot manager is not a boot manager product characteristic. Boot managers do not process user personal data in the course of normal boot operations.

NOTE: Where end-of-life data removal procedures require user action, product documentation describes the data categories processed by the boot manager and the steps to remove them.

5.15.1 [REQ-BM-FRDP-001]

Requirement: The boot manager shall provide a mechanism, invocable by an authorised entity, by which all user-modifiable data and settings stored by the boot manager are securely and permanently removed, and the boot manager is restored to its default state as placed on the market.

Applicability: Applies to boot managers that store user-modifiable data or settings (typically MEDIUM or HIGH security profile).

NOTE: User-modifiable data and settings include boot configuration, user-added trust anchors, audit logs, and any other persistent state not part of the boot manager's default state as placed on the market. The default state itself is not removed by this mechanism.

6 Assessment criteria for compliance with technical requirement

6.1 Introduction to the assessment and compliance criteria

6.1.0 Assessment criteria general structure

This clause provides objective and reproducible assessment criteria to determine whether a product complies with the technical security requirements of clause 5. The assessment criteria are without prejudice to the conformity assessment procedures defined in Annex VIII of Regulation (EU) 2024/2847 [i.1]; they apply irrespective of the procedure chosen by the manufacturer.

For each cybersecurity requirement defined in clause 5, the following clauses specify assessment criteria to determine whether the requirement is met. The mapping is one-to-one: ACC-BM-ESR-NNN assesses REQ-BM-ESR-NNN.

The assessment criteria for each requirement are structured as follows:

- **Assessment reference**: Refers to the identifier of the concerned technical requirement.
- **Assessment objective:** Defines the security property or capability that shall be verified, ensuring that the assessment remains focused on the intent of the requirement.
- **Assessment preparation:** Describes the environment, setup, and preconditions required before executing the assessment with a view to guaranteeing consistent evaluation results. It includes the following elements as applicable:
  - Test environment: Describe the hardware, software, and network setup used for the assessment, including versions, topology, and any relevant dependencies.
  - Preconditions: Specify any configurations, credentials, or operational states that should be established before the test (e.g. product initialized, certificates loaded, user roles created).
  - Required tools: Identify the tools or software necessary to perform the assessment (e.g. vulnerability scanners, protocol fuzzers, traffic analysers, static code analysers, cryptographic test suites).
  - Required information/documentation for the assessment: Specify all information that is necessary to perform the assessment.
  - Reference any vendor-provided setup guides, configuration instructions, or operational manuals, as well as any relevant standards or technical notes, that define how the product shall be configured or operated for the assessment.

- **Assessment activities:** Provides execution steps to be performed such that the same assessment verdict would be reached when repeating these steps now or at a later point in time. Assessment activities may include, as applicable:
  - Review information/documentation for the assessment to confirm that the described implementation matches the requirement (e.g. verify that the security architecture document specifies TLS 1.2 or higher for all external interfaces, or that the password policy aligns with the defined threshold).
  - Perform security functional tests to verify the completeness and correctness of the information/documentation for the assessment.
  - Perform security functional or penetration tests to verify that implemented controls are correctly implemented e.g. to prevent unauthorised access or data modification (e.g. via attempting to log in with invalid credentials to test lockout enforcement or trying to modify protected configuration files without administrative privileges).
  - Analyse code or binaries to identify potential security weaknesses or misconfigurations (e.g. perform static analysis to detect hardcoded credentials or use dynamic analysis tools to identify buffer overflow or injection vulnerabilities).
  - Inspect configurations to ensure that required security parameters are correctly applied (e.g. check that weak cipher suites are disabled, two-factor authentication is enabled, and least-privilege access controls are configured in the system).
  - Observe runtime behaviour to confirm that protections such as encryption, authentication, and integrity verification operate as intended (e.g. monitor network traffic to ensure data in transit is encrypted or observe system logs to verify successful validation of digital signatures during startup).
- **Assessment verdict:** Defines the pass/fail criteria.
  - **Pass:** The assessment is considered passed if the product demonstrably fulfils the requirement and meets the defined security thresholds. Examples of such thresholds include:
    - Minimum cryptographic strength (e.g. AES-128 or higher).
    - Password policy limits (e.g. minimum of 12 characters).
    - Login protection mechanisms (e.g. account lockout after five consecutive failed attempts).
    - Resistance to a specified attack potential (e.g. equivalent to CSA High/AVA_VAN.3 or higher).
  - **Fail:** The assessment is considered failed if the requirement is not fulfilled, or if the defined security thresholds are not achieved (e.g. insufficient key length, missing authentication enforcement, or inadequate resistance to the required attack potential).
- **Assessment evidence**: Defines the artefacts and documentation collected to demonstrate that the requirement has been assessed and fulfilled. The evidence shall be sufficient to enable independent verification of the assessment results and to demonstrate compliance with the relevant CRA essential requirements. Supporting evidence includes, where applicable:
  - Test or assessment reports showing the steps performed and results obtained.
  - Logs, configuration files, or audit traces demonstrating the implementation of the requirement.
  - Screenshots, captures, or console outputs confirming the correct execution or protection behaviour.
  - Relevant vendor or design documentation describing the applied security measures.

The indexing structure used in the present document is: ACC-BM-ESR-NNN where ACC identifies an assessment and compliance criterion, BM is the product short name (boot manager), ESR is the abbreviation of the essential requirement per clause 5, and NNN is the sequence number, matching the corresponding requirement identifier in clause 5.

6.1.1 Boot manager assessment considerations

Boot manager assessment may require:

- Platform-specific test environments due to hardware dependency.
- Specialised access to pre-OS execution environment.
- System integration context for component-level products.
- Physical access to hardware interfaces.

The assessment relies on manufacturer documentation of the test environment and any platform-specific limitations affecting assessment.

6.1.2 Scope of assessment

The present document defines requirements applicable to boot managers as products. Conformity assessment against the present document addresses the boot manager as the unit under assessment, independently of any finished product into which the boot manager may be integrated.

Where a requirement in clause 5 is conditional upon a capability of the platform or the boot target (see clause 4.4), the assessment uses the platform assumptions recorded in the product technical documentation. Verification that those assumptions hold in a deployed system is outside the scope of the present document.

6.1.3 Assessment report requirements

The assessment report includes:

- the product identifier and version;
- the declared use case and security profile described in Table 4.6.4-1;
- for each requirement in clause 5:
  - the requirement identifier;
  - a verdict of PASS, FAIL, or NOT APPLICABLE;
  - a reference to the evidence supporting the verdict; and
  - for a NOT APPLICABLE verdict, a justification meeting the criteria defined in clause 5.1.3.
- an overall conformity statement.

A verdict of NOT APPLICABLE for a requirement shall be justified only when all of the following conditions are met:

- the capability referenced in the requirement's Applicability statement is not implemented by the product;
- the product's declared use case shown in Table 4.6.4-1 does not assume that capability; and
- residual risks arising from the absence of that capability are documented in the product technical documentation.

6.2 Appropriate level of cybersecurity

6.2.0 Overview of requirements addressed in clause 6.2

Table 6.2.0-1: Requirements addressed in clause 6.2

Requirement	Profile applicability
REQ-BM-RRM-001	All
REQ-BM-RRM-002	All
REQ-BM-RRM-003	All
REQ-BM-RRM-004	If reliant on integrated components
REQ-BM-RRM-005	All

6.2.1 [ACC-BM-RRM-001]

Assessment reference: REQ-BM-RRM-001.

Assessment objective: verify that a cybersecurity risk assessment has been conducted for the boot manager product and is recorded in the product technical documentation.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.
- Required information: the product technical documentation, containing or referencing the cybersecurity risk assessment artefact.

Assessment activities:

- Confirm that the technical documentation contains or references a cybersecurity risk assessment artefact for the boot manager product.
- Confirm that the artefact identifies the boot manager product to which it applies and the version of the boot manager assessed.
- Confirm that the artefact records the methodology, scope, the assumptions from clause B.3 applicable to the product, identified risks, and conclusions.

Assessment verdict:

- Pass: the technical documentation contains a risk assessment artefact that identifies the product and version and records methodology, scope, applicable assumptions, the identified product risks, and final conclusions. For example, a risk assessment that names the product build, states its method and scope, lists the assumptions and identified risks, and draws conclusions satisfies the requirement.
- Fail: the requirement is not met if the technical documentation contains no risk assessment; if the risk assessment omits the product or version identification, methodology, scope, assumptions, identified risks, or conclusions; or where the recorded content is present but incomplete against those items.

Assessment evidence: reference to the risk assessment artefact in the technical documentation; record of completeness against the items above.

6.2.2 [ACC-BM-RRM-002]

Assessment reference: REQ-BM-RRM-002.

Assessment objective: confirm two properties of the risk assessment artefact: completeness of threat consideration against the applicable Annex B subset, and consistency of the identified risks with the declared use case.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.
- Required information: the cybersecurity risk assessment artefact; the declared use case under clause 5.1.2; the operational environment description in the technical documentation referencing clause 4.3; the use case definitions in clause 4.6.3; the security profile mapping in clause B.6.

Assessment activities:

- Derive the set of Annex B threats applicable to the declared use case and operational environment.
- Confirm that the risk assessment artefact considers each threat in that set, either by analysis or by a documented non-applicability rationale.
- Compare the identified risks against the use case definitions in clause 4.6.3 and the security profile mapping in clause B.6; confirm the identified risks are consistent with the declared use case.

Assessment verdict:

- Pass: the risk assessment addresses each Annex B threat applicable under the declared use case and operational environment, either by analysis or by a product-specific non-applicability rationale, and the identified risks remain within the threat surface implied by the declared use case. For example, a UC-VER product with network boot whose assessment addresses network protocol attacks and network boot persistence, and a sealed UC-IMM product that marks physical-access threats non-applicable with a rationale citing the sealing, satisfy the requirement.
- Fail: the requirement is not met if the assessment leaves an applicable threat unaddressed; if it surfaces risks the declared use case excludes; or where an update-capable product's assessment makes no mention of malicious update or update-distribution attacks.

Assessment evidence: derived set of applicable threats; cross-reference table mapping each applicable threat to its treatment in the risk assessment artefact; record of the consistency comparison.

6.2.3 [ACC-BM-RRM-003]

Assessment reference: REQ-BM-RRM-003.

Assessment objective: confirm that residual risks identified in the assessment are addressed by documented additional product-level measures.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.
- Required information: the cybersecurity risk assessment artefact; the applicable requirements derived from the declared use case across clauses 5.3 to 5.15; the technical documentation sections describing implemented security measures.

Assessment activities:

- For each risk identified in the risk assessment artefact, identify whether it is addressed by a requirement in clauses 5.3 to 5.15 applicable under the declared use case.
- For risks not addressed by those requirements, confirm that the technical documentation describes an additional product-level measure that addresses the risk.

Assessment verdict:

- Pass: every risk not addressed by clauses 5.3 to 5.15 is matched to a documented additional product-level measure. For example, a manufacturing-process supply-chain risk matched by a documented factory-side attested provisioning step, or a long-term key-handling risk matched by a documented HSM-backed key escrow with stated rotation, satisfies the requirement.

- Fail: the requirement is not met if a residual risk identified in the assessment has no corresponding additional measure documented; or where the documentation records the risk as accepted without naming a measure that addresses it.

Assessment evidence: Mapping of risks to requirements / additional measures; description of each additional measure as documented.

6.2.4 [ACC-BM-RRM-004]

Assessment reference: REQ-BM-RRM-004.

Assessment objective: confirm that the security needs the boot manager places on integrated components are stated and substantiated by evidence.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.
- Required information: the technical documentation section identifying integrated components on which the boot manager relies for security functions (per clause 4.4.3); the per-component statement of security needs; the documentary or test evidence supporting fulfilment of each need.

Assessment activities:

- Confirm that the integrated components on which the boot manager relies for security functions are enumerated in the technical documentation.
- For each enumerated component, confirm that the security needs placed on it are stated.
- For each stated need, confirm that the technical documentation records documentary evidence (for example a component certification, supplier attestation, or conformance statement) or functional test evidence showing the component meets the need.

Assessment verdict:

- Pass: each integrated component the boot manager relies on for security functions is enumerated in the technical documentation, with its security needs stated and evidence of fulfilment recorded. For example, a boot manager using an HSM for trust-anchor protection that enumerates the HSM with the needs "key confidentiality and physical tamper detection" and records the HSM's vendor conformity declaration covering those claims satisfies the requirement.
- Fail: the requirement is not met if an integrated component the boot manager depends on is missing from the enumeration; if the security needs placed on it are not stated; or where the evidence does not show the component meets those needs.

Assessment evidence: enumeration of integrated components; per-component security needs and evidence record.

6.2.5 [ACC-BM-RRM-005]

Assessment reference: REQ-BM-RRM-005.

Assessment objective: confirm that the technical documentation, taken as a whole, presents a coherent and complete risk-management record.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.
- Required information: the cybersecurity risk assessment artefact; the mapping of identified risks to applicable requirements under clauses 5.3 to 5.15; the description of additional measures implemented under REQ-BM-RRM-003 (where applicable); the supporting evidence for integrated components under REQ-BM-RRM-004 (where applicable) the product technical documentation.

Assessment activities:

- Confirm presence in the technical documentation of:

  a) the cybersecurity risk assessment artefact;

  b) the mapping of identified risks to applicable requirements in clauses 5.3 to 5.15;

  c) the description of any additional measures implemented under REQ-BM-RRM-003;

  d) the supporting evidence for integrated components under REQ-BM-RRM-004 where the boot manager relies on such components.

- Confirm that the recorded material is internally consistent: risks listed in the mapping match risks identified in the assessment artefact; additional measures address risks flagged as not covered by clauses 5.3 to 5.15; security needs stated for integrated components match the components enumerated in the assessment.

Assessment verdict:

- Pass: each applicable content item is present in the technical documentation and the recorded material is internally consistent across the items. For example, documentation in which each risk identified in the assessment also appears in the mapping with its assigned clause 5 requirements, and each integrated-component evidence item traces to a corresponding assessment entry, satisfies the requirement.

- Fail: the requirement is not met if an applicable content item is absent; if the recorded items contradict each other; or where a risk identified in the assessment is missing from the mapping or an additional measure is not traceable to a residual risk.

Assessment evidence: record of presence per content item; record of internal consistency check.

6.3 No known exploitable vulnerabilities

6.3.0 Overview of requirements addressed in clause 6.3

The present clause specifies assessment criteria for the requirements in clause 5.3.

Table 6.3.0-1: Requirements addressed in clause 6.3

Requirement	Profile applicability
REQ-BM-KEV-001	All
REQ-BM-KEV-002	MEDIUM, HIGH
REQ-BM-KEV-003	LOW only

6.3.1 [ACC-BM-KEV-001]

Assessment reference: REQ-BM-KEV-001.

Assessment objective: verify that, at the moment of placing on the market, the boot manager is free of known exploitable vulnerabilities in its code, configuration, and cryptographic mechanisms.

Assessment preparation:

- Test environment: a representative build of the boot manager as placed on the market.

- Required information: the pre-market vulnerability assessment record, the SBOM and component inventory, the list of cryptographic mechanisms used, and the declared component and dependency versions.

- Required tools: vulnerability scanners; SBOM analysis tools; access to recognised vulnerability databases including EUVD.

Assessment activities:

- Review the pre-market vulnerability assessment record and confirm that its scope covers the boot manager's code, configuration, and cryptographic mechanisms.
- Cross-reference the SBOM and component inventory against EUVD and other recognised vulnerability databases as of a cut-off date recorded in the assessment evidence.
- Inspect the shipped configuration for known insecure settings (e.g. deprecated cryptographic algorithms, disabled verification).
- Verify that for each vulnerability identified during cross-reference, the boot manager either:
  a) does not contain the affected component or affected version in the shipped build, or
  b) does not expose the vulnerable component through its documented attack surface as specified in clause 4.

Assessment verdict:

- Pass: no known exploitable vulnerability is present in components reachable through the boot manager's documented attack surface. A typical Pass record shows several cross-reference hits against bundled libraries, each either absent from the shipped build (version-bumped past the affected range, or component replaced) or non-reachable through the boot manager's interfaces (e.g. a log-parsing CVE in a library the boot manager never feeds external input to).
- Fail: an exploitable vulnerability remains reachable, or a non-reachable classification lacks architectural support. Recognisable patterns include an unpatched CVE in the bundled signature-verification library invoked during verified boot; an unpatched CVE in a network-protocol library invoked during network boot; or a "not exploitable" claim for a bundled crypto library CVE without identifying which clause 4 architectural element excludes the vulnerable code path.

Assessment evidence: this may include any of the following: pre-market vulnerability assessment report; SBOM and component inventory; record of the EUVD cross-reference with cut-off date; configuration inspection records; clause 4 architectural references supporting any non-reachable classification.

6.3.2 [ACC-BM-KEV-002]

Assessment reference: REQ-BM-KEV-002.

Assessment objective: verify that security updates addressing known exploitable vulnerabilities are available throughout the declared support period.

Assessment preparation:

- Required tools: access to the update channel; signature verification tools.
- Test environment: the boot manager in its configuration to receive updates.
- Required information: the declared support period as stated in the product technical documentation, the update distribution channel, the update release history, and the mapping between addressed vulnerabilities and released updates.
- Required tools: access to the update channel; signature verification tools.

Assessment activities:

- Review the declared support period and confirm it is stated in the product technical documentation.
- Verify that the vulnerability handling process produces security updates addressing known exploitable vulnerabilities within the support period; alignment with prEN 40000-1-3 [i.4] is one way to demonstrate this.
- Confirm that the update distribution channel is operational and that delivered updates are signed and verifiable (see the assessment of REQ-BM-INT-005 in clause 6.8.5).

- Sample the update release history and confirm that past known exploitable vulnerabilities applicable to the boot manager have been addressed by released updates.

Assessment verdict:

- Pass: security updates addressing known exploitable vulnerabilities are available throughout the declared support period. A typical Pass record shows the technical documentation declaring a support period; the update distribution channel operational; and a release history that includes security updates issued in response to published CVEs against bundled components within the support period window.

- Fail: security updates are not consistently available. Recognisable patterns include a support period stated but no operational distribution channel (the update endpoint is unreachable or serves unsigned artefacts); a published CVE applicable to a bundled component remaining outstanding without an addressing update; or technical documentation that does not declare a support period at all.

Assessment evidence: support period statement from technical documentation; update release history; sample signed update artefacts; vulnerability-to-update mapping records.

6.3.3 [ACC-BM-KEV-003]

Assessment reference: REQ-BM-KEV-003.

Assessment objective: verify that, for a boot manager without update capability, the product technical documentation states the absence of update capability as a residual risk, records the date of the pre-market vulnerability assessment, and describes the defence-in-depth measures compensating for the absence of post-deployment remediation.

Assessment preparation:

- Test environment: not applicable; assessment is documentation review.

- Required information: the product technical documentation.

Assessment activities:

- Inspect the product technical documentation and confirm each of the following is present:

  - the absence of update capability identified as a residual risk;

  - the date of the pre-market vulnerability assessment;

  - the defence-in-depth measures mitigating the inability to remediate post-deployment vulnerabilities.

Assessment verdict:

- Pass: the technical documentation states the absence of update capability as a residual risk, records the pre-market vulnerability assessment date, and describes defence-in-depth measures that each address an identified consequence of the no-update condition. A typical Pass record might describe a shipping configuration that enables only a single boot target with network-boot and removable-media interfaces disabled, restricted I/O exposure that limits the parsers reachable from untrusted input, and a documented operational environment requiring controlled physical access to the device with each linked to the exposure it compensates for.

- Fail: the technical documentation does not adequately document one or more required elements. Recognisable patterns include the absence of update capability listed as a feature note without acknowledging it as a residual risk; an absent or undated vulnerability assessment statement; or defence-in-depth measures described in generic terms (e.g. "standard hardening measures") without identifying which architectural element addresses which exposure.

Assessment evidence: relevant extracts from the product technical documentation.

6.4 Secure by default configuration

6.4.0 Overview of requirements addressed in clause 6.4

The present clause specifies assessment criteria for the requirements in clause 5.4.

Table 6.4.0-1: Requirements addressed in clause 6.4

Requirement	Profile applicability
REQ-BM-SBD-001	MEDIUM, HIGH (LOW if implemented)
REQ-BM-SBD-002	All
REQ-BM-SBD-003	All (if configuration capability with password authentication)
REQ-BM-SBD-004	MEDIUM, HIGH (LOW if implemented)
REQ-BM-SBD-005	All (if configuration capability)
REQ-BM-SBD-006	All (if configuration capability)

6.4.1 [ACC-BM-SBD-001]

Assessment reference: REQ-BM-SBD-001.

Assessment objective: verify that the boot manager has cryptographic validation enabled by default when placed on the market.

Assessment preparation:

- Test environment: a representative build of the boot manager in its shipped state, without any post-manufacturing configuration change.
- Required information: the documentation of the default state of cryptographic validation; the interfaces and mechanisms controlling the validation state.
- Required tools: a trusted-key test image (signed with a key in the documented trust store); an untrusted test image (unsigned or signed with a key not in the trust store).

Assessment activities:

- Submit the trusted-key test image to the boot manager and confirm that validation succeeds and boot completes.
- Submit the untrusted test image and confirm that validation rejects the image and boot is halted.
- Inspect the boot manager's default configuration and confirm that cryptographic validation is enabled without any prior enabling step.

Assessment verdict:

- Pass: cryptographic validation is enabled in the shipped state. A typical Pass record shows the untrusted test image rejected at the validation step with boot halted, the trusted-key test image accepted and boot completing, and the shipped configuration showing validation enabled without any operator-side step preceding these tests.
- Fail: cryptographic validation is not effective in the shipped state. Recognisable patterns include the untrusted test image accepted and boot proceeding (validation effectively disabled or bypassed); a shipped configuration that documents validation as disabled by default and requires an enable step before validation can occur; or validation that runs but treats untrusted signatures as warnings rather than rejections.

Assessment evidence: record of the default-state configuration; record of boot with a signed image; record of rejection of an untrusted image.

6.4.2 [ACC-BM-SBD-002]

Assessment reference: REQ-BM-SBD-002.

Assessment objective: verify the absence of unauthorised or undocumented mechanisms for bypassing the boot manager's authentication or security controls.

Assessment preparation:

- Test environment: the boot manager in its shipped state with access to all exposed interfaces.
- Required information: the documented list of interfaces, commands, and authentication mechanisms; the product technical documentation.
- Required tools: interface enumeration tools appropriate to the platform; credential testing tools; binary analysis tools where source is not available.

Assessment activities:

- Enumerate the interfaces exposed by the boot manager (e.g. shell, management commands, debug endpoints) and compare against the documented interface list; investigate any undocumented interface.
- Test each authenticated interface against default and hardcoded credential dictionaries and confirm that no such credential grants access.
- Inspect documentation and binary for maintenance or service modes; where present, confirm that entry is authenticated, documented, and does not bypass authentication or security controls.
- Review the binary for strings or code paths suggestive of backdoor or bypass functionality.

Assessment verdict:

- Pass: the boot manager exposes no unauthorised or undocumented bypass mechanism. A typical Pass record shows interface enumeration matching the documented list with no undocumented endpoints discovered; credential testing against default and hardcoded credential dictionaries returning no successful access; and any maintenance or service mode found being authenticated, documented, and operating within the boot manager's security controls.
- Fail: the boot manager exposes a bypass mechanism. Recognisable patterns include an undocumented debug or service interface accessible without authentication (e.g. a JTAG or serial console exposed in the shipped state); a default or hardcoded credential accepted by an authenticated interface (e.g. admin/admin on a management endpoint); or a maintenance mode that disables or weakens authentication or security controls upon entry.

Assessment evidence: interface enumeration record; credential test record; maintenance or service mode analysis; binary inspection findings.

6.4.3 [ACC-BM-SBD-003]

Assessment reference: REQ-BM-SBD-003.

Assessment objective: verify that the boot manager supports the establishment of passwords during initial deployment for interfaces that can compromise the system's core security guarantees.

Assessment preparation:

- Test environment: the boot manager in a state representing initial deployment (first boot, manufacturer defaults).
- Required information: the identification of interfaces that can compromise core security guarantees and are protected by passwords, from the product technical documentation; the initial-deployment workflow.
- Required tools: interaction tool for the identified interfaces.

Assessment activities:

- Identify, from the product technical documentation, each interface that can compromise core security guarantees and is protected by password authentication.

- Execute the initial-deployment workflow and confirm that access to each such interface is either gated on password establishment during the workflow or denied until a password is established.
- Confirm that no empty or default password grants access to the identified interfaces in the deployed state.

Assessment verdict:

- Pass: for each identified interface, the boot manager either requires a password to be set during the initial-deployment workflow or denies access until one is set; and no empty or default password grants access in the deployed state. A typical Pass record shows the initial deployment workflow including a "set password" step that gates access to a management interface, and post-deployment attempts using common default credentials being rejected.
- Fail: the boot manager does not gate access on password establishment for at least one identified interface, or a default credential grants access. Recognisable patterns include a management interface accessible immediately after first boot without any prompt establishment the password; or a default password such as "admin" or "root" accepted after initial deployment.

Assessment evidence: list of identified interfaces; record of the initial deployment workflow; records of access attempts after deployment.

6.4.4 [ACC-BM-SBD-004]

Assessment reference: REQ-BM-SBD-004.

Assessment objective: verify that fallback to a boot mode weakening verification or rollback protection requires explicit authorisation and does not occur automatically.

Assessment preparation:

- Test environment: the boot manager configured with verification and rollback protection active, with the ability to induce failure conditions in normal boot paths.
- Required information: the enumeration of fallback boot modes, the conditions under which each may be entered, and the authorisation step required for each.
- Required tools: fault-injection tools capable of triggering documented failure conditions (e.g. corrupted primary image, failed rollback counter read).

Assessment activities:

- For each documented fallback mode, induce the failure condition that would normally trigger consideration of fallback.
- Confirm that no transition to a weakened fallback mode occurs automatically in response.
- Confirm that the boot manager either halts or remains in a mode preserving verification and rollback protection in the absence of the documented authorisation step.

Assessment verdict:

- Pass: under each induced failure condition, the boot manager does not transition automatically to a fallback mode that weakens verification or rollback protection. A typical Pass record shows the boot manager halting at a verification-failed state when the primary boot image is corrupted, and halting rather than continuing the boot with rollback protection bypassed when the rollback counter read fails.
- Fail: under at least one induced failure condition, the boot manager transitions automatically to a weakened fallback mode. Recognisable patterns include a corrupted primary image triggering an automatic load of an unsigned recovery image; a failed rollback counter read causing the boot manager to bypass rollback protection on subsequent boots; or a verification key revocation triggering an automatic fallback to previously-trusted but now-revoked keys.

Assessment evidence: enumeration of fallback modes; records of fault-injection tests and the boot manager's response under each.

6.4.5 [ACC-BM-SBD-005]

Assessment reference: REQ-BM-SBD-005.

Assessment objective: verify that the boot manager presents a prominent indication appropriate for the device's interface capabilities when a security-relevant protection is reduced or disabled.

Assessment preparation:

- Test environment: the boot manager configured for observation through the indication channel(s) appropriate to the device (display, audio output, LED, or log capture).
- Required information: the enumeration of security-relevant protections whose reduction or disablement triggers an indication; the form and observation channel of each indication.
- Required tools: configuration tool to reduce or disable each enumerated protection; capture means appropriate to the indication channel (e.g. display capture, audio capture, LED state observation, log capture).

Assessment activities:

- For each enumerated security-relevant protection, reduce or disable the protection via its configuration interface.
- Observe the indication channel during and after the change and confirm that the documented indication is presented.
- Restore the protection and confirm that the indication is removed or updated consistently with the documented behaviour.

Assessment verdict:

- Pass: reducing or disabling each enumerated protection produces the documented indication on the appropriate channel; restoring the protection updates or removes the indication consistently. A typical Pass record shows a boot manager with a display presenting an on-screen "verification disabled" message when verified boot is turned off in configuration, and the message clearing when verification is restored; for a headless device, the same Pass record shows a structured log entry recorded at the time of the configuration change naming the protection that was reduced.
- Fail: reducing or disabling a protection produces no indication on the documented channel, or the indication is inconsistent with the documented behaviour. Recognisable patterns include verified boot being disabled via configuration without any indication appearing on any documented channel; an LED indicator that remains in "secure" state after a security protection is reduced; or a log entry that mentions the configuration change without naming the protection that was reduced.

Assessment evidence: enumeration of security-relevant protections; records of indication channel capture for each state transition.

6.4.6 [ACC-BM-SBD-006]

Assessment reference: REQ-BM-SBD-006.

Assessment objective: verify that the boot manager supports restoring its configuration to specified secure default settings.

Assessment preparation:

- Test environment: the boot manager configured to a non-default state via documented configuration interfaces.
- Required information: the specification of the secure default configuration; the restoration mechanism; any documented limitations to restoration arising from immutable configuration or keys.
- Required tools: configuration tool capable of applying and reading configuration state; inspection tool for immutable-configuration elements where relevant.

Assessment activities:

- Apply a documented non-default configuration and confirm the change is reflected in the boot manager's configuration state.
- Invoke the restoration mechanism per the documented procedure.
- Compare the resulting configuration against the specified secure default; confirm that all mutable configuration items are restored and that any items not restored correspond to documented immutable-configuration limitations.

Assessment verdict:

- Pass: the restoration mechanism returns all mutable configuration items to the specified secure default, and any items not restored correspond to documented immutable-configuration limitations. A typical Pass record shows the boot manager configured with non-default settings (e.g. custom password policy, additional trust anchors) returned to the documented secure default after the restoration mechanism is invoked, with a fuse-burned trust anchor remaining unchanged and explicitly listed in the documented limitations.
- Fail: mutable configuration items are not restored to the specified secure default, or limitations are not documented as they appear in practice. Recognisable patterns include the restoration mechanism completing but leaving a non-default configuration item in place (e.g. a custom password policy persists); or an immutable item excluded from restoration without appearing in the documented limitations list.

Assessment evidence: specification of secure defaults; record of non-default configuration; record of configuration after restoration; documented limitations reference.

6.5 Secure updates

6.5.0 Overview of requirements addressed in clause 6.5

The present clause specifies assessment criteria for the requirements in clause 5.5.

Table 6.5.0-1: Requirements addressed in clause 6.5

Requirement	Profile applicability
REQ-BM-SU-001	MEDIUM, HIGH (if UI implemented)
REQ-BM-SU-002	MEDIUM, HIGH
REQ-BM-SU-003	All (if configuration capability)
REQ-BM-SU-004	MEDIUM, HIGH
REQ-BM-SU-005	MEDIUM, HIGH (if HW present)
REQ-BM-SU-006	MEDIUM, HIGH
REQ-BM-SU-007	MEDIUM, HIGH

6.5.1 [ACC-BM-SU-001]

Assessment reference: REQ-BM-SU-001.

Assessment objective: verify that, where a user interface is available, the boot manager indicates when an update is available, or relies on authorised components to present that indication.

Assessment preparation:

- Test environment: the boot manager operated through its user interface, with the means to induce an update being available and no update being available, and any additional update states the product indicates.
- Required information: the set of update conditions the product indicates and the indication used for each; where the boot manager relies on authorised components, the identity of those components and the interface through which status is passed to them.

- Required tools: display or output capture for the user interface; an inspection method for any interface exposed to authorised components.

Assessment activities:

- Induce an available update and, separately, no available update; observe the user interface and record the indication presented in each case.
- Where the product indicates additional update states (for example update in progress or update failed), induce and observe those as well.
- Where the boot manager relies on authorised components instead of a user-visible indication, exercise each condition and confirm the corresponding status reaches the component through the declared interface.

Assessment verdict:

- Pass: an available update and the absence of one each produce a distinct, observable indication, or that status reaches a declared authorised component. For a boot manager with a display, a pending update surfaces an on-screen state distinct from the up-to-date state; for one that delegates, the firmware management service receives the matching status over the declared interface.
- Fail: an available update produces no indication and reaches no authorised component. A boot manager that continues to show an up-to-date state while an update is pending does not meet the requirement.

Assessment evidence: the list of update conditions and their indications; user-interface captures taken under each induced condition; records of the of interface inspection for code paths that rely on authorised components.

6.5.2 [ACC-BM-SU-002]

Assessment reference: REQ-BM-SU-002.

Assessment objective: verify that, when network capability is available, the boot manager supports a mechanism to check whether an update is available, or relies on authorised components to perform that check.

Assessment preparation:

- Test environment: the boot manager configured with network capability, in a controlled environment where the update source can be made to offer an available update or none.
- Required information: the update-availability check mechanism the product uses, including its protocol, endpoint, and schedule where applicable; or, where the check is delegated, the authorised components relied upon and the interface to them.
- Required tools: a controllable update source or emulator; a network traffic analyser; an inspection method for any interface to authorised components.

Assessment activities:

- Make an update available at the source and invoke the check; record whether the boot manager, or the component it relies on, reports the update as available.
- Offer no update at the source and invoke the check; record what is reported.
- Where the check is delegated, exercise it and confirm the boot manager hands the check to the authorised component through the declared interface.

Assessment verdict:

- Pass: the check reports an update as available when one is present and reports none when the source offers none, whether the boot manager performs the check itself or delegates it. A boot manager that queries its update endpoint and surfaces a pending update, or one that hands the query to the operating system's update service, both satisfy the requirement.

- Fail: the check does not reflect what the source offers. A boot manager that reports no update while one is available, that never contacts the source, or that neither performs nor delegates the check, does not meet the requirement.

Assessment evidence: the specification of the check mechanism; records of the check with an update available and with none available; records of the delegated interface where authorised components perform the check.

6.5.3 [ACC-BM-SU-003]

Assessment reference: REQ-BM-SU-003.

Assessment objective: verify that the boot manager allows security policy to be updated through configuration, allows features to be disabled so that can be addressed, and retains this configuration update capability for the product's support period.

Assessment preparation:

- Test environment: the boot manager accessible through its configuration interface.
- Required information: the security policy items that can be updated through configuration; the features that can be disabled; the documented commitment to retain configuration update capability over the product's lifetime.
- Required tools: a configuration tool; an inspection method for the boot manager's active policy and feature state.

Assessment activities:

- Apply a security policy change through the configuration interface and confirm it takes effect in the boot manager's active state (for example tightening a verification policy or replacing a trust anchor).
- Disable a feature through the configuration interface and confirm it is no longer active.
- Review how configuration update capability is retained over the product's lifetime; confirm nothing disables it beforehand (for example a fixed expiry or a one-time configuration lock) and that the commitment covers the declared lifetime.

Assessment verdict:

- Pass: a security policy change and a feature being disabled both take effect through the configuration interface, and the configuration update path stays available for the product's lifetime. A boot manager that accepts a stricter verification policy, lets a vulnerable network boot path be switched off, and keeps that configuration path open for its declared lifetime satisfies the requirement.
- Fail: a configuration change is rejected or has no effect, a feature cannot be disabled, or the configuration update path closes early. A boot manager that locks its configuration after first boot, or whose configuration update capability lapses while the product is still in service, does not meet the requirement.

Assessment evidence: records of the policy-change and feature-disable tests showing before and after state; the documented commitment to configuration update availability over the product's lifetime.

6.5.4 [ACC-BM-SU-004]

Assessment reference: REQ-BM-SU-004.

Assessment objective: verify that the boot manager, where it performs on-update verification, accepts updates only from sources that are both authenticated and authorised.

Assessment preparation:

- Test environment: the boot manager in its on-update verification configuration.

- Required information: the source authentication mechanism the product uses (for example signed update manifests, authenticated transport, or trusted source identifiers) and the authorisation policy applied to authenticated sources.
- Required tools: an update source that is authorised; one that is unauthenticated; one that is authenticated but not authorised.

Assessment activities:

- Offer an update from the authorised source and confirm it is accepted into the subsequent update steps.
- Offer an update from the unauthenticated source and confirm it is rejected before any update action begins.
- Offer an update from a source that authenticates successfully but is not authorised (for example a valid signature under an identity outside the authorised set) and confirm rejection.
- Confirm the source authentication and authorisation decision is reached before the package verification of REQ-BM-INT-005 is invoked (assessed in clause 6.8.5).

Assessment verdict:

- Pass: only the authenticated, authorised source is accepted, and both the unauthenticated source and the authenticated-but-unauthorised source are turned away before any update action. A boot manager that fetches from a trusted, identity-checked update server while refusing one offered by an unknown server, and rejecting a package signed by a retired signer no longer in the authorised set, satisfies the requirement.
- Fail: an update is accepted from a source that is not authenticated, or from one that authenticates but falls outside the authorisation policy, or the source decision is made only after the update has begun to be applied. A boot manager that accepts any reachable update server, or admits a correctly signed package from a decommissioned signer, does not meet the requirement.

Assessment evidence: the source authentication and authorisation specification; test records for the authorised, unauthenticated, and authenticated-but-unauthorised sources; the record showing the source decision precedes package verification.

6.5.5 [ACC-BM-SU-005]

Assessment reference: REQ-BM-SU-005.

Assessment objective: verify that, when hardware security components are available, the boot manager stores the keys used for update verification in those components.

Assessment preparation:

- Test environment: the boot manager on a platform whose hardware security components are documented and inspectable.
- Required information: whether hardware security components are available on the target platform; where they are, the identity of those components and the interface through which the keys are stored and used.
- Required tools: an inspection method for the hardware security component appropriate to the platform (for example a TPM command interface, or a secure-element interface within the authorised scope).

Assessment activities:

- Determine whether hardware security components are available on the platform under assessment; where none are, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Where hardware security components are available, locate where the update-verification keys are held; confirm they reside in the hardware security component and not in storage reachable by software.
- Exercise an update verification and confirm the boot manager uses the keys through the component's protected interface, with no key material leaving the component.

Assessment verdict:

- Pass: the update-verification keys are held inside the available hardware security component, and verification draws on them through the component's protected interface without the key material being exported. A boot manager that holds its update-verification key in a TPM or secure element and calls the component to perform the check, rather than reading the key out into software, satisfies the requirement.
- Fail: a hardware security component is present yet the update-verification keys sit in software-readable storage, or the boot manager exports the key from the component to verify in software. A boot manager that loads its verification key from an unprotected flash region while a TPM is available does not meet the requirement.

NOTE: Trust anchors may be firmware-embedded; assessment should consider documented storage location regardless of TPM/HSM presence.

Assessment evidence: record of whether hardware security components are present on the platform; record of where the update-verification keys are held; record of the interface through which the keys are used during verification.

6.5.6 [ACC-BM-SU-006]

Assessment reference: REQ-BM-SU-006.

Assessment objective: verify that the boot manager, where it performs on-update verification, keeps its update logic isolated from the normal boot.

Assessment preparation:

- Test environment: the boot manager exercised in both normal boot and on-update verification.
- Required information: how the update logic is separated from the normal boot path (for example a separate execution context, memory isolation, distinct code paths, or a state-machine boundary), and where the two paths share resources.
- Required tools: an inspection method for the boot manager's runtime state and the code paths it activates, as supported by the platform.

Assessment activities:

- Execute a normal boot and inspect which code paths and memory regions become active; confirm the update logic is not among them.
- Trigger on-update verification and inspect which code paths and memory regions become active; confirm the normal boot path's decision state (boot target selection, trust anchors, verification policy) is not writable from the update logic.
- Where the two paths share resources, follow the handoff between them and confirm the update logic cannot alter normal-boot state through that shared resource, and the reverse.

Assessment verdict:

- Pass: the update logic stays inactive during normal boot, and during update handling it cannot reach or modify the state the boot path relies on. A boot manager that runs update parsing in a separate context entered only after a stage boundary, so that a fault while parsing a malformed update package cannot overwrite the boot target or trust anchors, satisfies the requirement.
- Fail: the update logic runs as part of normal boot, or the two paths share memory or state such that update handling can influence a boot decision. A boot manager whose update-package parser writes into the same buffers that hold the boot target selection, so that a malformed package can redirect the boot, does not meet the requirement.

Assessment evidence: execution trace of normal boot; execution trace of on-update verification; record of the separation mechanism and of any shared-resource handoff.

6.5.7 [ACC-BM-SU-007]

Assessment reference: REQ-BM-SU-007.

Assessment objective: verify that the boot manager applies updates atomically, so that an interrupted or failed update does not leave it in a partially updated state, and that on a failure it reverts to the previously operational state and remains bootable.

Assessment preparation:

- Test environment: the boot manager in its update configuration, with the means to interrupt an update in progress (for example cutting power, forcing a reset, or severing the update source) and to corrupt or truncate an update package so that it fails partway.
- Required information: the point at which an update becomes committed; how the previously operational state is retained and selected on failure (for example a redundant image or a recovery image); the conditions under which the boot manager treats an update as failed.
- Required tools: a controllable update source; a means to interrupt the update at chosen points; an inspection method for the active boot manager image and configuration after each trial; a way to confirm the device still boots.

Assessment activities:

- Determine whether the boot manager applies updates itself; where update application is performed by a trusted platform component instead, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Apply a valid update without interruption and confirm it completes and the boot manager boots from the updated state.
- Interrupt the update at several points across its progress (before commit, during the write, and around the commit point); after each interruption, restart the device and record which state it boots into and whether any partially updated state remains.
- Supply an update that fails partway (for example a package truncated or corrupted after application has begun) and confirm the boot manager abandons it and returns to the previously operational state.
- After each failed or interrupted trial, confirm the boot manager boots and runs the previously operational image and configuration.

Assessment verdict:

- Pass: every interrupted or failed update leaves the boot manager either fully updated or fully on its previous operational state, never in between, and the device boots in each case. A boot manager that writes the new image to a second slot and switches only after the write and verification complete, so that power loss mid-write simply boots the old slot, satisfies the requirement.
- Fail: an interrupted or failed update leaves the boot manager unable to boot, or running a mixture of old and new components. A boot manager that overwrites its only image in place, so that power loss during the write leaves a corrupt image and no working fallback, does not meet the requirement.

Assessment evidence: record of the uninterrupted update; per-trial records of the interruption point, the state booted afterwards, and whether partial state was found; record of the retained previous state and the selection mechanism.

6.6 Authentication and access control

6.6.0 Overview of requirements addressed in clause 6.6

The present clause specifies assessment criteria for the requirements in clause 5.6.

Table 6.6.0-1: Requirements addressed in clause 6.6

Requirement	Profile applicability
REQ-BM-AC-001	All (if configuration capability)
REQ-BM-AC-002	All (if configuration capability)
REQ-BM-AC-003	All (if configuration capability)
REQ-BM-AC-004	All (if configuration capability with password authentication)
REQ-BM-AC-005	MEDIUM, HIGH (if verified or measured boot and configuration capability
REQ-BM-AC-006	All (if configuration capability)
REQ-BM-AC-007	MEDIUM, HIGH (if verified or measured boot)
REQ-BM-AC-008	All
REQ-BM-AC-009	All (if authority transfer implemented)
REQ-BM-AC-010	All
REQ-BM-AC-011	All (if authority transfer implemented)

6.6.1 [ACC-BM-AC-001]

Assessment reference: REQ-BM-AC-001.

Assessment objective: verify that changes to trusted keys, certificates, or trust anchor databases are authorised only through direct physical presence at the device, or through a signed artefact whose authenticity and integrity the boot manager verifies before applying it.

Assessment preparation:

- Test environment: the boot manager configured with an established trust anchor database, a physical-presence detection mechanism where supported, and the interface through which signed artefacts are submitted.
- Required information: the physical-presence detection mechanism, where implemented; the signed-artefact format, the trusted signer identities, and the verification algorithm.
- Required tools: a hardware means of asserting and de-asserting physical presence (for example a jumper, button, or hardware key); an artefact generation tool able to sign with trusted and untrusted signers.

Assessment activities:

- Attempt a change to a trusted key, certificate, or trust anchor with neither physical presence asserted nor a signed artefact submitted; confirm it is rejected.
- Attempt the same change with physical presence asserted; confirm it is accepted.
- Attempt the same change through an artefact signed by a trusted signer with valid authenticity and integrity; confirm it is accepted.
- Attempt the same change through an artefact signed by an untrusted signer, or with an altered signature or content; confirm it is rejected before the change is applied.

Assessment verdict:

- Pass: a change to a trust anchor takes effect only when physical presence is asserted at the device or when it arrives in a signed artefact that verifies against a trusted signer, and every attempt with neither is refused. A trust-anchor update offered over the configuration interface with no presence assertion and no valid signature is refused, while the same update carried in an artefact signed by the trusted key is applied.
- Fail: a trust-anchor change takes effect through neither asserted presence nor a verified signed artefact, or an artefact with a broken or untrusted signature is applied. A boot manager that accepts a trust-anchor edit over its configuration interface without any presence assertion, or that applies an artefact whose signature it never checks, does not meet the requirement.

Assessment evidence: specification of the physical-presence mechanism; specification of signed-artefact submission; test records for each of the four attempt categories.

6.6.2 [ACC-BM-AC-002]

Assessment reference: REQ-BM-AC-002.

Assessment objective: verify that the boot manager provides a means to protect boot-process configuration settings against unauthorised modification, and that where this protection is configured the protected settings cannot be modified without authorisation.

Assessment preparation:

- Test environment: the boot manager accessible through its configuration interface.
- Required information: the boot-process configuration settings the boot manager can protect (for example boot order, boot parameters, boot target or mode selection); how the protection is configured; the authorisation required once it is configured.
- Required tools: a configuration tool; an inspection method for the boot manager's configuration and its protection state.

Assessment activities:

- Identify the boot-process configuration settings the boot manager can protect, and confirm the protection can be configured for them.
- With protection configured for a setting, attempt to modify it without the required authorisation; confirm the modification is rejected.
- With protection configured, perform an authorised modification of the same setting; confirm it is accepted.
- Where protection is not configured for a setting, confirm the boot manager behaves as documented; the requirement makes the protection conditional on configuration, so an unprotected setting is not itself a failure.

Assessment verdict:

- Pass: the boot manager can protect its boot-process configuration settings, and once that protection is configured an unauthorised modification is refused while an authorised one succeeds. A boot manager that, with boot-order protection enabled, rejects a change to the boot order over an unauthenticated interface but accepts it after authorisation, satisfies the requirement.
- Fail: the boot manager offers no means to protect boot-process configuration settings, or a setting configured as protected can still be modified without authorisation. A boot manager that exposes a "lock boot configuration" option which, once set, still lets the boot order be rewritten without authorisation, does not meet the requirement.

Assessment evidence: the list of protectable boot-process configuration settings and how protection is configured; records of modification attempts with protection configured (authorised and unauthorised), and of the documented behaviour where protection is not configured.

6.6.3 [ACC-BM-AC-003]

Assessment reference: REQ-BM-AC-003.

Assessment objective: verify that the boot manager requires explicit authentication before any modification of security-critical configuration.

Assessment preparation:

- Test environment: the boot manager with its configuration interface available and an authentication mechanism configured.
- Required information: the security-critical configuration settings (for example settings that weaken or disable secure defaults, downgrade security parameters, or change verification policies or trust anchors); the authentication mechanism required before such a modification.

- Required tools: a configuration interaction tool; a means of attempting modification without authenticating and of replaying or scripting a modification to test for an implicit path.

Assessment activities:

- For a security-critical setting, attempt a modification without authenticating; confirm it is rejected.
- Attempt the same modification with the required authentication explicitly performed; confirm it is accepted.
- Attempt the modification through a path that supplies no fresh authentication (for example replaying a prior session's credentials, or a scripted invocation that bypasses the authentication step); confirm it is rejected.

Assessment verdict:

- Pass: a security-critical setting changes only after authentication is explicitly performed, and a modification that presents no fresh authentication is refused. A boot manager that lets a verification policy be relaxed only once the operator authenticates at that point, while rejecting a scripted attempt that reuses an earlier session, satisfies the requirement.
- Fail: a security-critical setting can be modified without authentication, or through a path that inherits or replays authentication rather than requiring it explicitly. A boot manager that applies a trust-anchor change carried in an automated sequence without a distinct authentication step does not meet the requirement.

Assessment evidence: the list of security-critical configuration settings and the authentication required; test records for the unauthenticated attempt, the explicitly authenticated attempt, and the implicit-path attempt.

6.6.4 [ACC-BM-AC-004]

Assessment reference: REQ-BM-AC-004.

Assessment objective: verify that, where passwords are used for authentication, the boot manager limits authentication attempts through hardware-bound throttling, by limiting attempts, imposing increasing delays, or using a deliberately slow key derivation function.

Assessment preparation:

- Test environment: the boot manager configured with password authentication on a documented interface.
- Required information: the throttling mechanism the product uses (its type and parameters, and whether it is hardware-bound or based on a slow key derivation function); the intended cost per authentication attempt.
- Required tools: an automated authentication attempt tool; a timing measurement tool.

Assessment activities:

- Measure the cost per authentication attempt under sustained unsuccessful attempts (time per attempt, the attempt count before lockout, and the delay pattern across attempts).
- Confirm the measured behaviour matches the throttling the product documents.
- Where the mechanism is hardware-bound, repeat unsuccessful attempts across power cycles and confirm the throttling state is carried over rather than reset.
- Where a slow key derivation function is the mechanism, confirm its parameters (for example iteration count or memory cost) produce the intended per-attempt cost.

Assessment verdict:

- Pass: the measured per-attempt cost and the lockout or delay behaviour match the documented mechanism, and a hardware-bound mechanism keeps its throttling state across power cycles. A boot manager that imposes an increasing delay backed by a monotonic hardware counter, so that power-cycling does not clear the accumulated delay, or one whose slow key derivation function measures at its documented cost, satisfies the requirement.

- Fail: throttling is absent, measures materially weaker than documented, or resets so that attempts can be retried freely. A boot manager whose attempt counter returns to zero on reboot, allowing unlimited guesses across power cycles, or whose claimed slow key derivation function completes far faster than documented, does not meet the requirement.

Assessment evidence: the throttling specification; per-attempt cost measurements; records of throttling persistence across power cycles where the mechanism is hardware-bound.

6.6.5 [ACC-BM-AC-005]

Assessment reference: REQ-BM-AC-005.

Assessment objective: verify that, where security configuration policies are provided as separate signed artefacts, the boot manager verifies their authenticity and integrity before importing or enforcing them.

Assessment preparation:

- Test environment: the boot manager with the interface for importing or enforcing security configuration policies accessible.
- Required information: whether security configuration policies are provided as separate signed artefacts; where they are, the artefact format, the trusted signer identities, and the verification algorithm.
- Required tools: an artefact generation tool able to produce artefacts signed by trusted and untrusted signers, and to alter content after signing.

Assessment activities:

- Where security configuration policies are not provided as separate signed artefacts (for example policies built into the boot manager), record the requirement as not applicable and close this assessment per clause 5.1.3.
- Submit an artefact signed by a trusted signer with unaltered content; confirm the policy is imported and enforced.
- Submit an artefact whose content has been altered after signing, so the signature no longer matches; confirm it is rejected before import or enforcement.
- Submit an artefact signed by an untrusted signer; confirm it is rejected before import or enforcement.

Assessment verdict:

- Pass: a policy artefact is imported and enforced only when it carries a trusted signer's signature over unaltered content, and any artefact that fails the authenticity or integrity check is turned away before the policy takes effect. A boot manager that imports a policy signed by the trusted configuration key, while refusing one whose bytes were changed after signing and one signed by a key outside its trust store, satisfies the requirement.
- Fail: an artefact with altered content or an untrusted signer is imported or enforced, or the verification happens only after the policy has been imported or enforced. A boot manager that applies a configuration policy and validates the signature afterwards, or that imports a policy whose signature does not match its content, does not meet the requirement.

Assessment evidence: the artefact format specification; test records for the trusted-unaltered, altered-content, and untrusted-signer artefacts; the record confirming verification precedes import or enforcement.

6.6.6 [ACC-BM-AC-006]

Assessment reference: REQ-BM-AC-006.

Assessment objective: verify that the boot manager indicates when it is running with non-default security-critical configuration, through a persistent visual indication or, where logging is supported, through logged events.

Assessment preparation:

- Test environment: the boot manager with a documented user interface or logging capability available.

- Required information: the security-critical configuration items whose non-default state is indicated; the indication used for each (visual, logged, or both) and its form.
- Required tools: a configuration tool; display or output capture for visual indications; a log capture tool for logged-event indications.

Assessment activities:

- Place each security-critical configuration item into its non-default state in turn.
- Where the indication is visual, observe the user interface across the boot manager's execution and confirm the indication is present throughout, not only momentarily.
- Where the indication is a logged event, capture the log and confirm the documented entry is recorded.
- Return the item to its default state and confirm the indication is removed or superseded.

Assessment verdict:

- Pass: each security-critical item in a non-default state produces its declared indication, and returning the item to default clears or supersedes it. A boot manager that holds a persistent on-screen notice while secure boot is disabled, or records a logged event noting a non-default verification policy, satisfies the requirement.
- Fail: a non-default item produces no indication, or a visual indication declared as persistent appears only briefly. A boot manager that shows a warning once at startup when verification is disabled and then nothing for the rest of its execution does not meet the requirement.

Assessment evidence: the list of security-critical items and their indications; records of visual indication capture across execution; records of logged-event capture.

6.6.7 [ACC-BM-AC-007]

Assessment reference: REQ-BM-AC-007.

Assessment objective: verify that the boot manager protects trusted certificate stores from unauthorised modification.

Assessment preparation:

- Test environment: the boot manager with the trusted certificate store populated and accessible through its documented interfaces.
- Required information: the trusted certificate store, its storage location, the protection mechanism (for example access control, integrity protection, or hardware-backed storage), and the authorised modification path.
- Required tools: a certificate-store inspection tool; modification attempt tools for each interface that exposes the store.

Assessment activities:

- Attempt to modify the trusted certificate store through each documented interface without following the authorised modification path (for example without physical presence, a signed artefact, or authentication, as applicable); confirm each attempt is rejected.
- Attempt to modify the store through any undocumented interface that exists (for example a direct memory access path or a debug interface); confirm rejection.
- Where inspection access allows, alter the store directly in its storage location and confirm that on the next use the integrity check detects the alteration before the store is relied upon.

Assessment verdict:

- Pass: the certificate store cannot be modified through any interface without the authorised path, and a direct alteration of its stored content is caught by an integrity check before the store is used. A boot manager that refuses a store write offered over its configuration interface without authorisation, blocks the same write over a debug interface, and on the next boot detects that the store was patched directly in flash, satisfies the requirement.
- Fail: the store can be modified through some interface without the authorised path, or a direct alteration goes undetected and the tampered store is used. A boot manager that gates its configuration interface but leaves a debug interface able to overwrite the store, or that relies on a directly patched store without checking its integrity, does not meet the requirement.

Assessment evidence: the certificate store specification; records of modification attempts per interface; the record of detection of a direct-storage alteration.

6.6.8 [ACC-BM-AC-008]

Assessment reference: REQ-BM-AC-008.

Assessment objective: verify that the boot manager is accompanied by documentation declaring its code execution authority model with the elements required by REQ-BM-AC-008.

Assessment preparation:

- Test environment: not applicable; this assessment is a documentation review, taking the boot manager's behaviour observed under the other assessments in clause 6.6 as a consistency reference.
- Required information: the documentation accompanying the boot manager that declares the code execution authority model.

Assessment activities:

- Confirm the documentation declares:
  a) the mechanisms and entities that constitute the code execution authority model;
  b) their scope and relationships;
  c) the default state as placed on the market;
  d) any external dependencies of the mechanisms; and
  e) the processes by which the model can be changed where any aspect is mutable.
- Where the documentation declares that no entity or mechanism has authority to decide which code may be executed, confirm the declaration is explicit (see note under REQ-BM-AC-008).
- Confirm the declared model does not contradict the boot manager's behaviour observed under the other assessments in this clause.

Assessment verdict:

- Pass: the documentation declares all five elements, or states explicitly that no entity or mechanism decides which code may run, and the declaration is consistent with the boot manager's observed behaviour. Documentation stating that verified boot with a pre-installed key is enabled at market placement, that the integrator may enrol an additional key through a defined process, and that verification depends on a platform security component, declares a usable model.
- Fail: an element is missing, stated too vaguely to convey the model, or contradicts the boot manager's behaviour. Documentation that sets out a key hierarchy but never states the default state at market placement, or omits an external dependency the boot manager in fact relies on, does not meet the requirement.

Assessment evidence: the documentation under review; a record identifying each declared element by its location in the documentation; a note of any contradiction with observed behaviour.

6.6.9 [ACC-BM-AC-009]

Assessment reference: REQ-BM-AC-009.

Assessment objective: verify that, for each authority transfer mechanism the boot manager provides, the accompanying documentation declares the entities permitted to initiate or approve the transfer, the components covered, and the permanence of the transfer.

Assessment preparation:

- Where the boot manager provides no authority transfer mechanism, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Enumerate the authority transfer mechanisms declared in the documentation.
- For each mechanism, confirm the documentation declares:
  a) the entities that can initiate or approve the transfer;
  b) the components covered by the transfer; and
  c) whether the transfer is permanent or reversible.
- Confirm that no authority transfer mechanism observed under the other assessments in this clause is absent from the documentation.

Assessment activities:

- Enumerate the authority transfer mechanisms declared in the documentation.
- For each mechanism, confirm the documentation declares: (a) the entities that can initiate or approve the transfer; (b) the components covered by the transfer; and (c) whether the transfer is permanent or reversible.

Assessment verdict:

- Pass: every authority transfer mechanism the boot manager provides is documented with its initiating or approving entities, the components it covers, and its permanence, and no provided mechanism is left undeclared. Documentation that, for a key-enrolment transfer, names the integrator as the approving entity, states that it covers the platform verification key, and that the transfer is reversible by a later enrolment, declares a usable mechanism.
- Fail: a declared mechanism is missing one of the three elements, or a mechanism the boot manager actually provides is not declared at all. Documentation that describes who may enrol a key but never states whether the enrolment is permanent, or that omits a recovery-mode transfer the boot manager in fact supports, does not meet the requirement.

Assessment evidence: the documentation under review; a record mapping each declared mechanism to its three declared elements; a note of any provided mechanism found undeclared.

6.6.10 [ACC-BM-AC-010]

Assessment reference: REQ-BM-AC-010.

Assessment objective: verify that the documentation declares how the integrator or a downstream party can verify whether a given piece of software is authorised to run on the boot manager, including the tools, public keys, or mechanisms required.

Assessment preparation:

- Test environment: the boot manager with its configured authority model in place; a sample of software signed by a declared trusted signer and a sample signed by an untrusted signer.
- Required information: the documentation describing the verification procedure, including the tools, public keys, or mechanisms it requires.

Assessment activities:

- Confirm the documentation sets out the verification procedure from input to result, naming the tools and keys needed and the expected outcome for an authorised and for an unauthorised piece of software.
- Follow the documented procedure on the software signed by the declared trusted signer; confirm it reports the software as authorised.
- Follow the documented procedure on the software signed by the untrusted signer; confirm it reports the software as not authorised.

Assessment verdict:

- Pass: the documented procedure can be followed to completion using only what it names, and it reports the trusted-signer sample as authorised and the untrusted-signer sample as not authorised. Documentation that gives the verification tool, the public key to check against, and the commands to run, so that an assessor reproduces both results without further information, satisfies the requirement.
- Fail: the procedure cannot be completed as written, needs information it does not provide, or reports a result that does not match the sample. Documentation that refers to a verification tool without saying where the trusted public key comes from, so the procedure cannot be run, does not meet the requirement.

Assessment evidence: the documentation under review; a record of executing the procedure, with the inputs, tools used, and outputs for both samples.

6.6.11 [ACC-BM-AC-011]

Assessment reference: REQ-BM-AC-011.

Assessment objective: verify that any authority transfer initiated through a mechanism the boot manager provides requires explicit authorisation by the entity declared in the authority model as competent to authorise it, and is not performed as part of an automatic update or as a condition for continued operation.

Assessment preparation:

- Test environment: the boot manager with its configured authority model in place; access to each authority transfer mechanism declared under REQ-BM-AC-009; the means to drive an automatic update cycle.
- Required information: the documentation of the authority transfer mechanisms; the declared identity of the entity competent to authorise each transfer.
- Required tools: a signed-artefact generation tool for the automatic update path; a means of asserting explicit authorisation as documented (for example physical presence, a signed transfer artefact, or console interaction).

Assessment activities:

- Where the boot manager provides no authority transfer mechanism, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each declared mechanism, drive an automatic update without explicit authorisation by the competent entity; confirm the authority transfer is not performed.
- Repeat with the boot manager configured so that continued operation would depend on the transfer; confirm the transfer is still refused without explicit authorisation.
- For each declared mechanism, perform the transfer with explicit authorisation by the competent entity as documented; confirm the transfer is performed.

Assessment verdict:

- Pass: an authority transfer takes place only when the competent entity explicitly authorises it, and neither an automatic update nor a dependency on continued operation brings one about. A boot manager that enrols a new platform key only after the integrator authorises the enrolment at the console, while ignoring a key-enrolment instruction carried in a routine update package, satisfies the requirement.

- Fail: a transfer occurs through an automatic update, as a precondition for the device to keep running, or otherwise without explicit authorisation by the competent entity. A boot manager that silently enrols a key delivered in an update, or that forces a trust-anchor change before it will continue to boot, does not meet the requirement.

Assessment evidence: test records for each declared authority transfer mechanism, covering the automatic-update path, the continued-operation path, and the explicit-authorisation path.

6.7 Confidentiality protection

6.7.0 Overview of requirements addressed in clause 6.7

The present clause specifies assessment criteria for the requirements in clause 5.7.

Table 6.7.0-1: Requirements addressed in clause 6.7

Requirement	Profile applicability
REQ-BM-CP-001	All (if cryptographic keys used)
REQ-BM-CP-002	MEDIUM, HIGH (LOW if verified boot implemented)
REQ-BM-CP-003	MEDIUM, HIGH (LOW if verified/measured boot implemented)
REQ-BM-CP-004	MEDIUM, HIGH (if logging implemented)
REQ-BM-CP-005	All (if configuration and recovery implemented)
REQ-BM-CP-006	MEDIUM, HIGH (if network boot implemented)
REQ-BM-CP-007	MEDIUM, HIGH (LOW if verified/measured boot implemented)
REQ-BM-CP-008	All (if secure disposal implemented)
REQ-BM-CP-009	MEDIUM, HIGH (if network boot implemented)
REQ-BM-CP-010	All

6.7.1 [ACC-BM-CP-001]

Assessment reference: REQ-BM-CP-001.

Assessment objective: verify that access to cryptographic key material is restricted to authorised boot components as defined by the boot manager's security policy.

Assessment preparation:

- Test environment: the boot manager with cryptographic keys provisioned and the security policy active.
- Required information: the security policy identifying key material and the authorised boot components for each; the access-control mechanism used (e.g. isolation boundaries, access tokens, protected storage with enforced access).
- Required tools: inspection tool for memory regions and access-control state on the target platform; test harness capable of invoking non-authorised components that attempt key access.

Assessment activities:

- For each key item, identify the authorised component set per the security policy.
- Attempt access to the key from each authorised component and confirm acceptance; attempt access from a non-authorised component and confirm rejection.
- Inspect memory regions and access-control state to confirm that key material is not reachable from components outside the documented authorised set, including via shared resources (e.g. caches, DMA paths).
- Where the boot manager does not use cryptographic key material, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.

Assessment verdict:

- Pass: each item of key material is accessible to the boot components named as authorised for it in the security policy and to no others; access attempts from components outside the authorised set are refused; and inspection finds no path, direct or via shared caches, DMA, or overlapping memory mappings, by which an unauthorised component could read the key. For example, a key held in a hardware key slot that is released only to the verified-boot stage and locked before control passes onward satisfies the requirement.
- Fail: a component outside the authorised set reads the key directly (e.g. a diagnostic or recovery module with unrestricted memory access); the key remains in a shared buffer or a DMA-reachable region after the authorised stage completes, so a later unauthorised stage can recover it; access is gated only by software convention (naming, documentation) with no enforced isolation or access-control boundary.

Assessment evidence: mapping of keys to components per security policy; records of access tests per component; record of memory and access-control inspection.

6.7.2 [ACC-BM-CP-002]

Assessment reference: REQ-BM-CP-002.

Assessment objective: verify that symmetric verified-boot operations use device-specific keys and that global secret keys are not used for verified-boot purposes.

Assessment preparation:

- Test environment: the boot manager configured for verified boot using symmetric cryptography.
- Required information: key-provisioning documentation identifying the derivation or generation path for each symmetric verified-boot key and demonstrating its device-specific nature (e.g. derivation from a per-device hardware secret, per-device factory provisioning).
- Required tools: tooling to extract or compare key values across two distinct devices of the same model; key-provenance inspection where manufacturer documentation does not suffice.

Assessment activities:

- Where the boot manager does not perform verified or measured boot, or uses no cryptographic key material, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.
- For each symmetric key used in verified boot, inspect the documented derivation or provisioning path and confirm it yields per-device unique values.
- Compare the symmetric verified-boot keys across two distinct devices of the same model; confirm non-equality.
- For a key whose designation excludes a given operation, invoke the boot manager along a path that would require that excluded use (e.g. presenting data that could only be processed by using a verification-only key for decryption); confirm the boot manager does not perform the operation with that key.

Assessment verdict:

- Pass: every symmetric key used in verified boot is unique to the device, established both by the documented derivation or provisioning path and by direct comparison across the tested devices. For example, each device deriving its verified-boot key from a per-device hardware secret fused at manufacture, with the compared devices yielding different key values, satisfies the requirement.
- Fail: the requirement is not met if the same symmetric verified-boot key value appears on two or more distinct devices of the same model; if the documented provisioning path is global or batch-wide, so keys cannot be per-device unique regardless of comparison results; or where the provisioning path is undocumented and device-uniqueness cannot be established by comparison.

Assessment evidence: documentation of key provisioning; record of key comparison across devices; record of provenance inspection where used.

6.7.3 [ACC-BM-CP-003]

Assessment reference: REQ-BM-CP-003.

Assessment objective: verify that key material is not used for cryptographic purposes other than those for which it was designated.

Assessment preparation:

- Test environment: the boot manager with cryptographic operations observable.
- Required information: key-designation table listing each key and the set of cryptographic purposes for which it is authorised (e.g. this key for signature verification only; that key for measurement attestation only).
- Required tools: inspection tool for the cryptographic operations performed by the boot manager, including the key used in each operation.

Assessment activities:

- Where the boot manager does not perform verified or measured boot, or uses no cryptographic key material, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.
- For each key, observe the cryptographic operations performed during a representative boot and record which key is used for which purpose.
- Compare observed use against the key-designation table; flag any use of a key outside its designated purpose set.
- For a key whose designation excludes a given operation, invoke the boot manager along a path that would require that excluded use (e.g. presenting data that could only be processed by using a verification-only key for decryption); confirm the boot manager does not perform the operation with that key.

Assessment verdict:

- Pass: every cryptographic operation observed during boot uses a key drawn from its designated purpose set, and attempts to exercise a key outside its designation are not carried out. For example, a key designated for signature verification is observed only in verification operations, is never used to decrypt or to derive other keys, and a path that would require such off-designation use is e-fused.
- Fail: the requirement is not met if an observed operation uses a key for a purpose outside its designated set, such as a verification key reused for key wrapping or decryption; if the boot manager carries out an operation along a path that requires off-designation key use; or where one key serves multiple unrelated purposes with no designation recorded, so confinement of each key to its designated purpose cannot be established.

Assessment evidence: table of key designations; record of observed operations; records of misuse-attempt tests.

6.7.4 [ACC-BM-CP-004]

Assessment reference: REQ-BM-CP-004.

Assessment objective: verify that sensitive data is not included in crash dumps or log output.

Assessment preparation:

- Test environment: the boot manager in a state where crash handling and log emission can be exercised; the sensitive-data categories documented.
- Required information: enumeration of sensitive data categories; the crash-dump and log-emission paths.
- Required tools: fault-injection tool capable of inducing each documented crash condition; log and crash-dump capture tool.

- Search both capture sets for each sensitive-data category, covering at minimum the categories in the requirement note (cryptographic key material, authentication credentials, security-critical configuration parameters, and personal or otherwise confidential information processed during boot) together with any further categories declared for the product; confirm no such data appears.

Assessment activities:

- Where the boot manager has no logging capability, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.
- Induce each documented crash condition and capture the resulting crash dump.
- Generate log output under nominal and error conditions and capture the log content.
- Search both capture sets for each sensitive-data category; confirm no sensitive data appears.

Assessment verdict:

- Pass: across every induced crash condition and every nominal and error log path, no data from any category in scope appears in crash dumps or log output. For example, a boot manager that logs a verification failure as an event code and component identifier, with the failing key and credential values absent from both the log and any crash dump, satisfies the requirement.
- Fail: the requirement is not met if a value from any in-scope category appears in a crash dump or log entry, such as a key, password, or recovery secret written verbatim on an error path; if a crash dump captures a memory region holding sensitive data without redaction; or where the declared category list omits a category named in the requirement note, so the search cannot establish that all required categories are excluded.

Assessment evidence: list of sensitive data categories; crash-dump captures; log captures; search results per category.

6.7.5 [ACC-BM-CP-005]

Assessment reference: REQ-BM-CP-005.

Assessment objective: verify that stored credentials are protected using approved mechanisms that prevent recovery of the original credential values.

Assessment preparation:

- Test environment: the boot manager with credentials provisioned and the credential storage inspectable.
- Required information: the manufacturer's documentation of the credential-storage mechanism (e.g. salted hash function, hardware-protected key-wrap) and the claim that original credentials cannot be recovered; reference to Annex K for approved mechanisms.
- Required tools: credential-storage inspection tool; cryptanalysis or recovery-attempt tooling appropriate to the declared mechanism.

Assessment activities:

- Where the boot manager stores no credentials, or has no configuration and recovery capability, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.
- Inspect the credential-storage format and confirm it matches the declared mechanism and that the mechanism is listed in Annex K.
- Confirm the stored form does not contain the original credential in recoverable form (e.g. plaintext, reversible encoding).
- Where applicable, attempt offline credential recovery against the stored form using tools proportionate to the declared mechanism's attack surface; confirm recovery is not feasible within the declared strength.

  Where a stored value is a tokenised reference rather than the credential itself, confirm from the documentation that the value is a token and exclude it from the recovery-resistance assessment; the protection obligation applies to the credential values, not to tokens that do not reveal them.

Assessment verdict:

- Pass: every stored credential is held through an Annex K approved mechanism whose stored form does not reveal the original value, and recovery attempts proportionate to the declared strength do not yield it; values identified as tokens are outside this obligation. For example, user passwords stored as salted hashes using an Annex K key-derivation function, where the stored digests cannot be reversed within the declared work factor, satisfy the requirement.
- Fail: the requirement is not met if a credential is stored in plaintext or in a reversible encoding; if the storage mechanism is not listed in Annex K; if offline recovery against the stored form succeeds within the declared strength; or where a value is claimed as a token to avoid protection but in fact reveals or reconstructs the underlying credential.

Assessment evidence: specification of credential storage; record of storage-form inspection; record of recovery-attempt tests where applicable.

6.7.6 [ACC-BM-CP-006]

Assessment reference: REQ-BM-CP-006.

Assessment objective: verify that network boot parameters containing authentication information are protected by approved mechanisms preventing unauthorised disclosure.

Assessment preparation:

- Test environment: the boot manager with network boot configured and the network-boot parameter storage and transmission paths accessible.
- Required information: the identification of network-boot parameters that contain authentication information; the applied protection mechanism for each (at rest, in transit); Annex K reference for approved mechanisms.
- Required tools: parameter-storage inspection tool; network traffic analyser capable of inspecting transmitted parameters.

Assessment activities:

- Where the boot manager stores no credentials, or has no configuration and recovery capability, record the requirement as not applicable per clause 5.1.3 and terminate this assessment.
- For each identified parameter, inspect its storage form and confirm it is protected using the declared mechanism listed in Annex K.
- Capture transmissions containing the parameters and confirm the transmitted form is protected (e.g. encrypted transport, encrypted payload).
- Attempt to read the parameter in plaintext from storage and from the transmission; confirm neither yields the authentication information.

Assessment verdict:

- Pass: every network-boot parameter carrying authentication information is protected by an Annex K approved mechanism both where it is stored and where it is transmitted, and neither storage inspection nor traffic capture yields the authentication information in readable form. For example, a network-boot credential held in sealed storage and carried only inside an encrypted transport session, where inspection of the stored form and of the captured traffic both yield only ciphertext, satisfies the requirement.
- Fail: the requirement is not met if an authentication parameter is readable in plaintext from storage or from captured transmission; if the protection mechanism applied to it is not listed in Annex K; or where the parameter is protected in only one location, such as encrypted in transit but stored in the clear, so disclosure remains possible at the unprotected point.

Assessment evidence: record of parameter identification; record of storage inspection; record of transmission capture.

6.7.7 [ACC-BM-CP-007]

Assessment reference: REQ-BM-CP-007.

Assessment objective: verify that secure disposal cryptographically erases sensitive data and clears security-critical configuration, where technically feasible.

Assessment preparation:

- Test environment: the boot provisioned with sensitive data and security-critical configuration, with the secure-disposal operation accessible.
- Required information: the specification of the configuration for sensitive data and security critical items subject to secure disposal; the cryptographic erase mechanism; the documented limitations od technical feasibility and the rationale for each.
- Required tools: storage inspection tool for sensitive data and configuration regions after disposal.

Assessment activities:

- Where the boot manager does not perform secure disposal, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Invoke the secure-disposal operation with representative sensitive data and security-critical configuration populated.
- Inspect the post-disposal storage for each enumerated item; confirm cryptographic erasure or clearance per the declared mechanism.
- For items documented as outside technical feasibility, confirm the rationale is recorded and the corresponding residual-risk entry exists in the product technical documentation.

Assessment verdict:

- Pass: every security-critical configuration and sensitive data item subject to secure disposal is cryptographically erased or cleared after the operation, and each item excluded on grounds of technical feasibility carries a documented rationale and a corresponding entry for residual risk in the product technical documentation. For example, a boot manager that destroys the key-encryption key sealing all stored secrets, rendering them unrecoverable, and records the contents of a one-time-programmable fuse as a documented feasibility exclusion with its residual risk, satisfies the requirement.
- Fail: the requirement is not met if an item subject to secure disposal retains recoverable content after the operation; if security-critical configuration survives disposal; or where an item is excluded from erasure without a documented feasibility rationale and entry for residual risk.
- Pass: every item subject to secure disposal is cryptographically erased or cleared; any items excluded by technical feasibility are documented with rationale and residual-risk entry.
- Fail: an item subject to secure disposal retains recoverable content after the operation, or an excluded item lacks documented rationale.

Assessment evidence: inventory of sensitive data and configuration; records of inspection after disposal; documentation of technical feasibility and residual risk.

6.7.8 [ACC-BM-CP-008]

Assessment reference: REQ-BM-CP-008.

Assessment objective: verify that the boot manager, where secure disposal is supported, indicates successful completion of sanitisation.

Assessment preparation:

- Test environment: the boot manager with secure disposal supported and the completion-indication channel accessible.
- Required information: the declaration of whether secure disposal is supported; where supported, the form and channel of the completion indication.
- Required tools: observation tool for the completion-indication channel (visual, log output, programmatic response).

Assessment activities:

- Where the boot manager does not support secure disposal, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Where supported, invoke secure disposal and observe the documented completion-indication channel; confirm the indication is presented upon successful completion.
- Abort or interrupt a secure-disposal operation and confirm that the success indication is not presented.

Assessment verdict:

- Pass: a successfully completed sanitisation produces the documented completion indication on its declared channel, and an interrupted or failed operation does not. For example, a boot manager that emits a signed completion record to its audit log only after sanitisation finishes, and emits nothing when the operation is aborted partway, satisfies the requirement.
- Fail: the requirement is not met if a successful sanitisation produces no completion indication; if the indication appears on a channel other than the one documented; or where the success indication is presented for an operation that was interrupted or did not complete.

Assessment evidence: declaration of support for secure disposal; record of indication on successful completion; record indication when the operation is interrupted.

6.7.9 [ACC-BM-CP-009]

Assessment reference: REQ-BM-CP-009.

Assessment objective: verify that boot configuration data transmitted over the network is protected for confidentiality and integrity.

Assessment preparation:

- Confirm that authentication of the network boot server and of the origin of network responses is assessed under clause 6.8 (ACC-BM-INT-013, ACC-BM-INT-014, ACC-BM-INT-015); this assessment addresses the confidentiality and integrity of the transmitted boot configuration data payload itself.
- Test environment: the boot manager configured for network boot in a controlled environment supporting active man-in-the-middle modifications.
- Required information: the documentation of the transport protection mechanism applied to boot configuration data (e.g. authenticated encryption, TLS, payload that is both signed and encrypted); Annex K reference.
- Required tools: passive network capture tool; active man-in-the-middle tool capable of modifying transmitted data.

Assessment activities:

- Where the boot manager has no network boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Capture network transmissions of boot configuration data and confirm the payload is not in plaintext form and is protected by the declared mechanism listed in Annex K.

- Execute a man-in-the-middle modification of a boot configuration data payload; confirm the boot manager detects the integrity violation and rejects the modified data.
- Attempt a passive decryption of the captured payload using tools proportionate to the declared mechanism's strength; confirm infeasibility.

Assessment verdict:

- Pass: boot configuration data transmitted over the network is carried in a form that an Annex K mechanism protects for both confidentiality and integrity, a modification injected in transit is detected and the data rejected, and passive decryption of the captured payload is infeasible within the declared strength. For example, boot configuration delivered inside an authenticated encryption transport session, where the captured payload is ciphertext, a tampered byte causes the boot manager to reject the data, and offline decryption of the capture does not succeed, satisfies the requirement.
- Fail: the requirement is not met if a boot configuration payload is transmitted in plaintext or in a form not protected by an Annex K mechanism; if a payload modified in transit is accepted rather than detected and rejected; or where passive decryption of the captured payload succeeds within the declared strength.

Assessment evidence: specification of transport protection; record of captured payload; record of integrity-violation detection.

6.7.10 [ACC-BM-CP-010]

Assessment reference: REQ-BM-CP-010.

Assessment objective: verify that the boot manager overwrites confidential or secret data after use, clears credentials and temporary data structures before boot target handoff, and does not persist authentication credentials or confidential cryptographic material beyond the operating system handoff, with the exception of measurements and credentials used for attestation.

Assessment preparation:

- Test environment: the boot manager with memory inspection capability at handoff and across power-cycle boundaries.
- Required information: enumeration of confidential or secret data, credentials, and temporary data structures processed during boot; the documented attestation material exempt from clearance; the clearing mechanism for each item.
- Required tools: memory inspection tool at handoff; persistent-storage inspection tool across power cycles.

Assessment activities:

- Drive the boot manager through a complete boot with enumerated sensitive items exercised.
- Immediately before handoff, inspect memory and relevant data structures for residual sensitive content; confirm overwriting or clearance for all enumerated items except the documented attestation exceptions.
- Power-cycle the device and inspect persistent storage for authentication credentials or confidential cryptographic material; confirm no persistence beyond the current boot cycle except for documented attestation material.

Assessment verdict:

- Pass: at boot target handoff every enumerated confidential or sensitive item is overwritten or cleared except the documented attestation material, and across a power cycle no authentication credential or confidential cryptographic material persists beyond the documented attestation exceptions. For example, a boot manager that zeroises the disk-encryption key and the user-entered passphrase from memory immediately before handing control to the operating system, while leaving the attestation quote in its designated register, satisfies the requirement.

- Fail: the requirement is not met if a sensitive item remains readable in memory at handoff without a documented attestation exception; if an authentication credential or confidential cryptographic material persists in storage across a power cycle outside the documented exceptions; or where the attestation carve-out is used to retain material beyond the measurements and attestation credentials the requirement permits.

Assessment evidence: inventory of sensitive items; record of memory inspection at handoff; record of inspection after power cycling.

6.8 Integrity protection

6.8.0 Overview of requirements addressed in clause 6.8

The present clause specifies assessment criteria for the requirements in clause 5.8.

Table 6.8.0-1: Requirements addressed in clause 6.8

Requirement	Profile applicability
REQ-BM-INT-001	All
REQ-BM-INT-002	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-003	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-004	All (if measured boot)
REQ-BM-INT-005	MEDIUM, HIGH
REQ-BM-INT-006	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-007	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-008	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-009	All
REQ-BM-INT-010	MEDIUM, HIGH (if HW supports)
REQ-BM-INT-011	All (if configuration capability)
REQ-BM-INT-012	All (if configuration capability)
REQ-BM-INT-013	MEDIUM, HIGH (if network boot)
REQ-BM-INT-014	MEDIUM, HIGH (if network boot)
REQ-BM-INT-015	MEDIUM, HIGH (if network boot)
REQ-BM-INT-016	MEDIUM, HIGH (if update or configuration capability)
REQ-BM-INT-017	HIGH only
REQ-BM-INT-018	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-019	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-020	All
REQ-BM-INT-021	MEDIUM, HIGH
REQ-BM-INT-022	MEDIUM, HIGH (LOW if implemented)
REQ-BM-INT-023	MEDIUM, HIGH
REQ-BM-INT-024	All (if code execution authority data is stored)

6.8.1 [ACC-BM-INT-001]

Assessment reference: REQ-BM-INT-001.

Assessment objective: verify that the boot manager enforces privilege boundaries between boot stages such that code in one stage cannot obtain privileges assigned to another stage or cross established trust boundaries.

Assessment preparation:

- Test environment: the boot manager at a point where two or more boot stages execute with distinct privileges.
- Required information: the privilege boundaries between stages (memory regions, privileged instructions, security-sensitive hardware access) and the enforcement mechanism for each.
- Required tools: test payload executed within a stage attempting cross-stage privilege access.

Assessment activities:

- For each privilege boundary, run a test payload in the lower-privileged stage that attempts to access the higher-privileged stage's memory, instructions, or hardware resources; confirm each attempt is rejected.
- Attempt an invocation across a trust boundary (for example calling a higher-trust routine with crafted arguments); confirm the enforcement mechanism prevents escalation.

Assessment verdict:

- Pass: code in one boot stage cannot reach the privileges of another, and crossing a trust boundary does not escalate privilege. For example, a payload in a later, lower-privileged stage that tries to read an earlier stage's key material, execute an instruction reserved to a higher-privileged stage, or call a higher-trust routine with crafted arguments is faulted or denied, and every documented boundary holds.
- Fail: the requirement is not met if a payload in one stage reads or writes memory belonging to another stage; if it executes a privileged instruction or reaches a hardware resource reserved to another stage; or where a crafted call across a trust boundary escalates privilege instead of being rejected.

Assessment evidence: the list of privilege boundaries; records of the cross-access tests per boundary.

6.8.2 [ACC-BM-INT-002]

Assessment reference: REQ-BM-INT-002.

Assessment objective: verify that the boot manager verifies integrity and authenticity of each boot stage using approved cryptographic mechanisms and establishes a chain of trust to a root trust anchor before transferring control.

Assessment preparation:

- Test environment: the boot manager with verified boot active and representative boot-stage artefacts.
- Required information: the chain of trust (stages, credentials, verification algorithm per stage) and the identity of the root trust anchor.
- Required tools: signed boot-stage artefacts using trusted and untrusted credentials; a means to observe each verification step.

Assessment activities:

- Where the boot manager does not perform on-boot verification, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Execute a full boot with all stages validly signed; confirm each stage is verified using an approved mechanism per Annex K and that the chain is traced to the root trust anchor before control transfer.
- Substitute one stage's credential with an untrusted credential; confirm the chain-of-trust check detects the break and halts before that stage's control transfer.

Assessment verdict:

- Pass: every stage is verified with an approved mechanism and the chain of trust reaches the root trust anchor before control passes to the next stage, and a broken chain stops the boot. For example, a boot with all stages correctly signed proceeds with each stage validated back to the root anchor, while substituting an untrusted credential at one stage halts the boot before that stage runs.
- Fail: the requirement is not met if a stage executes without being verified; if the chain of trust does not reach the root trust anchor before control transfer; or where a broken chain goes undetected and the affected stage runs.

Assessment evidence: chain-of-trust specification; observation of verification per stage; record of injection of broken chains.

6.8.3 [ACC-BM-INT-003]

Assessment reference: REQ-BM-INT-003.

Assessment objective: verify that the boot manager prevents boot continuation with components that fail integrity or authenticity verification.

Assessment preparation:

- Test environment: the boot manager with verified boot active and the ability to substitute components with documented failure conditions.
- Required information: the verification failure classes (invalid credential, expired credential, revoked credential, tampered component, missing or untrusted verification data).
- Required tools: a component-substitution tool covering each failure class.

Assessment activities:

- Where the boot manager does not perform on-boot verification, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each failure class, substitute a component exhibiting that failure condition and initiate boot.
- Confirm the boot manager halts before executing the failed component, and that no subsequent boot stage executes while the failure condition persists.

Assessment verdict:

- Pass: each failure class stops the boot before the failed component runs, and no later stage executes while the failure persists. For example, a component carrying a revoked credential, and separately one whose contents were tampered after signing, each cause the boot manager to halt before that component is given control.
- Fail: the requirement is not met if a component that fails verification is executed; if a later stage runs after a prior verification failure; or where a failure class produces no halt.

Assessment evidence: the list of failure classes; records of substitution and halt per class.

6.8.4 [ACC-BM-INT-004]

Assessment reference: REQ-BM-INT-004.

Assessment objective: verify that the boot manager computes and records a cryptographic measurement of each boot stage into hardware-protected storage before transferring control, establishing a measurement chain rooted in a hardware-based trust anchor.

Assessment preparation:

- Test environment: the boot manager with measured boot active on a platform with hardware-protected measurement storage (e.g. TPM).
- Required information: the measurement scheme (stages measured, hash algorithm, storage target, hardware root of trust).
- Required tools: a tool to read the hardware-protected measurement storage; an external verifier able to reproduce the expected measurement chain.

Assessment activities:

- Where the boot manager does not implement measured boot, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Execute a boot and read the measurement storage at each transition between stages; confirm each stage's measurement is recorded before control is transferred.

- Reproduce the expected measurement chain with the external verifier from the documented initial values; confirm the recorded chain matches.
- Confirm the root of the measurement chain is the hardware trust anchor.

Assessment verdict:

- Pass: each stage is measured into hardware-protected storage before control passes on, the recorded chain reproduces the expected computation, and it is rooted in the hardware trust anchor. For example, reading the TPM registers after boot shows a measurement recorded for every stage in order, and an external verifier rebuilds the same chain from the documented initial values back to the hardware root.
- Fail: the requirement is not met if a stage is not measured; if a measurement is recorded only after control has transferred; if the recorded chain deviates from the expected computation; or where the chain is not rooted in the hardware trust anchor.

Assessment evidence: measurement-scheme specification; capture of measurements per stage; record of reproducing the expected chain.

6.8.5 [ACC-BM-INT-005]

Assessment reference: REQ-BM-INT-005.

Assessment objective: verify that the boot manager verifies authenticity and integrity of update packages before installation, and additionally after the content is received where updates are written to protected storage

Assessment preparation:

- Test environment: the boot manager in its configuration for receiving and installing updates; where applicable, with protected intermediate storage for updates.
- Required information: the verification algorithm applied to update packages; whether protected intermediate storage is used; the verification points required by the requirement.
- Required tools: a tool to generate update packages with valid and with invalid authenticity or integrity; a means to capture and modify content in transit.

Assessment activities:

- Where the boot manager does not perform on-update verification, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Submit a package with valid authenticity and integrity; confirm verification succeeds and installation proceeds.
- Submit a package altered after signing; confirm verification rejects it before installation.
- Where protected intermediate storage is used, confirm the additional verification after the content is received and before it is committed to that storage; modify the content between receipt and commit and confirm the modification is detected.

Assessment verdict:

- Pass: a package with valid authenticity and integrity is installed, a package altered after signing is rejected before installation, and where protected intermediate storage is used the after-receipt verification catches content modified between receipt and commit. For example, a bit-flipped package is refused before any installation step, and a package modified while staged in protected storage is detected on the second verification rather than being committed.
- Fail: the requirement is not met if an altered package is installed; if verification occurs only after installation has begun; or where, with protected intermediate storage in use, a modification between receipt and commit goes undetected.

Assessment evidence: the verification specification; test records per package; records of the modification tests after receipt.

6.8.6 [ACC-BM-INT-006]

Assessment reference: REQ-BM-INT-006.

Assessment objective: verify that, where hardware security components are unavailable, the boot manager implements software-based cryptographic verification with keys loaded from protected storage.

Assessment preparation:

- Test environment: the boot manager operating on a platform where hardware security components are absent or documented as unavailable.
- Required information: the software-based verification mechanism and the protected-storage location for the keys.
- Required tools: a storage inspection tool; a means to observe the software verification path.

Assessment activities:

- Where the boot manager does not perform verified boot, or hardware security components are available, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Observe that the software-based verification path is used for verified boot and that the keys are retrieved from the documented protected storage.
- Attempt to modify the stored keys through any interface other than the documented authorised path; confirm rejection.

Assessment verdict:

- Pass: with hardware security components unavailable, the boot manager verifies boot in software using keys held in the documented protected storage, and the keys cannot be modified outside the authorised path. For example, the verification routine loads its key from a write-protected region and an attempt to overwrite that key through another interface is refused.
- Fail: the requirement is not met if no software-based verification is performed where hardware components are unavailable; if the keys reside in storage that is not protected; or where the keys can be modified through an interface other than the authorised path.

Assessment evidence: record of platform hardware-component availability; observation of the verification path; inspection of the key storage.

6.8.7 [ACC-BM-INT-007]

Assessment reference: REQ-BM-INT-007.

Assessment objective: verify that the boot manager supports combined verification using more than one approved signing algorithm, where the trust anchor is not immutable.

Assessment preparation:

- Test environment: the boot manager with verified boot active, a mutable trust anchor, and the configuration interface exposed.
- Required information: the set of approved signing algorithms, the interface for configuring it, and the authorisation required to change it.
- Required tools: a configuration tool.

Assessment activities:

- Where the boot manager does not perform verified boot, has no configuration capability, or has an immutable trust anchor, record the requirement as not applicable and close this assessment per clause 5.1.3.

- Inspect the configured set of signing algorithms and confirm each entry is an approved algorithm per Annex K.
- Substitute the configured algorithm through the configuration interface (for example replacing a deprecated algorithm with another approved one) and confirm the change takes effect in the boot manager's verification behaviour.
- Attempt to change the set without the documented authorisation; confirm rejection.

Assessment verdict:

- Pass: the configured set of signing algorithms holds only approved algorithms, the boot manager allows an approved algorithm to be substituted for another through the authorised interface, and an unauthorised change is refused. A configured set of one algorithm is acceptable provided the substitution capability is present. For example, replacing a deprecated approved algorithm with a current approved one through the configuration interface changes which algorithm the boot manager accepts, while the same change attempted without authorisation is rejected.
- Fail: the requirement is not met if the configured set cannot be changed where the trust anchor is mutable; if it holds an algorithm that is not approved under Annex K; if an authorised substitution does not take effect; or where an unauthorised change is accepted.

Assessment evidence: inspection of the configured set; test records of the substitution; records of the unauthorised-change attempts.

6.8.8 [ACC-BM-INT-008]

Assessment reference: REQ-BM-INT-008.

Assessment objective: verify that the boot manager supports combined verification using more than one approved signing algorithm, where the trust anchor is not immutable.

Assessment preparation:

- Test environment: the boot manager with verified boot active and configured for combined verification with more than one approved algorithm.
- Required information: the algorithms used in combination and how a component carries the combined signatures; the rule by which the combination is judged to pass.
- Required tools: boot-stage artefacts carrying the combined signatures, able to render any one of the constituent signatures invalid or absent.

Assessment activities:

- Where the boot manager does not perform verified boot, has no configuration capability, or has an immutable trust anchor, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Present a boot-stage artefact whose combined signatures are all valid; confirm verification succeeds.
- Present an artefact in which one of the combined signatures is invalid or absent while the others are valid; confirm verification fails.

Assessment verdict:

- Pass: the boot manager verifies a component against more than one approved algorithm in combination, accepting it only when every constituent signature is valid. For example, a stage carrying both a classical and a post-quantum signature is accepted when both verify, and is rejected when either one is invalid or missing.
- Fail: the requirement is not met if the boot manager cannot verify against more than one algorithm in combination; or where it accepts a component when one of the combined signatures is invalid or absent, treating the combination as satisfied by a single algorithm.

Assessment evidence: the combined-verification specification; verification records for the all-valid case and for each case with one constituent signature invalid or absent.

6.8.9 [ACC-BM-INT-009]

Assessment reference: REQ-BM-INT-009.

Assessment objective: verify that the boot manager prevents unauthorised modification of a component between its verification and its execution, for the TOCTOU threats within its documented threat model.

Assessment preparation:

- Test environment: the boot manager with inspection access to the memory region holding a verified component between verification and execution.
- Required information: the memory-protection mechanism applied between verification and execution (for example locked page tables, or execute-in-place from write-protected storage); the TOCTOU threats in scope per the security analysis in Annex B. Physical-attack TOCTOU on the boot medium (T-PHYS-5, T-PHYS-6) is assessed under Annex D and is outside this assessment.
- Required tools: a means to attempt modification of the component's memory concurrently with the window between verification and execution.

Assessment activities:

- Where the boot manager has no verified boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- During the window between a component's verification and its execution, attempt to modify the component's memory through each in-scope access path (for example DMA, other processor cores where applicable, peripheral-initiated access).
- Confirm the memory-protection mechanism prevents the modification, or detects it and halts execution, and that it stays in effect for the full window without early release.

Assessment verdict:

- Pass: across the in-scope TOCTOU vectors, a modification attempted between a component's verification and its execution is prevented, or detected and the boot halted, and the protection holds for the whole window. For example, a DMA write into a verified stage's memory after it has been checked but before control passes to it is blocked, or detected and the boot stopped, rather than allowing the altered stage to run.
- Fail: the requirement is not met if a modification during the window succeeds undetected; if the protection mechanism releases before the component executes; or where an in-scope vector is left unprotected.

Assessment evidence: the memory-protection specification; records of the concurrent-modification tests per in-scope vector.

6.8.10 [ACC-BM-INT-010]

Assessment reference: REQ-BM-INT-010.

Assessment objective: verify that, where the hardware platform supports protected memory, the boot manager loads components into protected memory before verification.

Assessment preparation:

- Test environment: the boot manager on a platform with documented protected-memory capability (for example locked memory regions or enclaves).
- Required information: the platform's protected-memory support and the mechanism used.
- Required tools: a tool to identify the protected memory regions.

Assessment activities:

- Where the boot manager has no verified boot capability, or the hardware platform does not support protected memory, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Observe the component-loading sequence and confirm components are placed into the protected region before verification begins.
- Attempt to modify the component content during the verification window through an unprotected access path; confirm the attempt fails because of the protection.

Assessment verdict:

- Pass: on a platform that supports protected memory, each component is placed into the protected region before it is verified, and an attempt to modify it through an unprotected path fails. For example, a stage is loaded into a locked memory region and then verified in place, and a write to that region from outside the protection during the verification window does not take effect.
- Fail: the requirement is not met if a component is verified before being placed into protected memory; or where a modification through an unprotected path succeeds during the verification window.

Assessment evidence: record of the platform's protected-memory capability; observation of the loading sequence; records of the modification attempts.

6.8.11 [ACC-BM-INT-011]

Assessment reference: REQ-BM-INT-011.

Assessment objective: verify that the boot manager protects the integrity and authenticity of sensitive configuration settings.

Assessment preparation:

- Test environment: the boot manager with configuration capability and sensitive configuration items populated.
- Required information: the sensitive configuration settings and the protection mechanism for each (hardware-protected storage, or authenticated encryption with freshness protection and non-deterministic encryption); the cryptographic parameters per Annex K.
- Required tools: a storage inspection tool; a means to tamper with a stored setting; a means to analyse ciphertext for determinism.

Assessment activities:

- Where the boot manager has no configuration capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each sensitive configuration setting, identify the protection mechanism and confirm it matches the declared category.
- Tamper with a protected setting in its stored form and confirm the boot manager detects the alteration and does not act on the tampered value.
- Where hardware-protected storage is used, inspect the storage and confirm the setting is not accessible or modifiable outside the protected path.
- Where authenticated encryption is used, confirm the algorithm is listed in Annex K; confirm that encrypting equal values produces differing ciphertext across events (unique nonces); and confirm freshness by replaying a prior protected value and observing rejection.

Assessment verdict:

- Pass: each sensitive setting is protected by its declared mechanism, a tampered setting is detected and not acted upon, and where authenticated encryption is used the algorithm is from Annex K, nonces are not reused, and a replayed value is rejected. For example, altering a stored verification policy is caught and the boot manager falls back to the protected value rather than the tampered one, and a captured earlier ciphertext for that setting is refused on replay.
- Fail: the requirement is not met if a sensitive setting lacks its declared protection; if a tampered setting is acted upon undetected; if an authenticated-encryption mechanism uses an algorithm outside Annex K, reuses nonces, or accepts a replayed value.

Assessment evidence: mapping of settings to protection mechanisms; records of storage inspection; ciphertext non-determinism test; replay-rejection test.

6.8.12 [ACC-BM-INT-012]

Assessment reference: REQ-BM-INT-012.

Assessment objective: verify that the boot manager restores secure default configuration settings when configuration corruption is detected, where technically feasible.

Assessment preparation:

- Test environment: the boot manager with configuration capability and the ability to inject configuration corruption.
- Required information: the corruption-detection mechanism; the secure default configuration; the documented technical-feasibility limitations (for example restrictions from immutable elements).
- Required tools: a means to inject configuration corruption.

Assessment activities:

- Where the boot manager has no configuration capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Inject corruption into each configurable category and confirm the boot manager detects it. Confirm that, where technically feasible, the boot manager restores the secure default for the affected category.
- For categories documented as outside technical feasibility (for example values held in immutable e-fuses or ROM), confirm the documented handling applies rather than a restoration.

Assessment verdict:

- Pass: corruption is detected in every configurable category, the secure default is restored wherever restoration is technically feasible, and the documented limitation applies to the categories declared infeasible. For example, corrupting a mutable boot-policy setting is detected and the setting reverts to its secure default, while a value fixed in an e-fuse is handled as documented rather than restored.
- Fail: the requirement is not met if corruption goes undetected in a configurable category; if restoration fails in a category documented as feasible; or where a category claimed infeasible has no supporting documentation.

Assessment evidence: records of corruption injection; records of restoration outcome; record of feasibility documentation.

6.8.13 [ACC-BM-INT-013]

Assessment reference: REQ-BM-INT-013.

Assessment objective: verify that the boot manager authenticates network boot servers using cryptographic certificates before accepting actions or configuration data from them.

Assessment preparation:

- Test environment: the boot manager configured for network boot in a controlled environment with distinct authorised and unauthorised server identities.
- Required information: the server authentication mechanism (certificate format, trust roots, verification algorithm) and the actions and configuration data it governs.
- Required tools: a controlled network boot server able to present a valid certificate, an untrusted certificate, or none.

Assessment activities:

- Where the boot manager has no network boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Connect the boot manager to a server presenting a valid trusted certificate; confirm it accepts actions and configuration data from the server.
- Connect to a server presenting no certificate; confirm the boot manager rejects the session before any action is launched or data accepted.
- Connect to a server presenting a certificate from an untrusted root; confirm rejection.

Assessment verdict:

- Pass: the boot manager accepts actions or configuration data only from a server that presents a cryptographically valid certificate chaining to a trusted root, and rejects a server that presents none or one from an untrusted root before anything is accepted. For example, a network boot proceeds from the server with a trusted certificate, while an attempt from a server with no certificate is dropped before any boot action.
- Fail: the requirement is not met if an action is launched or configuration data accepted from a server without a valid trusted certificate; or where authentication occurs only after data has been accepted.

Assessment evidence: the server authentication specification; connection records per scenario.

6.8.14 [ACC-BM-INT-014]

Assessment reference: REQ-BM-INT-014.

Assessment objective: verify that the boot manager rejects network boot connections presenting invalid or revoked server certificates, and rejects expired certificates where a dependable source of time is available.

Assessment preparation:

- Test environment: the boot manager configured for network boot with access to the certificate-validation reference data it uses (CRL, OCSP, or a documented equivalent).
- Required information: the certificate-validity checks applied; the revocation mechanism; whether a dependable source of time is available to the boot manager (see the assumption in Annex B, clause B.3.2).
- Required tools: a controlled server able to present certificates in each failure state (invalid signature, revoked, malformed, not-yet-valid, and expired).

Assessment activities:

- Where the boot manager has no network boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each invalid and revoked certificate state, connect the boot manager to the server and confirm the session is rejected before any network boot action.
- Confirm revocation status is checked through the documented mechanism (for example CRL retrieval, OCSP query, or consultation of a local revocation store); where the boot manager operates offline, confirm the offline revocation mechanism is present.

- Where a dependable source of time is available, present an expired certificate and confirm rejection. Where no dependable source of time is available, confirm expiration is not relied upon and revocation checking still applies.

Assessment verdict:

- Pass: invalid and revoked server certificates are rejected before any network boot action, revocation is checked through the documented mechanism, and an expired certificate is rejected wherever a dependable source of time is available. For example, a connection offering a revoked certificate is dropped after the revocation check, and on a platform with trusted time an expired certificate is likewise refused.

- Fail: the requirement is not met if an invalid or revoked certificate is accepted; if the revocation mechanism is absent or bypassable; or where, with a dependable source of time available, an expired certificate is accepted.

Assessment evidence: the list of certificate failure states; connection records per state; record of exercising the revocation mechanism; record of the time-source condition under which expiration was assessed.

6.8.15 [ACC-BM-INT-015]

Assessment reference: REQ-BM-INT-015.

Assessment objective: verify that the boot manager validates that network configuration and boot-related responses originate from authorised infrastructure.

Assessment preparation:

- Test environment: the boot manager configured for network boot in an environment that can present both authorised and rogue response sources.

- Required information: the response types (DHCP responses, PXE boot offers, network-provided boot configuration data, and similar) and the authorisation mechanism applied to each (for example signed responses, authenticated transport, or trusted source enforcement).

- Required tools: response injection tools representing authorised and rogue sources.

Assessment activities:

- Where the boot manager has no network boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.

- For each response type, present a response from an authorised source and confirm it is accepted.

- Present a response of the same type from a rogue source and confirm it is rejected.

- Where a rogue response can race an authorised one, confirm the boot manager takes the authorised response.

Assessment verdict:

- Pass: for every response type, the boot manager accepts responses from authorised infrastructure and rejects rogue responses, and where the two race it takes the authorised one. For example, a PXE boot offer from the authorised server is acted upon while a rogue PXE offer injected on the same segment is ignored, even when the rogue response arrives first.

- Fail: the requirement is not met if a rogue response is accepted for any response type; or where a race between an authorised and a rogue response results in the rogue one being used.

Assessment evidence: the list of response types; records of authorised and rogue inputs per type; records of the race-condition tests.

6.8.16 [ACC-BM-INT-016]

Assessment reference: REQ-BM-INT-016.

Assessment objective: verify that the boot manager enforces rollback protection when verifying boot stages and configuration data.

Assessment preparation:

- Test environment: the boot manager with update or configuration capability and representative artefacts at the current and at earlier versions.
- Required information: the rollback-protection mechanism (version counter, signed version metadata); where anti-rollback is configurable by the OEM, the documented policy bounding that configuration (see the note in clause 5.3.3).
- Required tools: a tool to generate artefacts at chosen versions.

Assessment activities:

- Where the boot manager has neither update nor configuration capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Present a boot stage or configuration artefact correctly signed but at a version earlier than the current one; confirm the boot manager rejects it.
- Present the current or a newer authorised version; confirm acceptance.
- Where anti-rollback is configurable by the OEM, confirm the configurable path stays within the documented policy and does not disable rollback protection for stages where the policy requires it.

Assessment verdict:

- Pass: an earlier-version artefact is rejected while the current or a newer authorised version is accepted, and any OEM-configurable relaxation stays within the documented policy. For example, a correctly signed boot stage carrying a version below the recorded one is refused, while the current version loads, and an OEM setting cannot switch off rollback protection where the policy mandates it.
- Fail: the requirement is not met if an earlier-version artefact is accepted; if a newer authorised version is rejected; or where OEM configuration disables rollback protection in a case the policy forbids.

Assessment evidence: records of the version tests; record of the OEM policy inspection.

6.8.17 [ACC-BM-INT-017]

Assessment reference: REQ-BM-INT-017.

Assessment objective: verify that, at HIGH security profile, the boot manager stores anti-rollback counters in hardware-backed or tamper-evident storage and verifies signed version metadata before accepting updates.

Assessment preparation:

- Test environment: the boot manager at HIGH security profile with update capability and the anti-rollback counter storage inspectable.
- Required information: the counter storage mechanism (hardware-backed or tamper-evident) and the signed version metadata format.
- Required tools: a counter storage inspection tool; a tool to produce version metadata that is valid, altered, or unsigned.

Assessment activities:

- Where the boot manager is not at HIGH security profile or has no update capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Inspect the counter storage and confirm it is hardware-backed or tamper-evident per the declared mechanism.

- Submit an update with correctly signed version metadata at a version at or above the counter; confirm acceptance.
- Submit an update whose version metadata is altered after signing, and separately one with unsigned metadata; confirm each is rejected before any update action.
- Attempt to decrement the counter through any interface; confirm rejection.

Assessment verdict:

- Pass: the anti-rollback counter is held in hardware-backed or tamper-evident storage, signed version metadata is verified before an update is accepted, altered or unsigned metadata is rejected, and the counter cannot be decremented. For example, an update carrying validly signed metadata at or above the stored counter installs, while one whose metadata was changed after signing is refused and an attempt to roll the counter back through a maintenance interface fails.
- Fail: the requirement is not met if the counter storage is neither hardware-backed nor tamper-evident; if version metadata is not verified, or verified only after the update has begun; if altered or unsigned metadata is accepted; or where the counter can be decremented.

Assessment evidence: inspection of the counter storage; submission records per metadata variant; records of the counter-decrement attempts.

6.8.18 [ACC-BM-INT-018]

Assessment reference: REQ-BM-INT-018.

Assessment objective: verify that the boot manager validates a certificate chain to a trusted root before using a certificate in a security-relevant operation.

Assessment preparation:

- Test environment: the boot manager configured to use certificate chains for a security-relevant operation, with the trusted-root set in place.
- Required information: the trusted-root set; the chain-validation policy (chain from end-entity to root, validity-period checking, revocation, policy constraints); whether a dependable source of time is available (see the assumption in Annex B, clause B.3.2).
- Required tools: a certificate generation tool able to produce chains rooted in trusted and untrusted roots, with and without each validation property.

Assessment activities:

- Where the boot manager does not use certificate chains for security-relevant operations, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Present a certificate with a valid chain to a trusted root, within validity, not revoked, and policy-compliant; confirm acceptance.
- Present certificates each failing one validation property: a chain ending at an untrusted root; a revoked intermediate; a policy violation at some level; confirm rejection in each case.
- Where a dependable source of time is available, present a chain with an expired certificate and confirm rejection; where no dependable source of time is available, confirm validity-period checking is not relied upon while the other checks still apply.

Assessment verdict:

- Pass: only a certificate whose chain reaches a trusted root and meets the applicable validation criteria is accepted for a security-relevant operation, and each failure class is rejected.

- For example, a certificate chaining to a trusted root and passing revocation and policy checks is accepted, while one chaining to an untrusted root, and one whose intermediate is revoked, are each refused; on a platform with trusted time an expired chain is also refused.
- Fail: the requirement is not met if a chain that does not reach a trusted root is accepted; if a revoked or policy-violating chain is accepted; or where, with a dependable source of time available, an expired chain is accepted.

Assessment evidence: the trusted-root set; test records for the valid chain and for each failure class; record of the time-source condition under which validity-period checking was assessed.

6.8.19 [ACC-BM-INT-019]

Assessment reference: REQ-BM-INT-019.

Assessment objective: verify that the boot manager supports revocation of mutable compromised keys and certificates.

Assessment preparation:

- Test environment: the boot manager with verified boot and measured boot supporting remote attestation, and a mutable trust anchor or certificate store.
- Required information: the revocation mechanism (revocation database, CRL, OCSP, authenticated configuration update, or revocation metadata); the revocation-slot constraints.
- Required tools: a tool to generate revocation data.

Assessment activities:

- Where the boot manager does not provide verified boot and measured boot with remote attestation, or its keys and certificates are immutable, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Designate a previously valid key or certificate as compromised and apply revocation through the documented mechanism.
- Attempt to use the revoked key or certificate in a security-relevant operation; confirm it is accepted before revocation and rejected after.
- Confirm the revocation state survives a power cycle and cannot be bypassed by downgrading or replaying the pre-revocation state.

Assessment verdict:

- Pass: applying revocation through the documented mechanism causes a previously valid key or certificate to be rejected, and the revocation state persists across power cycles and resists downgrade or replay, within the documented slot constraints. For example, a key that verified a stage before revocation no longer does so after a revocation entry is applied, and re-presenting the earlier pre-revocation state does not restore acceptance.
- Fail: the requirement is not met if a revoked key or certificate is still accepted after revocation; if the revocation state is lost on power cycle; or where the pre-revocation state can be downgraded or replayed to bypass it.

Assessment evidence: the revocation-mechanism specification; usage records before and after revocation; records of the persistence and bypass attempts.

6.8.20 [ACC-BM-INT-020]

Assessment reference: REQ-BM-INT-020.

Assessment objective: verify that the boot manager uses cryptographic algorithms, key sizes, and parameters in accordance with Annex K.

Assessment preparation:

- Test environment: the boot manager with its cryptographic operations observable.
- Required information: the inventory of cryptographic algorithms, key sizes, and parameters the boot manager uses, with reference to Annex K.
- Required tools: a means to inspect the algorithm, key size, and parameters of each cryptographic operation.

Assessment activities:

- For each cryptographic operation the boot manager performs, observe and record the algorithm, key size, and parameters used.
- Compare each against Annex K and confirm the combination is listed and within the applicable deprecation schedule.

Assessment verdict:

- Pass: every algorithm, key size, and parameter combination the boot manager uses is listed in Annex K and within its deprecation schedule. For example, the signature verification and hashing operations observed during boot all resolve to Annex K entries that have not passed their deprecation date.
- Fail: the requirement is not met if any combination the boot manager uses is absent from Annex K, or has passed its deprecation schedule.

Assessment evidence: the inventory of cryptographic operations with their Annex K cross-reference.

6.8.21 [ACC-BM-INT-021]

Assessment reference: REQ-BM-INT-021.

Assessment objective: verify that the boot manager supports migration between approved cryptographic algorithms.

Assessment preparation:

- Test environment: the boot manager configured to operate with at least one approved cryptographic algorithm and capable of operating with at least one alternative approved algorithm.
- Required information: the algorithm-migration mechanism (configuration metadata, multi-algorithm support, controlled algorithm selection, or boot manager replacement); the approved algorithms supported, with reference to Annex K.
- Required tools: tooling to invoke the migration mechanism as documented (for example a metadata update tool, an algorithm selection interface, or the update mechanism where migration is achieved through boot manager replacement).

Assessment activities:

- Where the boot manager has no update capability or its trust anchor is immutable, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Confirm at least one approved algorithm is in active use as the verification algorithm.
- Through the documented migration mechanism, transition the boot manager from the current approved algorithm to a different approved algorithm from Annex K.
- Confirm that after migration the boot manager verifies under the new algorithm, that artefacts signed under the new algorithm are accepted, and that artefacts signed under the previous algorithm are handled per the documented policy (for example a grace period, immediate rejection, or a documented transition).
- Where concurrent multi-algorithm support is documented, confirm artefacts signed under any active algorithm are verified.

Assessment verdict:

- Pass: the documented migration mechanism moves the boot manager from one approved algorithm to another, verification afterwards operates under the new algorithm, and artefacts under the previous algorithm are handled per the documented transition policy. For example, migrating from one Annex K signature algorithm to another through the documented mechanism causes newly signed artefacts to verify under the new algorithm, while artefacts under the old algorithm follow the documented grace-period or rejection policy.
- Fail: the requirement is not met if no functional migration mechanism exists; if verification does not operate under the new algorithm after migration; or where the handling of artefacts signed under the previous algorithm diverges from the documented policy.

Assessment evidence: specification of the algorithm-migration mechanism; records of verification before and after migration; record of the migration policy; records of multi-algorithm verification where applicable.

6.8.22 [ACC-BM-INT-022]

Assessment reference: REQ-BM-INT-022.

Assessment objective: verify that the boot manager uses collision-resistant and preimage-resistant one-way functions for origin authentication and verification of boot code provenance.

Assessment preparation:

- Test environment: the boot manager with verified boot or measured boot active.
- Required information: the one-way functions used for origin authentication and provenance verification, with reference to Annex K for collision- and preimage-resistance; any function used only for integrity protection in combination with device-specific keys.
- Required tools: a means to inspect the one-way functions invoked during boot.

Assessment activities:

- Where the boot manager has neither verified nor measured boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Observe the one-way functions invoked for origin authentication and provenance verification; confirm each is listed in Annex K and is collision-resistant and preimage-resistant.
- Confirm no such function has an Annex K deprecation applying at the assessment date.
- Where a non-collision-resistant function is used, confirm it is used only for integrity protection in combination with a device-specific key, and not for origin authentication or provenance verification.

Assessment verdict:

- Pass: every one-way function used for origin authentication and provenance verification is collision-resistant, preimage-resistant, Annex K approved, and within its deprecation schedule, and any non-collision-resistant function is confined to integrity protection with a device-specific key. For example, boot-stage provenance is established with a SHA-2 or SHA-3 function from Annex K, while a non-collision-resistant construction such as GHASH appears only in an integrity check bound to a device-specific key.
- Fail: the requirement is not met if a function used for origin authentication or provenance verification is not collision- and preimage-resistant, is absent from Annex K, or has passed its deprecation schedule; or where a non-collision-resistant function is used for origin authentication or provenance verification.

Assessment evidence: the inventory of invoked functions with their Annex K cross-reference and their purpose (origin authentication or provenance verification, or integrity protection with a device-specific key).

6.8.23 [ACC-BM-INT-023]

Assessment reference: REQ-BM-INT-023.

Assessment objective: verify that the boot manager supports replacement of trust anchors used for boot verification.

Assessment preparation:

- Test environment: the boot manager with update capability and a mutable trust anchor store.
- Required information: the trust-anchor replacement mechanism (for example an authenticated update path, a key provisioning interface, or a recovery procedure); the trust anchors and the procedure for replacing each; the authority required to authorise a replacement.
- Required tools: the trust-anchor replacement tool or interface as documented; signing material under both the existing and the replacement trust anchor.

Assessment activities:

- Where the boot manager has no update capability or its trust anchor is immutable, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Confirm the trust anchor is mutable and can be replaced through the documented authorised path.
- Replace the trust anchor with a new anchor and confirm the boot manager then verifies artefacts signed under the new anchor.
- Confirm artefacts signed under the prior anchor are handled per the documented policy after replacement (for example immediate rejection, a documented transition, or a configurable grace period).
- Attempt a trust-anchor replacement through an unauthorised path; confirm rejection.

Assessment verdict:

- Pass: the trust anchor can be replaced through the documented authorised path, verification afterwards operates under the new anchor, artefacts under the prior anchor follow the documented policy, and an unauthorised replacement is refused. For example, replacing the anchor through the authenticated update path causes artefacts signed under the new anchor to verify, while artefacts under the old anchor follow the documented grace-period or rejection policy and an unauthorised replacement attempt is rejected.
- Fail: the requirement is not met if the trust anchor cannot be replaced through the authorised path; if verification does not operate under the new anchor after replacement; if handling of artefacts under the prior anchor diverges from the documented policy; or where an unauthorised replacement is accepted.

Assessment evidence: record of the trust-anchor replacement; verification records before and after replacement; record of the rejection of the unauthorised attempt; documentation of the replacement authority.

6.8.24 [ACC-BM-INT-024]

Assessment reference: REQ-BM-INT-024.

Assessment objective: verify that the storage holding trust anchors, key enrolment records, and policy database contents used by the code execution authority is protected against modification by entities not authorised by the documented authority model.

Assessment preparation:

- Test environment: the boot manager with the code execution authority active, the storage holding authority data open to inspection, and at least one unauthorised entity available to attempt modification.
- Required information:

    a) the storage location for trust anchors, key enrolment records, and policy database contents;

b) the protection mechanism applied to that storage (for example a hardware-protected region, signed configuration, or authenticated update); and

c) the authority model identifying which entities may modify each data category.

- Required tools: a storage-inspection tool appropriate to the storage technology (for example a flash-region reader or a secure-element command interface); a means to attempt modification through each documented and undocumented interface to the storage.

Assessment activities:

- Where the boot manager implements no code execution authority mechanism requiring stored authority data, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Inspect the documented storage and confirm trust anchors, key enrolment records, and policy database contents are held there as described. The protection of any trusted certificate store falling within this storage is assessed under ACC-BM-AC-007 (clause 6.6.7); the present assessment covers the authority data storage more broadly.
- For each unauthorised entity in the authority model, attempt to modify each data category through the documented interfaces; confirm rejection.
- Attempt to modify each data category through undocumented interfaces (direct storage write, debug interfaces where exposed, alternative configuration paths); confirm rejection or detection.
- Where the protection relies on cryptographic verification, confirm a modification with an invalid signature is rejected, and a modification with a valid signature from an unauthorised signer is rejected.

Assessment verdict:

- Pass: the authority data is held only in the documented protected storage, and every attempt to modify it by an entity outside the authority model is rejected or detected, through documented and undocumented interfaces alike. For example, a write to the policy database from an entity the authority model does not permit is refused over the configuration interface and over an exposed debug interface, and a signed change from an unauthorised signer is likewise rejected.
- Fail: the requirement is not met if an unauthorised entity modifies any authority data category through any interface; or where authority data is found held outside the documented protection mechanism.

Assessment evidence: the specification of the authority-data storage and its protection mechanism; the record of authority-model entities; records of the modification attempts per entity per data category, with the outcome of each.

6.9 Data minimisation

6.9.0 Overview of requirements addressed in clause 6.9

The present clause specifies assessment criteria for the requirements in clause 5.9.

Table 6.9.0-1: Requirements addressed in clause 6.9

Requirement	Profile applicability
REQ-BM-DM-001	MEDIUM, HIGH (if implemented)
REQ-BM-DM-002	MEDIUM, HIGH (if network boot)

6.9.1 [ACC-BM-DM-001]

Assessment reference: REQ-BM-DM-001.

Assessment objective: verify that information disclosed by the boot manager to network infrastructure during boot is limited to what is necessary for boot operations.

Assessment preparation:

- Test environment: the boot manager configured for network boot in a controlled network environment permitting passive traffic capture.
- Required information: documentation of network boot protocols used, the list of fields populated by the boot manager in each protocol exchange, and the justification for each field.
- Required tools: network traffic analyser; protocol-aware decoder for the network boot protocols in use (e.g. DHCP, PXE, TFTP, HTTP(S)).

Assessment activities:

- Where the boot manager does not perform network boot, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Capture the complete network traffic generated by the boot manager during a successful network boot, from first transmission to successful handoff.
- Parse each outbound message and enumerate the information fields populated by the boot manager.
- Verify that each populated field is justified by a documented boot operation and that no sensitive or unnecessary information is included (cross-check with ACC-BM-DM-002 for the sensitive-information categories).
- Repeat the capture under documented failure modes (e.g. no response from server, invalid response) and confirm that the information disclosed does not expand under those conditions.

Assessment verdict:

- Pass: every information field the boot manager transmits to network infrastructure is justified by a documented boot operation, and no sensitive or unnecessary information is disclosed under nominal or failure conditions. For example, a network boot that requests only the files required, negotiates only the protocol parameters needed to establish the session, and exposes only identifiers such as the MAC address and device class discloses nothing further when the server fails to respond or returns an invalid response.
- Fail: the requirement is not met if the boot manager transmits a field not justified by a documented boot operation; if it discloses sensitive or unnecessary information; or where the information disclosed expands under a documented failure mode.

Assessment evidence: captured network traces; field-by-field justification table; records of failure-mode capture.

6.9.2 [ACC-BM-DM-002]

Assessment reference: REQ-BM-DM-002.

Assessment objective: verify that the boot manager does not disclose sensitive device information on the network and that temporary network credentials are not retained beyond their required use.

Assessment preparation:

- Test environment: the boot manager in a controlled network environment supporting traffic capture and post-boot inspection of the handoff state.
- Required information: the enumeration of sensitive device information (at minimum: hardware serial numbers, internal configuration, diagnostic data); a statement of which temporary credentials (if any) are used during network boot and their intended lifetime.
- Required tools: network traffic analyser; inspection tool for the boot manager's handoff state (memory, registers, passed structures as applicable to the platform).

Assessment activities:

- Where the boot manager does not perform network boot, record the requirement as not applicable and close this assessment per clause 5.1.3.

- Capture the network traffic generated by the boot manager during a complete network boot.
- Search the captured traffic for instances of the documented sensitive-information categories; confirm none are disclosed.
- Where temporary network credentials (e.g. session tokens, ephemeral keys, bearer tokens) are used, inspect the handoff state after boot completion and confirm that those credentials are neither retained in memory nor passed to the operating system.
- Where network boot is executed under failure modes, confirm that sensitive information is not disclosed as part of error or diagnostic responses.

Assessment verdict:

- Pass: none of the documented sensitive-information categories appears in the captured network traffic under nominal or failure conditions, and any temporary network credentials used during boot are not retained beyond their required use. For example, a complete network boot exposes no hardware serial number, internal configuration, or diagnostic data, and a session token used to fetch the boot image is absent from the handoff state and is not passed to the operating system.
- Fail: the requirement is not met if a documented sensitive-information category appears in network traffic under nominal or failure conditions; if a temporary network credential persists in the handoff state after boot completion; or where such a credential is retained beyond its intended lifetime.

Assessment evidence: list of sensitive-information categories; network trace records with search results; records of state inspection at handoff for temporary credential retention.

6.10 Availability protection

6.10.0 Overview of requirements addressed in clause 6.10

The present clause specifies assessment criteria for the requirements in clause 5.10.

Table 6.10.0-1: Requirements addressed in clause 6.10

Requirement	Profile applicability
REQ-BM-AP-001	All (if configuration capability)
REQ-BM-AP-002	HIGH
REQ-BM-AP-003	All
REQ-BM-AP-004	MEDIUM, HIGH (if verified or network boot)
REQ-BM-AP-005	All
REQ-BM-AP-006	MEDIUM, HIGH
REQ-BM-AP-007	MEDIUM, HIGH (not in immutable memory)
REQ-BM-AP-008	All (if recovery capability)
REQ-BM-AP-009	MEDIUM, HIGH (if network boot)
REQ-BM-AP-010	MEDIUM, HIGH (if verified or measured boot with logging)
REQ-BM-AP-011	All
REQ-BM-AP-012	All (if recovery capability)
REQ-BM-AP-013	MEDIUM, HIGH (if verified boot)
REQ-BM-AP-014	All
REQ-BM-AP-015	All (if recovery capability)
REQ-BM-AP-016	MEDIUM, HIGH (if update and recovery capability)

6.10.1 [ACC-BM-AP-001]

Assessment reference: REQ-BM-AP-001.

Assessment objective: verify that the boot manager supports fallback to a previous known-good configuration.

Assessment preparation:

- Test environment: the boot manager with a current configuration and a documented known-good configuration stored.
- Required information: the specification of the known-good configuration mechanism, the fallback trigger conditions, and the fallback procedure.
- Required tools: configuration-corruption or configuration-replacement tool; fallback-trigger tool.

Assessment activities:

- Where the boot manager has no configuration capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Corrupt or replace the current configuration in a manner documented as triggering fallback.
- Invoke the fallback trigger and confirm the boot manager boots using the known-good configuration.
- Confirm the known-good configuration is not corrupted by the trigger or fallback itself.

Assessment verdict:

- Pass: the boot manager returns to the documented known-good configuration when the documented fallback trigger occurs, and the known-good configuration remains intact after fallback. For example, a boot manager that detects a corrupted active configuration, boots from the stored known-good configuration, and leaves that known-good copy unmodified satisfies the requirement.
- Fail: the requirement is not met if fallback does not occur on the documented trigger; if the boot manager boots from a configuration other than the known-good one; or where the known-good configuration is lost or corrupted by the trigger or the fallback itself.

Assessment evidence: record of the known-good configuration; test records of corruption and fallback.

6.10.2 [ACC-BM-AP-002]

Assessment reference: REQ-BM-AP-002.

Assessment objective: verify that, at HIGH security profile, the boot manager automatically recovers from interruptions during the update process.

Assessment preparation:

- Test environment: the boot manager at HIGH security profile, configured to perform updates.
- Required information: the enumeration of interruption classes during update (power loss, transport interruption, intermediate-stage failure); the documented recovery mechanism for each.
- Required tools: interruption-injection tools for each documented class (e.g. power cut at arbitrary update stages).

Assessment activities:

- Where the boot manager is not at HIGH security profile, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each interruption class, induce the interruption at representative points during an update and confirm the boot manager automatically recovers to either the pre-update or post-update state without manual intervention.
- Confirm the resulting state is consistent (no split state with part of the update applied).
- Confirm recovery does not require operations outside the boot manager's control.

Assessment verdict:

- Pass: the boot manager automatically recovers to a consistent pre-update or post-update state, without manual intervention, for every documented interruption class. For example, a boot manager interrupted by power loss midway through an update that resumes on the next boot to a fully pre-update or fully post-update state, with no operator action, satisfies the requirement.
- Fail: the requirement is not met if an interruption leaves the boot manager in an inconsistent or split state; if recovery requires manual intervention; or where recovery depends on operations outside the boot manager's control.

Assessment evidence: enumeration of interruption classes; injection records per class; records of state inspection after interruption.

6.10.3 [ACC-BM-AP-003]

Assessment reference: REQ-BM-AP-003.

Assessment objective: verify that the boot manager enforces timeouts to prevent indefinite blocking during boot-related operations.

Assessment preparation:

- Test environment: the boot manager with control over the response latency of its inputs (network peers, peripherals, storage).
- Required information: the enumeration of boot-related operations subject to timeout; each timeout value.
- Required tools: response-latency control tool (e.g. slow-responding emulator); time measurement.

Assessment activities:

- For each enumerated operation, delay the input source beyond the documented timeout and measure the boot manager's response.
- Confirm the boot manager terminates the operation at or near the documented timeout and does not block indefinitely.
- Confirm the boot manager's subsequent behaviour is as documented (e.g. fallback, halt, error state).

Assessment verdict:

- Pass: every enumerated boot-related operation is terminated at or near its documented timeout, the boot manager does not block indefinitely on any documented input, and it proceeds afterwards to its documented behaviour. For example, a boot manager awaiting a network response that halts the attempt at the documented timeout and falls back to local boot rather than waiting indefinitely satisfies the requirement.
- Fail: the requirement is not met if an enumerated operation blocks beyond its documented timeout; if any documented input can cause the boot manager to block indefinitely; or where the post-timeout behaviour is inconsistent with the documentation.

Assessment evidence: enumeration of operation timeouts; records of delay and response measurements per operation.

6.10.4 [ACC-BM-AP-004]

Assessment reference: REQ-BM-AP-004.

Assessment objective: verify that the boot manager limits retry attempts and resource consumption for signature verification and network operations.

Assessment preparation:

- Test environment: the boot manager in states exercising signature verification and network operations.

- Required information: the specification of retry-attempt limits and resource-consumption bounds for each operation class.
- Required tools: attempt-stimulation tool (e.g. repeated invalid-signature submissions, repeated network attempts); resource-monitoring tool.

Assessment activities:

- Where the boot manager performs neither verified boot nor network boot, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Repeatedly submit inputs to signature verification and network-operation paths beyond the documented retry limit; confirm the boot manager halts retries at the documented limit.
- Measure resource consumption during sustained retry stimulus; confirm it remains within the documented bounds.
- Confirm that exceeding the retry limit produces the documented terminal behaviour rather than unbounded retrying.

Assessment verdict:

- Pass: the boot manager observes its documented retry limits and resource-consumption bounds under sustained stimulus, and exceeding a retry limit produces the documented terminal behaviour. For example, a boot manager that stops re-attempting signature verification after the documented number of failed attempts, holds memory and time within the documented bounds throughout, and then halts to a documented error state satisfies the requirement.
- Fail: the requirement is not met if retries continue beyond the documented limit; if resource consumption exceeds the documented bounds under sustained stimulus; or where exceeding the limit produces undocumented behaviour instead of the documented terminal state.

Assessment evidence: specification of retry and resource policy; test records with measurements under sustained stimulus.

6.10.5 [ACC-BM-AP-005]

Assessment reference: REQ-BM-AP-005.

Assessment objective: verify that the boot manager prevents unauthorised bypass of security verification steps during normal operation.

Assessment preparation:

- Test environment: the boot manager in normal operational mode; documented debug or in-field modes disabled or appropriately gated.
- Required information: the enumeration of security verification steps; the documented authorisation mechanism for any permitted bypass (e.g. in-field debug mode).
- Required tools: tools to attempt bypass via input manipulation, state manipulation, or invocation of documented bypass paths without authorisation.

Assessment activities:

- For each security verification step, attempt to bypass it through input manipulation and via any documented bypass path without providing the documented authorisation; confirm the step is executed and the bypass is rejected.
- Where a documented bypass path exists (e.g. in-field debug mode), attempt invocation without authentication and confirm rejection.
- Confirm that in normal operational mode no mechanism permits unauthorised bypass.

Assessment verdict:

- Pass: every security verification step executes in normal operation; documented bypass paths require authentication; unauthorised bypass attempts are rejected.
- Fail: a security verification step can be bypassed without authentication, or normal operational mode permits unauthorised bypass.

Assessment evidence: enumeration of verification steps; records of bypass attempts per step; test records of bypass-path authorisation.

6.10.6 [ACC-BM-AP-006]

Assessment reference: REQ-BM-AP-006.

Assessment objective: verify that, at MEDIUM or HIGH security profile, the boot manager detects and corrects errors in critical data using mechanisms supported by the underlying platform.

Assessment preparation:

- Test environment: the boot manager at MEDIUM or HIGH security profile with access to error-inducing channels on the critical-data storage.
- Required information: enumeration of critical data items; the error-detection and correction mechanism used for each (e.g. ECC, FEC, cross-copy comparison); the platform features relied upon.
- Required tools: bit-flip injection tool or equivalent for the platform; correction-observation tool.

Assessment activities:

- Where the boot manager is not at MEDIUM or HIGH security profile, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each critical-data item, inject a documented-recoverable error into the stored value.
- Confirm the boot manager detects the error during its next access.
- Confirm the error is corrected and the data is readable at its expected value.
- Inject a documented non-recoverable error and confirm detection plus documented failure handling.

Assessment verdict:

- Pass: recoverable errors in critical data are detected and corrected, and non-recoverable errors are detected and produce documented failure handling. For example, a boot manager that detects a single-bit error in a stored key, corrects it using the platform error-correction mechanism, and refuses to boot when presented with an uncorrectable error in that key satisfies the requirement.
- Fail: the requirement is not met if an injected recoverable error is not detected or not corrected; if a non-recoverable error is not detected; or where a detected non-recoverable error does not lead to the documented failure handling.

Assessment evidence: enumeration of critical data; records of error injection and correction per item.

6.10.7 [ACC-BM-AP-007]

Assessment reference: REQ-BM-AP-007.

Assessment objective: verify that, at MEDIUM or HIGH security profile where the boot manager is not stored in immutable memory, essential boot code remains available in the presence of storage corruption or failure.

Assessment preparation:

- Test environment: the boot manager at MEDIUM or HIGH security profile on a platform whose boot-code storage is mutable and accessible for corruption injection.
- Required information: identification of essential boot code; the availability mechanism (e.g. redundant storage, fallback partition); the corruption scenarios the mechanism is expected to withstand.
- Required tools: storage-corruption injection tool; boot-execution observation tool.

Assessment activities:

- Where the boot manager is stored in immutable memory, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Corrupt the primary copy of the essential boot code within the documented withstand-scenario; confirm the boot manager continues boot using the redundant copy or fallback partition.
- Corrupt at the boundary of the documented withstand-scenario (e.g. both copies) and confirm the boot manager produces the documented failure indication rather than silently continuing.
- Confirm the availability mechanism itself is protected from unauthorised corruption.

Assessment verdict:

- Pass: essential boot code remains available under the documented withstand-scenarios, the boot manager produces documented failure handling beyond those scenarios, and the availability mechanism is itself protected from unauthorised corruption. For example, a boot manager that boots from a redundant copy when its primary essential-boot-code copy is corrupted, signals a documented failure when both copies are corrupted, and prevents unauthorised modification of the redundant copy satisfies the requirement.
- Fail: the requirement is not met if essential boot code becomes unavailable within a documented withstand-scenario; if corruption beyond that scenario does not produce the documented failure handling; or where the availability mechanism can itself be bypassed or corrupted by an unauthorised entity.

Assessment evidence: identification of essential boot code; test records of corruption injection; record of inspection of the availability mechanism.

6.10.8 [ACC-BM-AP-008]

Assessment reference: REQ-BM-AP-008.

Assessment objective: verify that sensitive actions performed in recovery mode require authentication or physical presence.

Assessment preparation:

- Test environment: the boot manager with recovery capability and recovery mode entry controllable.
- Required information: enumeration of recovery-mode actions classified as sensitive; the authentication or physical-presence requirement for each.
- Required tools: recovery-mode entry tool; authentication and physical-presence stimulation tools.

Assessment activities:

- Where the boot manager has no recovery capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Enter recovery mode by each documented trigger (including automatic triggers on boot failure).
- Attempt each sensitive action without authentication or physical presence; confirm rejection.
- Attempt each sensitive action with the documented authentication or physical-presence assertion; confirm acceptance.

Assessment verdict:

- Pass: every recovery-mode action classified as sensitive requires the documented authentication or physical presence, and attempts to perform such an action without either are rejected. For example, a boot manager that enters recovery mode automatically after repeated boot failures but refuses to erase keys or change the trust anchor until an authenticated operator acts satisfies the requirement.
- Fail: the requirement is not met if a sensitive recovery-mode action can be performed without the documented authentication or physical presence; or where a documented recovery trigger reaches a sensitive action without enforcing the authorisation requirement.

Assessment evidence: enumeration of sensitive actions; authorisation test records per action.

6.10.9 [ACC-BM-AP-009]

Assessment reference: REQ-BM-AP-009.

Assessment objective: verify that the boot manager, where network boot is supported, handles network boot failures in a manner that maintains boot availability.

Assessment preparation:

- Test environment: the boot manager configured for network boot in a controlled environment permitting network-level failure injection.
- Required information: enumeration of network-boot failure classes; the documented handling mechanism for each (e.g. timeout, fallback to local boot, backoff retries, alternative servers).
- Required tools: network failure-injection tool (server unavailable, partial response, malformed response); boot-outcome observation.

Assessment activities:

- Where the boot manager has no network boot capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each documented failure class, induce the failure and observe the boot manager's handling.
- Confirm the documented handling mechanism is applied (timeout fires, fallback is entered, backoff retries proceed, or alternative servers are attempted as documented).
- Confirm the boot manager reaches a documented terminal state (successful boot via alternative path, documented failure indication) rather than hanging indefinitely.

Assessment verdict:

- Pass: each documented network-boot failure class invokes the documented handling mechanism and the boot manager reaches a documented terminal state rather than hanging. For example, a boot manager that, on an unreachable boot server, applies its documented timeout, performs backoff retries, then falls back to local boot satisfies the requirement.
- Fail: the requirement is not met if a documented failure class causes the boot manager to hang or behave in an undocumented way; if the documented handling mechanism is not invoked; or where no documented terminal state is reached.

Assessment evidence: enumeration of failure classes; injection and outcome records per class.

6.10.10 [ACC-BM-AP-010]

Assessment reference: REQ-BM-AP-010.

Assessment objective: verify that the boot manager generates logs for security-relevant failure conditions.

Assessment preparation:

- Test environment: the boot manager with verified or measured boot and logging capability active.
- Required information: the manufacturer's enumeration of security-relevant failure conditions (e.g. verification failures, integrity violations, authentication failures, unexpected modification of boot components); the documented log-entry content for each.
- Required tools: fault-injection tool for each condition; log capture tool.

Assessment activities:

- Where the boot manager has no logging capability, or performs neither verified nor measured boot, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Induce each enumerated security-relevant failure condition.
- Capture the log output and confirm that a log entry consistent with the documented content is generated for each.
- Confirm the entry identifies the condition type and the affected component or step.

Assessment verdict:

- Pass: every enumerated security-relevant failure condition produces a log entry consistent with the documented content, identifying the condition type and the affected component or step. For example, a boot manager that records a distinct, parseable log entry naming the failed component each time signature verification fails satisfies the requirement.
- Fail: the requirement is not met if an enumerated condition produces no log entry; if an entry does not match the documented content; or where an entry fails to identify the condition type or the affected component.

NOTE: Logging of security-relevant failure conditions is also addressed by REQ-BM-MON-003, assessed under clause 6.14. Where both requirements apply, one body of log evidence may serve both this assessment and the assessment of ACC-BM-MON-003.

Assessment evidence: enumeration of failure conditions; records of injection and log capture per condition.

6.10.11 [ACC-BM-AP-011]

Assessment reference: REQ-BM-AP-011.

Assessment objective: verify that the boot manager uses time representations valid beyond the year 2038 for all security-relevant uses of time.

Assessment preparation:

- Test environment: the boot manager with the ability to set internal time or to process inputs containing time values beyond 2038-01-19 03:14:07 UTC.
- Required information: enumeration of security-relevant uses of time (certificate validity, internal system time for boot decisions, timestamps in logs); the representation used for each.
- Required tools: time-manipulation tool to set inputs beyond the 2038 boundary; inspection tool for each time representation.

Assessment activities:

- For each enumerated use, inspect the representation width and confirm it accommodates values beyond the 2038 boundary (e.g. 64-bit timestamp).
- Process inputs containing a time value beyond the 2038 boundary (e.g. a certificate with a post-2038 expiry) and confirm correct handling.

- Set the boot manager's internal time beyond the 2038 boundary where supported and confirm subsequent operations behave correctly.

Assessment verdict:

- Pass: every enumerated security-relevant use of time uses a representation that accommodates values beyond the 2038 boundary, and the boot manager handles post-2038 inputs and internal time correctly. For example, a boot manager that holds certificate validity and log timestamps in a 64-bit representation and correctly accepts a certificate with a post-2038 expiry satisfies the requirement.

- Fail: the requirement is not met if an enumerated representation wraps or truncates at or after the 2038 boundary; if a post-2038 input is processed incorrectly; or where internal time set beyond the boundary produces incorrect behaviour.

Assessment evidence: enumeration of time uses with inspection of representation; records of input processing after 2038.

6.10.12 [ACC-BM-AP-012]

Assessment reference: REQ-BM-AP-012.

Assessment objective: verify that recovery mechanisms are protected from unauthorised modification or disablement.

Assessment preparation:

- Test environment: the boot manager with recovery capability and the recovery-mechanism storage and configuration accessible.

- Required information: specification of the recovery mechanism and the protection applied to its code, data, and enablement state.

- Required tools: modification and disablement attempt tools for each interface exposing the recovery mechanism.

Assessment activities:

- Where the boot manager has no recovery capability, record the requirement as not applicable and close this assessment per clause 5.1.3.

- Attempt modification of the recovery mechanism's code and data through each documented interface without the authorised path; confirm rejection.

- Attempt disablement of the recovery mechanism through each interface without authorised path; confirm rejection.

- Attempt modification via undocumented or residual interfaces; confirm rejection.

Assessment verdict:

- Pass: the recovery mechanism's code, data, and enablement state are protected from modification and disablement through every interface lacking the authorised path. For example, a boot manager that rejects attempts to overwrite its recovery partition or clear its recovery-enabled flag through any interface other than the authenticated update path satisfies the requirement.

- Fail: the requirement is not met if the recovery mechanism's code or data can be modified without the authorised path; if the mechanism can be disabled without the authorised path; or where an undocumented or residual interface permits either.

Assessment evidence: specification of the recovery mechanism; records of modification and disablement attempts per interface.

6.10.13 [ACC-BM-AP-013]

Assessment reference: REQ-BM-AP-013.

Assessment objective: verify that partial execution of boot components is prevented when verification has not completed successfully.

Assessment preparation:

- Test environment: the boot manager with verified boot and the ability to interrupt or fail verification mid-process.
- Required information: specification of the execution-gate mechanism between verification and execution.
- Required tools: verification-fault injection tool; execution observation tool.

Assessment activities:

- Where the boot manager does not perform verified boot, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Induce verification failure at various stages (before completion, at a boundary, immediately prior to execution) and confirm no portion of the component is executed.
- Induce a verification process that does not complete (e.g. crashed verifier) and confirm the boot manager does not proceed to execution.
- Inspect the transition between verification and execution for any race or shortcut path.

Assessment verdict:

- Pass: no portion of a boot component is executed when its verification fails or has not completed, and no race or shortcut path bypasses the gate between verification and execution. For example, a boot manager that holds a component inert until its signature check completes successfully, and that halts rather than executing when the verifier crashes mid-check, satisfies the requirement.
- Fail: the requirement is not met if any portion of a component executes while its verification is incomplete or has failed; or where a race condition or shortcut path allows execution to begin before the verification gate completes.

NOTE: Continuation of boot with components that fail verification is addressed by REQ-BM-INT-003, assessed under clause 6.8; the present requirement addresses partial execution of a component while verification is incomplete.

Assessment evidence: records of verification-failure injection; records of execution observation; record of inspection of the transition path.

6.10.14 [ACC-BM-AP-014]

Assessment reference: REQ-BM-AP-014.

Assessment objective: verify that the boot manager does not perform security-sensitive operations when required cryptographic components or services are unavailable.

Assessment preparation:

- Test environment: the boot manager with the ability to induce unavailability of required cryptographic components or services (e.g. TPM unavailable, cryptographic provider faulted).
- Required information: numeration of security-sensitive operations and the cryptographic components or services each depends upon.
- Required tools: cryptographic-component unavailability injection tool.

Assessment activities:

- For each security-sensitive operation, make the required cryptographic component or service unavailable and attempt the operation.

- Confirm the operation is not performed under the unavailability condition and the boot manager produces documented handling.
- Confirm that no fallback uses a weaker substitute that would circumvent the security intent.

Assessment verdict:

- Pass: no security-sensitive operation is performed while a cryptographic component or service it requires is unavailable, and no weaker substitute is used in place of the required one. For example, a boot manager that declines to verify a signature and halts to a documented error state when the TPM it relies on is unavailable, rather than skipping verification or using an unapproved fallback, satisfies the requirement.
- Fail: the requirement is not met if a security-sensitive operation is performed while its required cryptographic component or service is unavailable; or where a weaker substitute is used that circumvents the security intent.

Assessment evidence: mapping of operations to components; unavailability test records per operation; record of fallback inspection.

6.10.15 [ACC-BM-AP-015]

Assessment reference: REQ-BM-AP-015.

Assessment objective: verify that, where recovery capability exists, the boot manager does not bypass security controls when handling security violations or verification failures.

Assessment preparation:

- Test environment: the boot manager with recovery capability and the ability to induce security violations and verification failures.
- Required information: the manufacturer's enumeration of violation and failure classes; the handling path for each, including the security controls retained.
- Required tools: violation and failure injection tools; inspection tool for the security-control state during and after handling.

Assessment activities:

- Where the boot manager has no recovery capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Induce each enumerated violation or failure class and observe the handling path.
- Confirm that the security controls documented as retained are active during and after handling (e.g. verification is still required before execution; authentication is still required for sensitive actions).
- Confirm that the handling path does not reach a state where security controls are weakened or bypassed.

Assessment verdict:

- Pass: every documented handling path for a security violation or verification failure retains the security controls documented as retained, and no path weakens or bypasses them. For example, a boot manager that, on detecting an integrity violation, enters its handling path while still requiring signature verification before any subsequent execution and authentication before any sensitive action satisfies the requirement.
- Fail: the requirement is not met if a handling path weakens or bypasses a security control documented as retained; or where handling a violation or failure reaches a state in which verification or authentication is no longer enforced.

Assessment evidence: enumeration of violation classes with mapping of retained control; injection records per class with records of control state.

6.10.16 [ACC-BM-AP-016]

Assessment reference: REQ-BM-AP-016.

Assessment objective: verify that recovery capability is preserved across firmware and configuration updates.

Assessment preparation:

- Test environment: the boot manager with recovery and update capabilities in an environment supporting applying representative updates.
- Required information: the manufacturer's specification of the recovery-preservation mechanism across updates; the update scenarios required to preserve recovery capability.
- Required tools: update-application tool; recovery-mechanism exercise tool.

Assessment activities:

- Where the boot manager has no update capability or no recovery capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Apply firmware and configuration updates through the documented update paths.
- After each update, exercise the recovery mechanism and confirm it is functional.
- Include in the test sample a firmware update that alters boot-manager components and a configuration update that alters recovery-related configuration.

Assessment verdict:

- Pass: recovery capability remains functional after every documented class of firmware and configuration update. For example, a boot manager whose recovery mechanism still operates after a firmware update that replaces boot-manager components and after a configuration update that alters recovery-related settings satisfies the requirement.
- Fail: the requirement is not met if recovery capability is broken, disabled, or degraded by any documented update class; or where a documented update scenario leaves the recovery mechanism non-functional.

Assessment evidence: enumeration of update classes; records of recovery exercises per update.

6.11 Non-interference

6.11.0 Overview of requirements addressed in clause 6.11

The present clause specifies assessment criteria for the requirements in clause 5.11.

Table 6.11.0-1: Requirements addressed in clause 6.11

Requirement	Profile applicability
REQ-BM-IM-001	MEDIUM, HIGH (if network boot)

6.11.1 [ACC-BM-IM-001]

Assessment reference: REQ-BM-IM-001.

Assessment objective: verify that the boot manager does not generate harmful or disruptive network behaviour.

Assessment preparation:

- Test environment: the boot manager configured for network boot in a controlled network environment with other hosts and network segments present, permitting passive observation of all traffic originating from the boot manager.

- Required information: the manufacturer's enumeration of network behaviours produced by the boot manager (broadcast scope, multicast participation, repeated retries, peer-discovery patterns) and the documented bounds on each.
- Required tools: network traffic analyser capable of identifying broadcast storms, loop formation, and abnormal traffic rates; topology probe able to detect loop conditions.

Assessment activities:

- Capture the complete network traffic generated by the boot manager across a successful network boot and across each documented failure-mode scenario.
- Analyse the captured traffic for behaviour patterns that would be harmful or disruptive to other hosts or network segments, at minimum: broadcast rate relative to documented bounds; presence of loop-inducing frame forwarding; sustained retry patterns beyond documented bounds; traffic targeted at unintended destinations.
- Exercise the boot manager in a network topology where loops could form if frame-forwarding behaviour were incorrect; confirm no loop is induced.

Assessment verdict:

- Pass: observed network behaviour stays within the documented bounds; no broadcast storm, loop, or sustained disruptive pattern is produced under nominal or failure-mode operation.
- Fail: the boot manager produces a broadcast storm, induces a network loop, or generates sustained traffic beyond documented bounds.

Assessment evidence: documented bounds on network behaviour; traffic captures under nominal conditions and under failure modes; record of loop-topology exercises.

6.12 Attack surface minimisation

6.12.0 Overview of requirements addressed in clause 6.12

The present clause specifies assessment criteria for the requirements in clause 5.12.

Table 6.12.0-1: Requirements addressed in clause 6.12

Requirement	Profile applicability
REQ-BM-AMS-001	All
REQ-BM-AMS-002	All
REQ-BM-AMS-003	All
REQ-BM-AMS-004	All
REQ-BM-AMS-005	All
REQ-BM-AMS-006	All
REQ-BM-AMS-007	All (if configuration capability)

6.12.1 [ACC-BM-AMS-001]

Assessment reference: REQ-BM-AMS-001.

Assessment objective: verify that the boot manager, before handing off control to the boot target, ensures that only the resources necessary for the boot target remain enabled.

Assessment preparation:

- Test environment: the boot manager executed to the handoff point with inspection capability over the platform resource state at that point (peripherals, memory regions, DMA controllers, interrupt sources, CPU modes).

- Required information: the enumeration of resources enabled during boot; for each, whether it is required by the boot target at handoff and the rationale.
- Required tools: platform-state inspection tool capable of enumerating enabled resources at the handoff moment.

Assessment activities:

- Capture the resource state at the handoff point (e.g. by halting immediately prior to handoff, or by instrumenting the handoff transition).
- For each resource enumerated as enabled, confirm it is documented as required by the boot target; investigate any undocumented enabled resource.
- For each resource documented as not required at handoff but enabled during earlier boot stages, confirm it has been disabled, quiesced, or revoked before handoff.

Assessment verdict:

- Pass: only the resources necessary for the boot target remain enabled at the handoff point, and every resource used during earlier boot stages but not required at handoff has been disabled, quiesced, or released. For example, a boot manager that disables its DMA-capable diagnostic interface and releases its network and communication buffers before transferring control, leaving enabled only the memory regions and devices the boot target requires, satisfies the requirement.
- Fail: the requirement is not met if a resource not required by the boot target remains enabled at handoff; if a resource enabled during an earlier boot stage is not disabled, quiesced, or released before handoff; or where an enabled resource at the handoff point has no documented justification.

Assessment evidence: record of resource enumeration with required/not-required classification; record of state capture at handoff.

6.12.2 [ACC-BM-AMS-002]

Assessment reference: REQ-BM-AMS-002.

Assessment objective: verify that non-essential code is excluded from production builds of the boot manager.

Assessment preparation:

- Test environment: the production build of the boot manager as placed on the market.
- Required information: build configuration and documented differences between development, test, and production builds; the list of code paths, modules, or features categorised as non-essential for production (e.g. debug output, test harnesses, development shortcuts, example configurations).
- Required tools: binary analysis tools capable of identifying string constants, code paths, and linked modules; where source access is available, build-manifest inspection.

Assessment activities:

- Inspect the build configuration and confirm that non-essential modules and flags are not enabled for the production build.
- Inspect the binary for indicators of non-essential code (e.g. debug strings, test symbols, unreached branches linked only by non-production flags) and investigate any finding.
- Where source access is available, confirm that guards around non-essential code exclude it under the production build configuration.

Assessment verdict:

- Pass: the production build excludes code documented as non-essential, and no indicator of non-essential code is found in the production binary beyond those documented. For example, a production build whose debug-output module and test harness are excluded by the production build flags, and whose binary shows no debug strings or test symbols beyond the documented set, satisfies the requirement.
- Fail: the requirement is not met if the production build contains modules or code paths documented as non-essential; or where binary analysis surfaces non-essential code not present in the documentation.

Assessment evidence: record of build configuration; findings of binary analysis; review of source guards where applicable.

6.12.3 [ACC-BM-AMS-003]

Assessment reference: REQ-BM-AMS-003.

Assessment objective: verify that interfaces and functions not intended for operational use are disabled or protected.

Assessment preparation:

- Test environment: the boot manager in its operational configuration on a representative platform.
- Required information: enumeration of interfaces and functions exposed by the boot manager, each classified as operational or non-operational; for non-operational items, the applied mechanism (disabled, physically protected, authentication-gated, fuse-locked).
- Required tools: interface enumeration tools for the platform; interaction tools for each interface category (e.g. serial debug, JTAG, USB debug, memory-mapped registers).

Assessment activities:

- Enumerate the interfaces accessible on the operational product and compare against the documented list.
- For each interface classified as non-operational, confirm the documented protection is in effect; test the protection by attempting to exercise the interface without the documented unlock path.
- For each interface classified as operational, confirm it matches the manufacturer's intended purpose and exposes only documented functions.

Assessment verdict:

- Pass: every non-operational interface or function is protected by its documented mechanism and cannot be exercised without the documented unlock path, and operational interfaces expose only documented functions. For example, a boot manager whose serial debug console is fuse-locked in production and unreachable without the documented unlock path, and whose operational update interface exposes no functions beyond those documented, satisfies the requirement.
- Fail: the requirement is not met if a non-operational interface is reachable without its documented protection; or where an operational interface exposes undocumented functions.

Assessment evidence: enumeration of interfaces with record of operational classification; record of protection tests per interface.

6.12.4 [ACC-BM-AMS-004]

Assessment reference: REQ-BM-AMS-004.

Assessment objective: verify that unused or non-essential interfaces and protocols have been removed from the boot manager.

Assessment preparation:

- Test environment: the production build of the boot manager.
- Required information: enumeration of interfaces and protocols supported by the boot manager; for each, the operational purpose that justifies its inclusion.
- Required tools: interface and protocol enumeration tools; binary analysis tools capable of identifying linked protocol stacks or interface handlers.

Assessment activities:

- Enumerate the interfaces and protocols exposed at runtime and, via binary analysis, those compiled into the boot manager.
- For each interface or protocol, confirm an operational-purpose justification exists and the protocol is actively used in documented boot scenarios.
- Identify any interface or protocol present in the binary or at runtime without a documented operational purpose and record it as an unused or non-essential item not removed.

Assessment verdict:

- Pass: every interface and protocol present at runtime or in the binary maps to a documented operational purpose. For example, a boot manager that links only the network protocol stack required for its documented network-boot scenario, with no unused discovery service compiled in, satisfies the requirement.
- Fail: the requirement is not met if an interface or protocol is present at runtime or in the binary without a documented operational purpose.

Assessment evidence: runtime and binary interface/protocol enumeration; justification mapping record.

6.12.5 [ACC-BM-AMS-005]

Assessment reference: REQ-BM-AMS-005.

Assessment objective: verify that the boot manager validates all inputs for conformance with expected formats.

Assessment preparation:

- Test environment: the boot manager accessible through each documented input channel (e.g. configuration files, signed artefacts, network inputs, user interfaces, inter-component messages).
- Required information: the manufacturer's enumeration of input channels; for each, the expected input format and the validation mechanism applied.
- Required tools: input fuzzing tools appropriate to each channel; controlled-input generation tools.

Assessment activities:

- For each input channel, submit well-formed inputs and confirm acceptance.
- Submit inputs violating the expected format in documented ways (e.g. truncated, oversized, malformed encoding, out-of-range values); confirm rejection before any security-relevant processing.
- Submit a sample of non-targeted fuzz inputs on each channel and confirm that the boot manager handles them without entering an undefined state.

Assessment verdict:

- Pass: well-formed inputs are accepted, documented malformed inputs are rejected before any security-relevant processing, and fuzz inputs do not drive the boot manager into an undefined state. For example, a boot manager that accepts a well-formed configuration file, rejects a truncated or oversized one before parsing security-relevant fields, and remains in a defined state under fuzzed input on every documented channel satisfies the requirement.

- Fail: the requirement is not met if a documented malformed input is processed without rejection; or where a fuzz input drives the boot manager into an undefined state.

Assessment evidence: enumeration of input channels; well-formed and malformed test records per channel; fuzz test summary.

6.12.6 [ACC-BM-AMS-006]

Assessment reference: REQ-BM-AMS-006.

Assessment objective: verify that the boot manager detects errors in critical operations, rejects non-compliant inputs, and transitions to error handling on fault.

Assessment preparation:

- Test environment: the boot manager configured for network boot in a controlled environment where inputs can be manipulated and faults can be induced.
- Required information: enumeration of critical operations; for each, the error conditions, the compliance criteria for inputs, and the documented error-handling transition.
- Required tools: controllable input source; fault-injection tools; state inspection tool for the boot manager's error-handling path.

Assessment activities:

- For each enumerated critical operation, induce the documented error condition (e.g. truncated response, invalid signature in a signed resource, protocol-level fault); confirm detection.
- Submit non-compliant inputs (e.g. violating protocol structure, outside value ranges); confirm rejection before the operation completes.
- Confirm that on detection or rejection the boot manager transitions to the documented error-handling state rather than continuing into a potentially compromised path.

Assessment verdict:

- Pass: each enumerated error condition is detected, non-compliant inputs are rejected, and the documented error-handling transition is taken on every fault. For example, a boot manager that detects an invalid signature on a fetched resource, rejects a structurally non-compliant input before the operation completes, and transitions to its documented error-handling state rather than continuing satisfies the requirement.
- Fail: the requirement is not met if an enumerated error condition is not detected; if a non-compliant input is processed; or where the boot manager continues past a detected fault without taking the documented error-handling transition.

Assessment evidence: enumeration of critical operations; fault-injection and non-compliant input test records, per operation; records of error-handling transitions.

6.12.7 [ACC-BM-AMS-007]

Assessment reference: REQ-BM-AMS-007.

Assessment objective: verify that the boot manager provides a mechanism to disable configuration options that are not required by the deployment integrator.

Assessment preparation:

- Test environment: the boot manager accessible through its configuration interface.
- Required information: enumeration of configuration options; for each, the mechanism provided for the deployment integrator to disable or suppress the option; the documentation directed at deployment integrators explaining the disable mechanism.

- Required tools: configuration tool; inspection tool for the boot manager's effective configuration state.

Assessment activities:

- Where the boot manager has no configuration capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- For each configuration option that may be disabled per manufacturer documentation, invoke the disable mechanism and confirm that the option is no longer exposed or active in the boot manager's effective state.
- Confirm that the disable mechanism is documented for the deployment integrator and does not require manufacturer-restricted access.
- Confirm that re-enabling a previously disabled option, where documented, requires the same mechanism accessible to the deployment integrator.

Assessment verdict:

- Pass: each documented configuration option can be disabled through the deployment-integrator-accessible mechanism, disablement takes effect in the boot manager's effective state, and re-enablement where documented uses the same mechanism. For example, a boot manager that lets the deployment integrator disable an optional network-boot path through documented configuration, with the option absent from the effective state afterwards and re-enabled only through the same integrator-accessible mechanism, satisfies the requirement.
- Fail: the requirement is not met if a documented configuration option cannot be disabled by the deployment integrator; if disablement does not take effect; or where re-enablement requires manufacturer-restricted or otherwise undocumented access.

Assessment evidence: enumeration of configuration options with mapping of disable mechanisms; test records of disable and re-enable per option.

6.13 Exploit mitigation

6.13.0 Overview of requirements addressed in clause 6.13

The present clause specifies assessment criteria for the requirement in clause 5.13.

Requirements relevant to exploitation mitigation are also specified in other clauses of the present document and are assessed under the corresponding clauses of clause 6:

- Privilege separation and trust-boundary enforcement: REQ-BM-INT-001, assessed under clause 6.8.
- Resource minimisation and DMA restriction at handoff: REQ-BM-AMS-001, assessed under clause 6.12.
- Access control on cryptographic key material: REQ-BM-CP-001, assessed under clause 6.7.
- Attack-surface reduction: REQ-BM-AMS-002, REQ-BM-AMS-003, REQ-BM-AMS-004, REQ-BM-AMS-007, assessed under clause 6.12.
- Input validation and error handling: REQ-BM-AMS-005, REQ-BM-AMS-006, assessed under clause 6.12.
- Prevention of partial execution on verification failure and of boot continuation with failed components: REQ-BM-AP-013, assessed under 6.10, and REQ-BM-INT-003, assessed under clause 6.8.

Table 6.13.0-1: Requirements addressed in clause 6.13

Requirement	Profile applicability
REQ-BM-EM-001	MEDIUM, HIGH (runtime code-execution or privilege-escalation threats)

6.13.1 [ACC-BM-EM-001]

Assessment reference: REQ-BM-EM-001.

Assessment objective: verify that the boot manager implements exploitation mitigation mechanisms aligned with the documented threat model, addressing unauthorised code execution, privilege escalation between boot stages, and uncontrolled propagation of compromise in the pre-OS execution environment.

Assessment preparation:

- Test environment: the boot manager with exploitation mitigation mechanisms enabled in the documented default configuration, executable in a controlled environment in which targeted exploitation attempts can be applied.
- Required information: documentation of:
  a) the threat model including the threat categories the exploitation mitigation mechanisms are intended to address;
  b) the exploitation mitigation mechanisms implemented in the boot manager (for example compiler-level protections such as stack protection, ASLR, control-flow integrity, position-independent code; runtime integrity protections; isolation between boot stages; memory protection at handoff);
  c) the toolchain and platform features on which those mechanisms depend;
  d) the boot-manager components covered by each mechanism; and
  e) the documented limitations of each mechanism in the pre-OS context.
- Required tools: binary inspection tool appropriate to the boot manager binary format to confirm presence of compiler-level and platform-level mitigations; targeted exploitation framework appropriate to the boot-manager execution environment for attempted bypass; isolation-violation probes for the trust-boundary enforcement covered by REQ-BM-INT-001 and the resource and DMA restrictions covered by REQ-BM-AMS-001.

Assessment activities:

- Where the documented threat model does not include runtime code-execution attacks or stage-to-stage privilege escalation, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Threat-model alignment: confirm that the documented set of exploitation mitigation mechanisms addresses the threat categories identified in the threat model and at minimum covers unauthorised code execution, privilege escalation between boot stages, and uncontrolled propagation of compromise where these are within the scope of the threat model.
- Functional verification: by binary inspection or equivalent technique appropriate to the platform, confirm presence of each documented compiler-level or platform-level mitigation in the boot-manager binary as shipped in the default configuration. Where a mitigation depends on a toolchain or platform feature, confirm the feature is enabled.
- Targeted exploitation attempts: for each documented mitigation, apply at least one targeted exploitation attempt designed to be blocked by that mitigation; confirm the attempt is rejected, contained, or detected. Attempts include, where applicable: code-injection attempts against writable-and-executable regions; control-flow hijack attempts against indirect calls; privilege-escalation attempts crossing documented trust boundaries between boot stages; propagation attempts from a deliberately-faulted boot-manager component to subsequent boot stages.
- Limitation acknowledgement: confirm that documented limitations of each mitigation in the pre-OS context (for example mitigations not available because the underlying hardware does not provide them) are recorded in the product documentation.

Assessment verdict:

- Pass: the documented mitigations address the threat categories in the threat model that fall within the scope of the requirement; each documented mitigation is present in the boot-manager binary and toolchain configuration; targeted exploitation attempts within the scope of those mitigations are rejected, contained, or detected; and the documented limitations are recorded. For example, a binary built with stack protection, control-flow integrity, and position-independent code shows those features present on inspection, a code-injection attempt against a writable region and a control-flow hijack against an indirect call are both blocked, and the hardware features each mitigation depends on are confirmed enabled.

- Fail: the requirement is not met if the documented mitigations leave a threat category within the scope of the requirement unaddressed; if a documented mitigation is absent from the boot-manager binary or toolchain configuration; if a targeted exploitation attempt within the scope of a documented mitigation succeeds; or where a documented limitation is not recorded.

Assessment evidence: The documented threat model relevant to exploitation mitigation; description of documented mitigations; record of binary / toolchain inspection per mitigation; record of targeted exploitation attempts and outcomes; record of documented limitations.

NOTE 1: The mechanisms available to boot managers for exploitation mitigation are constrained by the pre-OS execution environment. Where hardware-side controls would mitigate physical-attack threats (T-PHYS-5, T-PHYS-6, T-PHYS-9, T-PHYS-11), those are addressed under Annex D and are not within the scope of this assessment.

NOTE 2: Informative guidance on exploitation mitigation techniques is provided in Annex D. The techniques listed there are non-exhaustive; the assessment focuses on the mitigations documented by the manufacturer rather than on a fixed checklist of techniques.

6.14 Monitoring

6.14.0 Overview of requirements addressed in clause 6.14

The present clause specifies assessment criteria for the requirements in clause 5.14.

Table 6.14.0-1: Requirements addressed in clause 6.14

Requirement	Profile applicability
REQ-BM-MON-001	MEDIUM, HIGH (if measured boot and logging implemented)
REQ-BM-MON-002	MEDIUM, HIGH (if measured boot and logging implemented)
REQ-BM-MON-003	MEDIUM, HIGH (if logging implemented)
REQ-BM-MON-004	All
REQ-BM-MON-005	MEDIUM, HIGH (if logging and authority transfer implemented)

6.14.1 [ACC-BM-MON-001]

Assessment reference: REQ-BM-MON-001.

Assessment objective: verify that the boot manager records measurements of boot components and security-critical configuration in a tamper-evident way before handoff.

Assessment preparation:

- Test environment: the boot manager executed to completion with access to the retention mechanism used for measurements (e.g. platform configuration registers, tamper-proof storage).

- Required information: the manufacturer's documentation of the measurement retention mechanism, the set of components and configuration items measured, and the tamper-evidence properties claimed for the mechanism.

- Required tools: inspection tool for the retention mechanism appropriate to the platform (e.g. TPM command interface for PCR read, secure-storage inspection utility).

Assessment activities:

- Where the boot manager does not implement measured boot with logging capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Execute a complete boot and, immediately before handoff, read the retention mechanism to obtain the measurement set actually recorded.
- Compare the recorded measurements against the documented set of components and configuration items; confirm the set is complete.
- Verify the tamper-evidence property of the retention mechanism by attempting to alter a recorded measurement via documented and undocumented interfaces; confirm that alteration is either prevented or produces a detectable indication.
- Where retention is volatile (e.g. PCR), confirm that retention across the boot manager's execution to handoff is sufficient for the assessment objective.

Assessment verdict:

- Pass: measurements of all documented components and security-critical configuration items are recorded before handoff, and the retention mechanism is tamper-evident both by inspection and under attempted alteration. For example, a complete boot populates a platform configuration register with the documented measurement set before control is handed off, and an attempt to overwrite a recorded measurement through documented and undocumented interfaces is either prevented or leaves a detectable indication.
- Fail: the requirement is not met if a documented measurement is missing or incomplete; if measurements are recorded only after handoff; or where the retention mechanism permits a recorded measurement to be altered without detection.

Assessment evidence: record of retention mechanism inspection; record of measurement-set completeness; record of tamper-alteration tests.

6.14.2 [ACC-BM-MON-002]

Assessment reference: REQ-BM-MON-002.

Assessment objective: verify that the boot manager provides measurement records in a format that supports remote attestation and includes mechanisms to ensure freshness against replay.

Assessment preparation:

- Test environment: the boot manager configured to emit measurement records to an external attestation verifier.
- Required information: the specification of the measurement record format, the supported freshness mechanism (e.g. nonces, monotonic counters, timestamps, challenge-response), and the attestation protocol or format the records conform to.
- Required tools: attestation verifier or parser capable of consuming the specified record format; tool for generating attestation challenges where the freshness mechanism is challenge-based.

Assessment activities:

- Where the boot manager does not implement measured boot with logging capability, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Capture a measurement record produced by the boot manager and parse it using an attestation verifier implementing the specified format.
- Confirm that the parsed record contains at least the fields required for remote attestation: component identifiers, measurement digests, and the freshness element.
- Issue a challenge (or consume a nonce) as supported by the freshness mechanism; verify that the boot manager's response binds to the challenge or nonce.

- Replay a previously valid measurement record to the verifier and confirm that the replay is detected by the freshness mechanism.

Assessment verdict:

- Pass: the measurement records parse against the specified attestation format and contain the fields required for remote attestation, and the freshness mechanism binds each record to a fresh element and rejects replayed records. For example, an attestation verifier parses a captured record, finds the component identifiers, measurement digests, and a nonce bound to the verifier's challenge, and rejects a previously valid record replayed against the same challenge.

- Fail: the requirement is not met if a measurement record fails to parse against the specified format; if a field required for remote attestation is missing; if the freshness mechanism is absent; or where a replayed record is not detected.

Assessment evidence: sample of parsed measurement records; record of challenge-response exchange; record of replay rejection.

6.14.3 [ACC-BM-MON-003]

Assessment reference: REQ-BM-MON-003.

Assessment objective: verify that the boot manager produces log indications for security-relevant failures and state changes.

Assessment preparation:

- Test environment: the boot manager executed under conditions that can be induced to produce each class of security-relevant event.

- Required information: the enumeration of security-relevant failures and state changes recorded by the boot manager (at minimum: verification failures, authentication failures, security policy violations, recovery mode activation, execution of unsigned code), the log output mechanism, and the log record format.

- Required tools: fault-injection or stimulation tools for each event class (e.g. corrupted boot image for verification failure, invalid credentials for authentication failure); log capture tool appropriate to the log output mechanism.

Assessment activities:

- Where the boot manager does not implement logging capability, record the requirement as not applicable and close this assessment per clause 5.1.3.

- For each enumerated event class, induce the event (e.g. present a boot image with an invalid signature to trigger a verification failure) and capture the boot manager's log output.

- Confirm that an identifiable log entry is produced for each induced event, that the entry identifies the event class, and that the entry is consistent with the documented log record format.

- Confirm that log entries are produced at the time the event occurs and are not suppressed under ongoing failure conditions.

Assessment verdict:

- Pass: each enumerated class of security-relevant failure or state change produces an identifiable log entry that names the event class and matches the documented record format, at the time the event occurs. For example, presenting a boot image with an invalid signature produces a verification-failure entry distinguishable from an authentication-failure entry, each in the documented format, and the entries are not suppressed when failures continue.

- Fail: the requirement is not met if an enumerated event class produces no log entry; if the entry cannot be distinguished by event class; if it is inconsistent with the documented format; or where entries are suppressed under ongoing failure conditions.

NOTE: Logging of security-relevant failure conditions is also required by REQ-BM-AP-010 (assessed under clause 6.10). Where both apply, the log capture performed here also serves that assessment and need not be repeated.

Assessment evidence: enumeration of event classes; records of fault-injection tests; captured log entries for each induced event.

6.14.4 [ACC-BM-MON-004]

Assessment reference: REQ-BM-MON-004.

Assessment objective: verify that the boot manager provides version information accessible to the operating system, management systems, or authorised entity.

Assessment preparation:

- Test environment: the boot manager executed to completion with the interfaces accessible to operating system, management system or authorised entity consumers.
- Required information: the documentation of the version information exposed (content and format), the access mechanism (e.g. ACPI table, UEFI variable, runtime service, platform-specific register), and the identity of the consumer role accessing the information.
- Required tools: consumer-role tool for the access mechanism (e.g. operating-system utility reading the exposed structure).

Assessment activities:

- Query the boot manager's version information through each documented access mechanism.
- Confirm that the returned information contains the documented version fields (e.g. boot manager version, build identifier, relevant component versions).
- Confirm that the access is available to the documented consumers (operating system, management system, authorised entity) and that no authorisation step beyond those documented is required.

Assessment verdict:

- Pass: the version information is accessible through each documented access mechanism, to each documented consumer role, and contains the documented version fields, with no authorisation step beyond those documented. For example, an operating-system utility reads the exposed structure and obtains the boot manager version and build identifier without any undocumented authorisation, and a management system reaches the same information through its documented interface.
- Fail: the requirement is not met if a documented access mechanism fails to return the version information; if the returned information is missing a documented field; or where access requires an authorisation step beyond those documented.

Assessment evidence: sample query and response through each documented access mechanism; record of field completeness.

6.14.5 [ACC-BM-MON-005]

Assessment reference: REQ-BM-MON-005.

Assessment objective: verify that the boot manager records each authority transfer as a security-relevant event in the security log, with the prescribed event fields.

Assessment preparation:

- Test environment: the boot manager with logging capability active and at least one authority transfer mechanism exposed.

- Required information: the manufacturer's enumeration of authority transfer mechanisms supported by the boot manager (e.g. trust anchor enrolment, ownership transfer, policy update), the log record format, the log output mechanism, and the field-by-field specification of the authority transfer event (mechanism invoked, initiating entity, approving entity, components affected).
- Required tools: invocation tool for each authority transfer mechanism; log capture tool appropriate to the log output mechanism; record-format parser.

Assessment activities:

- Where the boot manager does not implement logging capability or provides no authority transfer mechanism, record the requirement as not applicable and close this assessment per clause 5.1.3
- For each documented authority transfer mechanism, invoke a transfer under documented conditions; capture the boot manager's log output.
- Confirm that the log captures an authority transfer event corresponding to each invocation, identifiable by event class.
- Parse the captured event against the documented format; confirm presence of each prescribed field (mechanism invoked, initiating entity, approving entity, components affected) and that the recorded values match the invocation.
- Invoke a transfer that fails authorisation; confirm that the failure is also recorded with the prescribed fields, identifying the attempted-but-rejected nature of the event.

Assessment verdict:

- Pass: every documented authority transfer mechanism produces a security-relevant log event recording the mechanism invoked, the initiating entity, the approving entity, and the components affected, with values consistent with the invocation, and a transfer that fails authorisation is recorded with the same fields. For example, invoking trust-anchor enrolment produces a log event naming the enrolment mechanism, the entity that initiated it, the entity that approved it, and the affected trust store, and an enrolment attempt that fails authorisation is recorded as a rejected event with those same fields.
- Fail: the requirement is not met if a documented authority transfer mechanism produces no log event; if a recorded event omits any prescribed field; if a recorded value is inconsistent with the invocation; or where a rejected transfer is not recorded.

Assessment evidence: enumeration of authority transfer mechanisms; per-mechanism log capture; field-by-field verification record; record of rejected-transfer logging.

6.15 Factory reset and data portability

6.15.0 Overview of requirements addressed in clause 6.15

The present clause specifies assessment criteria for the requirements in clause 5.15.

Table 6.15.0-1: Requirements addressed in clause 6.15

Requirement	Profile applicability
REQ-BM-FRDP-001	MEDIUM, HIGH (if user-modifiable data stored)

6.15.1 [ACC-BM-FRDP-001]

Assessment reference: REQ-BM-FRDP-001.

Assessment objective: verify that the boot manager provides a mechanism that, when invoked by an authorised entity, securely and permanently removes all user-modifiable data and settings stored by the boot manager and restores the boot manager to its default state as placed on the market.

Assessment preparation:

- Test environment: the boot manager configured with non-default user-modifiable data and settings populated (boot configuration deltas, user-added trust anchors, audit log entries, any other persistent state outside the as-shipped default).
- Required information: the inventory of user-modifiable data categories stored by the boot manager; the specification of the default state as placed on the market; the documented invocation interface for the factory reset mechanism; the authority model for "authorised entity" in the context of factory reset; the description of the secure removal technique used per data category.
- Required tools: invocation tool for the factory reset mechanism; storage-inspection tool appropriate to each storage technology used (flash region reader, secure-element command interface, etc.); recovery-attempt tool capable of attempting to read previously-stored values from the underlying storage media after reset.

Assessment activities:

- Where the boot manager stores no user-modifiable data or settings, record the requirement as not applicable and close this assessment per clause 5.1.3.
- Populate each user-modifiable data category enumerated in the inventory with non-default values; confirm the populated values via inspection.
- Attempt invocation of the factory reset mechanism by an entity not in the documented authorised-entity scope; confirm rejection.
- Invoke the factory reset mechanism by a documented authorised entity; confirm successful completion as documented.
- Inspect each storage location after reset and confirm: (a) the values populated in step 1 are no longer present and (b) the documented default state is restored.
- For each storage technology used, perform a recoverability check appropriate to the technology (e.g. raw flash read for flash storage, secure-element residue check for SE-held data); confirm that the populated values are not recoverable.
- Confirm that the default state as placed on the market is itself not removed by the mechanism (the as-shipped trust anchors, default configuration, and any non-user-modifiable state remain present after reset).

Assessment verdict:

- Pass: the factory reset mechanism rejects invocation by an entity outside the documented authorised scope, and under authorised invocation it removes every enumerated user-modifiable data category from the documented storage with no recoverable residue, restores the documented default state, and leaves that default state itself intact. For example, an unauthorised caller is refused, an authorised reset clears the populated boot configuration, user-added trust anchors, and audit log so that a raw read of the underlying storage recovers none of them, and the as-shipped trust anchors and default configuration remain present afterwards.
- Fail: the requirement is not met if an unauthorised invocation succeeds; if any populated user-modifiable data category remains present or is recoverable from the underlying storage after reset; if the documented default state is not restored; or where the default state is itself removed by the mechanism.

Assessment evidence: Inventory of user-modifiable data categories; record of populated values; record of unauthorised invocation rejection; record of authorised invocation; per-storage post-reset inspection; per-storage recoverability record; record of default-state preservation.

NOTE: Secure disposal of cryptographic key material is also required by REQ-BM-CP-008 (assessed under clause 6.7). Where key material falls within the user-modifiable data removed here, its disposal is assessed under clause 6.7 and need not be repeated.

Annex A (informative): Relationship between the present document and the essential cybersecurity requirements of EU Regulation (EU) 2024/2847 - The Cyber Resilience Act

The present document has been prepared under the Commission's standardisation request C(2025) 618 [i.3] final to provide one voluntary means of conforming to the requirements of Regulation (EU) 2024/2847 of the European Parliament and of the Council of 23 October 2024 on horizontal cybersecurity requirements for products with digital elements and amending Regulations (EU) No 168/2013 and (EU) No 2019/1020 and Directive (EU) 2020/1828 (Cyber Resilience Act) (CRA)[i.1].

Once the present document is cited in the Official Journal of the European Union under Regulation (EU) 2024/2847 [i.1], compliance with the normative clauses of the present document given in Table A.1 confers, to products with digital elements in the scope of the present document, a presumption of conformity with the corresponding essential requirements of that Regulation and associated EFTA regulations.

Table A.1: Correspondence between the European Standard and Annex I Part I of Regulation (EU) 2024/2847

Description	Essential Requirements of Regulation (EU) 2024/2847	Clause(s) of the present document	Remarks / Notes
Annex I, Part I, (1)	"Products with digital elements shall be designed, developed and produced in such a way that they ensure an appropriate level of cybersecurity based on the risks."	Clause 5.2 Clause 6.2	
Annex I, Part I, (2)(a)	"Products with digital elements shall be made available on the market without known exploitable vulnerabilities."	Clause 5.3 Clause 6.3	
Annex I, Part I, (2)(b)	"Products with digital elements shall be made available on the market with a secure by default configuration, unless otherwise agreed between manufacturer and business user in relation to a tailor-made product with digital elements, including the possibility to reset the product to its original state."	Clause 5.4 Clause 6.4	
Annex I, Part I, (2)(c)	"Products with digital elements shall ensure that vulnerabilities can be addressed through security updates, including, where applicable, through automatic security updates that are installed within an appropriate timeframe enabled as a default setting, with a clear and easy-to-use opt-out mechanism, through the notification of available updates to users, and the option to temporarily postpone them"	Clause 5.5 Clause 6.5	
Annex I, Part I, (2)(d)	"Products with digital elements shall ensure protection from unauthorised access by appropriate control mechanisms, including but not limited to authentication, identity or access management systems, and report on possible unauthorised access"	Clause 5.6 Clause 6.6	
Annex I, Part I, (2)(e)	"Products with digital elements shall protect the confidentiality of stored, transmitted or otherwise processed data, personal or other, such as by encrypting relevant data at rest or in transit by best practice mechanisms, and by using other technical means."	Clause 5.7 Clause 6.7	

Description	Essential Requirements of Regulation (EU) 2024/2847	Clause(s) of the present document	Remarks / Notes
Annex I, Part I, (2)(f)	"Products with digital elements shall protect the integrity of stored, transmitted or otherwise processed data, personal or other, commands, programs and configuration against any manipulation or modification not authorised by the user, and report on corruptions."	Clause 5.8 Clause 6.8	
Annex I, Part I, (2)(g)	"Products with digital elements shall process only data, personal or other, that are adequate, relevant and limited to what is necessary in relation to the intended purpose of the product with digital elements (data minimisation)."	Clause 5.9 Clause 6.9	
Annex I, Part I, (2)(h)	"Products with digital elements shall protect the availability of essential and basic functions, also after an incident, including through resilience and mitigation measures against denial-of-service attacks."	Clause 5.10 Clause 6.10	
Annex I, Part I, (2)(i)	"Products with digital elements shall minimise the negative impact by the products themselves or connected products on the availability of services provided by other products or networks."	Clause 5.11 Clause 6.11	
Annex I, Part I, (2)(j)	"Products with digital elements shall be designed, developed and produced to limit attack surfaces, including external interfaces."	Clause 5.12 Clause 6.12	
Annex I, Part I, (2)(k)	"Products with digital elements shall be designed, developed and produced to reduce the impact of an incident using appropriate exploitation mitigation mechanisms and techniques."	Clause 5.13 Clause 6.13	
Annex I, Part I, (2)(l)	"Products with digital elements shall provide security related information by recording and monitoring relevant internal activity, including the access to or modification of data, services or functions, with an opt-out mechanism for the user."	Clause 5.14 Clause 6.14	
Annex I, Part I, (2)(m)	"Products with digital elements shall provide the possibility for users to securely and easily remove on a permanent basis all data and settings and, where such data can be transferred to other products or systems, ensure that this is done in a secure manner."	Clause 5.15 Clause 6.15	

Key to columns:

Description A textual reference to the requirement.

Requirements of Regulation

Identification of point(s) defining the requirement in Regulation (EU) 2024/2847 - the Cyber Resilience Act.

Clause(s) of the present document

Identification of clause(s) addressing the requirement in the present document

Presumption of conformity stays valid only as long as a reference to the present document is maintained in the list published in the Official Journal of the European Union. Users of the present document should consult frequently the latest list published in the Official Journal of the European Union.

Other Union legislation may be applicable to the product(s) falling within the scope of the present document.

Annex B (informative): Security analysis

B.0 Introduction

This annex applies state of the art methodology to identify assets, threats, identify and evaluate risk factors, and define security profiles applicable to the different use cases identified in the product context. It identifies assets (clause B.1), risk factors (clause B.2), assumptions (clause B.3), threats (clause B.4), and maps risk factors to use cases (clause B.5) and security profiles (clause B.6).

Boot managers face unique challenges: pre-OS execution, establishment of platform trust, and extreme product diversity from 256-byte ROM code to full UEFI systems. This diversity makes requirements based on deployment scenarios impractical since manufacturers often cannot predict final deployment context.

The standard uses a capability-based model where objective technical features determine which threats are relevant and which requirements apply. Each product capability has a dual security impact: it introduces an attack surface requiring protective measures and enables security mechanisms that mitigate specific threats. For example, update capability enables vulnerability remediation while requiring protection against malicious update injection. Requirements address both aspects.

B.1 Assets

B.1.1 A-CRYPT: Cryptographic keys and certificates

Keys and certificates establishing trust in the boot chain. Compromise enables signing malicious code as legitimate or decrypting protected data.

- Root verification keys (platform keys, root certificates).
- Signature database.
- Device attestation keys.
- Symmetric encryption keys.
- Revocation data.

NOTE: Platform keys require special attention due to supply chain risks including default keys and key leakage. See T-SUPPLY-6.

B.1.2 A-ROLLBACK: Rollback protection data

Counters and version information that prevent downgrade to vulnerable versions. Compromise enables rollback attacks where attackers exploit known patched vulnerabilities.

- Version counters (monotonic, non-decreasing).
- Minimum version numbers.
- Anti-rollback enforcement data.

B.1.3 A-CONFIG: Configuration data

Settings that control boot manager security behaviour. Compromise allows attackers to weaken security policies or enable unauthorised boot paths.

- Security policy settings.
- Trust anchor configuration.
- Boot device selection and priority.
- Network boot credentials (when present).
- Recovery mode settings.
- Debug interface lockdown state.
- Parsed input data (e.g. boot images, logo images, UEFI variables, configuration files).

NOTE: Parsed input is attack surface for parser vulnerabilities. See T-INTEGRITY-14.

B.1.4 A-AUTH: Authentication credentials

Credentials controlling access to boot manager configuration and recovery functions. Compromise enables unauthorised configuration changes and security policy modifications.

- Setup/BIOS passwords (hashed).
- Administrator passwords (hashed).
- Physical presence authentication.
- Recovery authentication data.

B.1.5 A-MEASURE: Measurement and attestation data

Records of boot components and configuration for remote verification. Compromise allows attackers to hide malicious modifications and appear trustworthy during attestation.

- Event log.
- Measurement metadata (component identities, versions, algorithms used).

B.1.6 A-AUDIT: Audit logs

Records of security-relevant events during boot. Compromise allows attackers to hide evidence of attacks and prevent detection or forensic investigation.

- Verification success/failure events.
- Authentication attempts.
- Configuration change records.
- Security policy violation attempts.
- Recovery mode activation events.

B.1.7 A-RUNTIME: Runtime state data

Security state during boot execution. Compromise enables attackers to bypass security checks, manipulate boot decisions, or enable unlimited authentication attempts.

- Security mode state (setup mode, user mode, deployed mode).
- Verification completion flags.
- Boot attempt counters.
- Memory protection configuration.
- Boot phase state machine.

NOTE: State transitions are attack surface for privilege escalation. See T-INTEGRITY-15.

B.1.8 A-CODE: Boot manager executable code

The boot manager code itself while executing or staged for update. Compromise provides persistent privileged access and complete control over boot process.

- Boot manager code in writable memory during execution.
- Runtime services code (if persisting after boot target handoff).
- Update staging area contents.

B.1.9 A-UPDATE: Update packages and delivery mechanisms

Update packages during delivery, staging, and installation, plus update tooling.

- Update packages in transit.
- Staged updates awaiting installation.
- Update signature metadata.
- Update verification tools.
- Update delivery infrastructure trust anchors.

NOTE: Update tools themselves are attack targets (BIOS Disconnect). See T-INTEGRITY-16, T-SUPPLY-7.

B.2 Risk Factors

B.2.1 General

Risk factors characterise the security-relevant aspects of boot manager deployment. They provide the analytical basis for mapping of the use cases to the security profiles (clauses B.5 and B.6). Four risk factors are defined. RF-SURFACE is determined by the product's implemented capabilities. RF-NET, RF-AVAIL, and RF-IMPACT are determined by the deployment context described in clause 4.3.

B.2.2 RF-SURFACE: Capability-driven attack surface

The set of capabilities implemented in the boot manager determines the attack surface exposed during boot. Each additional capability introduces interfaces, protocols, or state that an attacker may target.

Table B.2.2-1: Capability-driven attack surface levels

Level	Description	Examples
Minimal	Immutable code, no configuration or update interfaces	ROM-only, fixed boot path
Standard	Verified boot, update capability, configuration management, logging	Updateable boot manager with local interfaces
Extended	Standard plus network boot, remote configuration, debug interfaces, or measured boot	Full-featured boot manager with network and remote management

B.2.3 RF-NET: Network reachability

Network reachability of the boot manager during boot, update, or configuration operations. Elevated network exposure increases remote attack surface before operating system security mechanisms are active. See clause 4.3.3 for deployment context.

Table B.2.3-1: Network reachability levels

Level	Description	Examples
Isolated	No network connectivity during boot	Air-gapped, internal storage only
Managed	Network connectivity within managed perimeter	Enterprise with network boot via VPN
Exposed	Direct internet or untrusted network exposure	Edge devices

B.2.4 RF-AVAIL: Operational continuity

The operational continuity expectations for the system in which the boot manager is deployed. Higher continuity expectations increase the impact of boot failures and constrain acceptable recovery strategies. See clauses 4.4.5 and 4.4.8 for deployment context.

Table B.2.4-1: Operational continuity levels

Level	Description	Examples
Flexible	Downtime acceptable for maintenance and recovery	Development boards, consumer devices
Standard	Normal business availability expectations	Office equipment, personal computing
Critical	Continuous operation required; boot failure causes safety impact or critical service disruption	Medical devices, industrial control, telecommunications infrastructure

B.2.5 RF-IMPACT: Compromise severity

The severity of consequences if the boot manager is compromised, considering data sensitivity, safety implications, and downstream effects on other system components. See clauses 4.4.5 and 4.4.8 for deployment context.

Table B.2.5-1: Compromise severity levels

Level	Description	Examples
Limited	No sensitive data, no safety impact	Development boards, consumer devices
Significant	Personal or business data, moderate operational disruption	Consumer computing, enterprise workstations
Critical	Safety-critical systems, highly sensitive data, or critical infrastructure	Medical devices, financial infrastructure, government systems

B.3 Assumptions

B.3.1 General

The security analysis in this annex assumes the following conditions hold. These assumptions describe properties that the boot manager depends on but cannot itself verify or enforce. Where an assumption does not hold, the corresponding threats identified in clause B.4 are not mitigated by the present document.

B.3.2 Hardware platform assumptions

AS-HW-1: The hardware root of trust (initial boot code, e-fuses, or equivalent) is correctly implemented and has not been modified without authorisation after manufacture. [T-PHYS-5, T-PHYS-6, T-PHYS-9]

AS-HW-2: Hardware memory protection and privilege separation mechanisms function as specified by the platform. [T-PHYS-2]

AS-HW-3: Hardware security components (TPM, secure element, HSM) correctly isolate key material and resist physical probing at the level claimed by the component. [T-PHYS-3, T-PHYS-4]

AS-HW-4: Hardware debug interfaces (JTAG, SWD) are disabled or protected by the platform when claimed. [T-PHYS-11]

AS-HW-5: The platform provides sufficient entropy for cryptographic key generation. [T-INTEGRITY-12]

AS-HW-6: The platform provides hardware mechanisms to resist fault injection (voltage glitching, EM interference) when claimed. [T-PHYS-2]

B.3.3 Process assumptions

AS-PROC-1: The manufacturing environment provisions correct initial key material and firmware. [T-SUPPLY-1]

AS-PROC-2: Development toolchains have not been compromised. [T-SUPPLY-3]

AS-PROC-3: Update distribution infrastructure is operated securely by the manufacturer. [T-SUPPLY-4, T-SUPPLY-5]

AS-PROC-4: Update tooling is authentic and has not been tampered with. [T-SUPPLY-7]

B.3.4 Boot target assumptions

AS-TARGET-1: The post-handoff stage (operating system, hypervisor, or next-stage loader) implements security properties consistent with the declared use case, or another verification mechanism takes over at handoff. Applies to all use cases.

AS-TARGET-2: For UC-IMM, the post-handoff stage maintains equivalent immutability, or the boot target itself is immutable. Where this assumption does not hold, the absence of post-deployment remediation that defines UC-IMM's risk posture extends only to the boot manager and not to the running system.

B.4 Threats

B.4.1 Security property impacts

B.4.1.1 General

Boot managers present unique risks due to their privileged position, persistence, invisibility, and role as trust foundation:

- Privileged position: Operates at highest privilege level with direct hardware access.
- Persistence: Survives OS reinstall and traditional security remediation.
- Invisibility: Executes below OS visibility and security tools.
- Trust establishment: Forms the foundation for all subsequent security decisions

These characteristics mean boot manager compromises have disproportionate impact compared to application or OS-level compromises. Different capabilities introduce different threat exposures.

B.4.1.2 Confidentiality

The confidential information that the bootloader may have access to include the storage encryption keys, network access keys, and the user or device identity. This information may be stored in non-volatile storage, in a trusted hardware element, or retrieved from another system as part of a remote attestation.

If this data is stored in clear text in non-volatile storage, a local attacker may be able to gain access to the confidential information. This sort of storage is only suggested for use cases where physical access is protected by other means.

Even if the confidential data is stored in a trusted hardware component, the local bus that connects the bootloader to the external hardware may be compromised by a local attacker. In some cases, the secrets are returned to the bootloader in plain text on the bus, putting them at risk of disclosure.

If the secrets are encrypted on the bus, but the measurements from the bootloader to the trusted hardware component are not, the device may be susceptible to a replay attack leading to disclosure of the secrets.

If the devices on the local bus are not authenticated in some way, a local attacker might be able to install an interposer to intercept the measurements and modify them, similar to the replay attack, to reveal the secrets.

If the attacker can downgrade from verified to measured boot, they should not be able to unseal confidential data since the configuration measurements will not match, but they might be able to present an authentic looking authentication screen to the user and trick them into revealing recovery or other secrets.

During a remote attestation, an attacker may be able to force a protocol downgrade that could lead to a disclosure of user identity or the returned secrets.

A loss of confidentiality of disk encryption keys could lead to a larger loss of confidentiality of the disk contents, as well as could be leveraged to a persistent loss of integrity in the operating system.

B.4.1.3 Integrity

Integrity for the bootloader means that the OS that it hands control to is the one that the system's owner intends, that the system is configured as intended, and any hardware measurements correctly reflect this configuration. Any attack that compromises integrity can also compromise availability and confidentiality, and potentially compromise the availability, confidentiality and integrity of the operating system that it loads.

An attacker that can modify the bootloader configuration can lead to a loss of integrity, such as by changing the verification keys to load an attacker signed operating system or enabling a debug mode that does not perform validation.

An attacker that can force the bootloader to execute untrusted code, through misconfiguration or other means such as TOCTOU, can lead to a loss of integrity.

An attacker might be able to modify measurements when they are being sent to the trusted hardware element, leading to a failure of integrity of the measurements, which could lead to faked attestations.

An attacker might rollback the system configuration to an older, vulnerable but still signed version that could be exploited to compromise integrity unless there are rollback protections.

Additionally, some attacks on bootloader integrity can lead to persistence in flash, disk, or other storage, allowing the attacker to compromise the integrity of not only the current boot, but all future boots as well.

Attacks on bootloader runtime services from the operating system or from a hypervisor guest are also a potential attack on overall system integrity since they often have elevated privilege levels above the hypervisor itself. This is another means that an attacker on the bootloader could gain persistence on the system after the bootloader has passed control to the operating system.

Audit and security event logs may be targeted by attackers to hide evidence of boot compromise. Loss of audit log integrity prevents forensic investigation and attack detection.

B.4.1.4 Availability

If an attacker changes any of the boot configuration in a verified boot system, the device may no longer boot and might not be accessible by the user. Depending on when during the boot process the configuration verification fails, communicating the nature of the failure to the user may be challenging. System designers should consider recovery flows for re-establishing trust in the device if the bootloader verification fails.

Corruption of rollback protection, measurement data, or runtime state can cause boot failures. Critical asset unavailability can render the system unbootable, creating support and recovery challenges.

B.4.1.5 Impact of other devices

Integrity failures during the bootloader process can leave the attacker in a position to interfere with other devices on the network or future components in the boot chain.

Availability failures could lead to a boot loop where the bootloader continuously tries to authenticate with a remote server.

B.4.2 Threat identification methodology

B.4.2.0 Introduction

This risk analysis uses three interconnected elements:

Assets (clause B.1) Nine categories requiring protection: code, configuration, runtime state, cryptographic material, measurements, audit logs, authentication credentials, update packages and rollback counters.

Threats (clause B.4.3): Six threat categories: T-INTEGRITY, T-PERSIST, T-PHYS, T-SUPPLY, T-NET, T-AVAIL.

Product capabilities (clause 4.2): Technical features determining which threats are relevant and which security mechanisms are available.

The analysis maps product characteristics to threat exposure:

- Which threats become relevant based on technical features.
- Which assets are impacted.
- Risk implications for each characteristic.

Each product characteristic clause provides:

- Relevant threats for each characteristic variant.
- Assets affected.

- Threat exposure analysis.
- Risk implications for manufacturers.

B.4.2.1 Development approach

The threat taxonomy in clause B.4.3 was developed through analysis of documented attacks, security frameworks, regulatory requirements, and expert review.

B.4.2.2 Threat sources

- NIST SP 800-193 [i.10] Platform Firmware Resiliency Guidelines, Appendix A attack scenarios.
- Threat frameworks: MITRE EMB3D [i.12], CAPEC [i.11], CWE firmware patterns [i.13].
- Documented attacks, for example BlackLotus (CVE-2022-21894), BootHole (CVE-2020-10713), LogoFAIL (CVE-2023-40238), BootBandit (CVE-2022-34301).
- CRA Annex I: Essential cybersecurity requirements for supply chain, updates, vulnerability handling.

B.4.2.3 Categorisation rationale

Six threat categories separate attacks by vector and persistence mechanism. Categories reflect attack mechanisms while acknowledging sophisticated attacks combine multiple vectors.

B.4.2.4 Limitations

Taxonomy represents threats known at standard development.

B.4.3 Threat catalogue

B.4.3.1 Introduction

This clause provides the threat catalogue referenced by requirements in clause 5. Each threat is identified by a unique identifier. Threat relevance is determined by capability presence. Threat exposure level is determined by risk factor analysis per clause B.2.

NOTE: The "Mitigated by" column uses the following markers: REQ-BM-ESR-NNN: Requirement(s) that mitigate this threat.

Threats marked [HW] or [PROC] are not addressable by boot manager software alone.

B.4.3.2 T-INTEGRITY: Boot integrity attacks

B.4.3.2.0 Introduction

Boot integrity attacks target the authenticity and trustworthiness of boot components, attempting to execute unauthorised code, weaken security policies, bypass verification mechanisms, or exploit parsing and input validation weaknesses. Threat exposure increases with RF-SURFACE (additional capabilities introduce verification and parsing attack surface).

B.4.3.2.1 Code execution attacks

Attacks that achieve execution of unauthorised code during boot.

Table B.4.3.2.1-1: Code execution attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-INTEGRITY-1	A-CODE	All products	Malicious boot component substitution	REQ-BM-INT-002, REQ-BM-INT-003
T-INTEGRITY-2	A-CODE	Verified boot	Signature verification bypass	REQ-BM-INT-002, REQ-BM-INT-003
T-INTEGRITY-3	A-CODE, A-UPDATE	Update capability	Malicious update injection	REQ-BM-INT-005, REQ-BM-SU-004, REQ-BM-SU-005
T-INTEGRITY-4	A-RUNTIME, A-CODE	All products	Runtime code modification	REQ-BM-INT-009

B.4.3.2.2 Configuration tampering

Attacks that modify security policies and boot settings to weaken protections.

Table B.4.3.2.2-1: Configuration tampering

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-INTEGRITY-5	A-CONFIG, A-RUNTIME	Configuration capability	Security policy weakening via configuration interface	REQ-BM-AC-002, REQ-BM-AC-003, REQ-BM-AC-004
T-INTEGRITY-6	A-CONFIG	Configuration capability	Boot order manipulation to load attacker media	REQ-BM-AC-002
T-INTEGRITY-7	A-CONFIG, A-CRYPT	Configuration and update capability	Trust anchor database modification	REQ-BM-AC-001, REQ-BM-CP-003
T-INTEGRITY-8	A-CONFIG, A-RUNTIME	Runtime update and configuration capability	Configuration TOCTOU exploitation	REQ-BM-INT-009
T-INTEGRITY-9	A-CONFIG	Recovery capability	Recovery configuration abuse	REQ-BM-AP-008, REQ-BM-AP-012

B.4.3.2.3 Cryptographic compromise

Attacks targeting keys, certificates, and cryptographic material.

Table B.4.3.2.3-1: Cryptographic compromise

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-INTEGRITY-10	A-CRYPT	Software-only implementation	Key extraction via software attacks	REQ-BM-RRM-005 [HW}
T-INTEGRITY-11	A-CRYPT	Configuration and update capability	Key injection/replacement via interfaces	REQ-BM-AC-001
T-INTEGRITY-12	A-CRYPT	Verified and measured boot	Weak key generation from insufficient entropy	REQ-BM-RRM-005 [HW]
T-INTEGRITY-13	A-CRYPT	Long lifetime	Cryptographic algorithm exploitation	REQ-BM-INT-020, REQ-BM-INT-021

B.4.3.2.4 Parser and input validation attacks

Table B.4.3.2.4-1: Parser and input validation attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-INTEGRITY-14	A-CODE, A-RUNTIME	All products	Boot image and data format parsing vulnerabilities	REQ-BM-AMS-005
T-INTEGRITY-15	A-CODE, A-CONFIG	Network boot	Network protocol parsing vulnerabilities	REQ-BM-AMS-005, REQ-BM-AMS-006
T-INTEGRITY-16	A-CRYPT	Verified boot	Certificate and cryptographic format parsing vulnerabilities	REQ-BM-INT-018
T-INTEGRITY-17	A-RUNTIME	All products	Boot phase state machine manipulation	REQ-BM-INT-001, REQ-BM-AMS-002
T-INTEGRITY-18	A-CRYPT, A-CODE	Long lifetime, verified and measured boot	Cryptographic algorithm deprecation over extended deployment	REQ-BM-INT-020, REQ-BM-INT-021, REQ-BM-AP-011
T-INTEGRITY-19	A-CRYPT	Long lifetime, verified boot	Key compromise accumulation over extended periods	REQ-BM-RRM-005 [HW]

B.4.3.2.5 Rollback attacks

Attacks that install older, vulnerable versions of firmware or configuration.

Table B.4.3.2.5-1: Rollback attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-INTEGRITY-20	A-ROLLBACK, A-CODE	Update capability	Forced version downgrade or anti-rollback bypass	REQ-BM-INT-016, REQ-BM-INT-017, REQ-BM-SBD-004
T-INTEGRITY-21	A-ROLLBACK	Update capability, software-only implementation	Version counter manipulation or reset	REQ-BM-INT-016, REQ-BM-INT-017, [HW]
T-INTEGRITY-22	A-CONFIG	Configuration capability	Configuration rollback to insecure settings	REQ-BM-INT-016, REQ-BM-MON-001

NOTE: Products without update capability are immune to firmware rollback attacks but face elevated risk from vulnerability accumulation.

B.4.3.3 T-PERSIST: Persistent firmware threats

Persistent firmware threats establish malware or compromise that survives operating system reinstallation, disk formatting, and traditional security remediation. Threat exposure increases with RF-SURFACE (writable storage and update mechanisms enable persistence).

Table B.4.3.3-1: Persistent firmware threats

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-PERSIST-1	A-CODE	Fixed storage, removable media, external expansion	Bootkit installation in boot manager storage	REQ-BM-INT-002, REQ-BM-INT-003, REQ-BM-INT-006
T-PERSIST-2	A-CODE	Fixed storage, removable media	Boot image tampering in persistent storage	REQ-BM-INT-002, REQ-BM-INT-006
T-PERSIST-3	A-CONFIG	Recovery capability	Recovery mechanism compromise	REQ-BM-AP-008, REQ-BM-AP-012
T-PERSIST-4	A-MEASURE	Measured boot	Attestation data forgery to hide compromise	REQ-BM-MON-001, REQ-BM-MON-002
T-PERSIST-6	A-CODE, A-CONFIG	Network boot	Network boot infrastructure compromise	REQ-BM-INT-013, REQ-BM-INT-015
T-PERSIST-7	A-RUNTIME, A-AUDIT	All products	Boot resource exhaustion preventing successful boot completion	REQ-BM-AP-003, REQ-BM-AP-004

B.4.3.4 T-PHYS: Physical attacks

Physical attacks require direct hardware access to extract secrets, modify hardware, inject faults, or manipulate components. Physical security context determines attack feasibility more than boot manager capabilities. Threat exposure is independent of risk factors. Physical security context is an assumption (see clause B.3.2).

Table B.4.3.4-1: Physical attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-PHYS-1	A-CRYPT, A-RUNTIME, A-CODE	Boot from external expansion, software-only implementation	DMA attacks via peripheral buses	REQ-BM-AMS-001, REQ-BM-CP-005
T-PHYS-2	A-RUNTIME, A-CODE	Software-only implementation, verified boot	Fault injection (voltage glitching, EM interference)	REQ-BM-EM-001, REQ-BM-RRM-005 (see Annex D)
T-PHYS-3	A-CRYPT, A-MEASURE	Hardware security storage	Hardware security component initialisation attacks	REQ-BM-EM-001, REQ-BM-RRM-005 (see Annex D)
T-PHYS-4	A-CRYPT, A-AUTH	Hardware security storage, verified and measured boot	Side-channel key extraction (power, EM, timing)	REQ-BM-EM-001, REQ-BM-RRM-005 (see Annex D)
T-PHYS-5	A-CODE, A-CONFIG, A-CRYPT	Physical access	Hardware tampering and component modification	REQ-BM-RRM-005 (see Annex D)
T-PHYS-6	A-CODE, A-CONFIG	Fixed storage boot	Flash memory direct manipulation	REQ-BM-RRM-005 (see Annex D)
T-PHYS-7	A-CODE, A-CONFIG	Removable media boot	Removable media substitution	REQ-BM-AMS-001, REQ-BM-INT-003
T-PHYS-8	A-CODE, A-CONFIG	External expansion boot	External storage device compromise	REQ-BM-AMS-001, REQ-BM-INT-003
T-PHYS-9	A-CODE	Internal storage boot	ROM extraction via chip de-packaging	REQ-BM-RRM-005 (see Annex D)
T-PHYS-10	A-CODE, A-CRYPT, A-RUNTIME	Debug interface boot	Debug interface exploitation and authentication bypass	REQ-BM-AC-001, REQ-BM-AMS-002, REQ-BM-AMS-003
T-PHYS-11	A-CRYPT, A-CODE	Debug interface boot	Memory and firmware extraction via debug interfaces	REQ-BM-RRM-005 (see Annex D)

B.4.3.5 T-SUPPLY: Supply chain attacks

Supply chain attacks compromise boot managers during manufacturing, development, distribution, or through systemic trust failures. These threats affect all products regardless of capabilities, though update mechanisms expand the attack surface. Threat exposure increases with RF-SURFACE (update capability extends the supply chain attack surface).

Table B.4.3.5-1: Supply chain attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-SUPPLY-1	A-CODE, A-CRYPT	All products	Manufacturing environment compromise	REQ-BM-RRM-004, [PROC]
T-SUPPLY-2	A-CODE	All products	Third-party component compromise	REQ-BM-SU-006
T-SUPPLY-3	A-CODE	All products	Development tool compromise (compiler attacks)	REQ-BM-RRM-004, [PROC]
T-SUPPLY-4	A-CODE	Update capability	Distribution interception	REQ-BM-SU-004, REQ-BM-SU-006
T-SUPPLY-5	A-CODE, A-CRYPT	Update capability	Update infrastructure compromise	REQ-BM-SU-004
T-SUPPLY-6	A-CRYPT	Verified boot	Default or leaked platform keys	REQ-BM-CP-002, REQ-BM-SU-005
T-SUPPLY-7	A-UPDATE	Update capability	Update tooling compromise	REQ-BM-SU-004
T-SUPPLY-8	A-CRYPT	Verified and measured boot	Trust anchor revocation failures	REQ-BM-INT-008, REQ-BM-INT-019

B.4.3.6 T-NET: Network-based attacks

Network-based attacks exploit network connectivity during boot, update, or configuration processes, providing remote attack surface. These attacks occur before operating system security mechanisms are active. Threat exposure increases with RF-NET (network reachability enables remote exploitation before OS security is active).

Table B.4.3.6-1: Network based attacks

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-NET-1	A-CODE, A-CONFIG, A-CRYPT	Network update capability	Man-in-the-middle and boot image tampering in transit	REQ-BM-DM-002, REQ-BM-INT-005, REQ-BM-INT-014
T-NET-2	A-CODE, A-CONFIG	Network boot	Rogue DHCP/TFTP/HTTP boot servers	REQ-BM-INT-013, REQ-BM-INT-015, REQ-BM-AMS-004
T-NET-3	A-CODE	Network update capability	Network protocol implementation vulnerabilities	REQ-BM-AMS-004, REQ-BM-AMS-005, REQ-BM-AMS-006
T-NET-4	A-CONFIG, A-AUTH	Remote configuration capability	Remote management interface exploitation	REQ-BM-AMS-003, REQ-BM-CP-006
T-NET-5	A-UPDATE	Network update capability	Update denial of service	REQ-BM-IM-001, REQ-BM-AP-004
T-NET-6	A-CONFIG, A-CRYPT, A-MEASURE	Network boot, Measured boot	Unauthorised disclosure of boot configuration, attestation, or cryptographic material via network	REQ-BM-CP-009

B.4.3.7 T-AVAIL: Availability and resilience threats

Availability and resilience threats prevent successful boot completion, cause denial of service, or undermine recovery mechanisms that ensure operational continuity. Threat exposure increases with RF-AVAIL (higher continuity expectations amplify the impact of boot failures) and RF-SURFACE (additional capabilities introduce failure modes).

Table B.4.3.7-1: Resilience threats

Threat ID	Affected Assets	Introduced by	Description	Mitigated by
T-AVAIL-1	A-CODE, A-CONFIG	Update capability	Update process causes denial of service through resource exhaustion or deadlock	REQ-BM-SU-006, REQ-BM-SU-007, REQ-BM-AP-010
T-AVAIL-2	A-RUNTIME, A-CONFIG	Network boot capability	Network boot resource exhaustion or infrastructure unavailability	REQ-BM-AP-003, REQ-BM-AP-004, REQ-BM-AP-005, REQ-BM-AP-009
T-AVAIL-3	A-CONFIG, A-RUNTIME	Configuration management	Boot configuration corruption preventing successful boot	REQ-BM-AP-001, REQ-BM-AP-011
T-AVAIL-4	A-CODE, A-RUNTIME	Multiple boot sources	Primary boot path failure without fallback mechanism	REQ-BM-AP-002, REQ-BM-AP-007
T-AVAIL-5	A-CONFIG, A-CODE	Recovery capability	Recovery mechanism unavailable or corrupted when needed	REQ-BM-AP-012, REQ-BM-AP-016, REQ-BM-SU-007

NOTE 1: Availability threats often interact with integrity threats (T-INTEGRITY) and persistence threats (T-PERSIST).

NOTE 2: Many availability protections are reliability features (timeouts, retries, fallbacks) rather than direct security countermeasures. These requirements support recovery from attacks but are not mitigations in themselves.

B.5 Mapping of risk factors to use cases

Table B.5-1 maps the risk factors defined in clause B.2 to the use cases defined in clause 4.6. Values reflect typical deployments. Actual values for a specific product may vary within the use case.

NOTE: The mappings in this clause are informative. They support the rationale for the use case definitions in clause 4.6 and the security profiles in clause B.6, but do not in themselves impose requirements on the boot manager.

Table B.5-1: Risk factor mapping to use cases

Risk factor	UC-IMM	UC-VER	UC-HW
RF-SURFACE	Minimal	Standard	Standard
RF-NET	Isolated	varies	varies
RF-AVAIL	Flexible	varies	varies
RF-IMPACT	Limited	varies	Critical
NOTE 1: "varies" indicates that the risk factor is determined by the deployment context of the device in which the boot manager is integrated, not by the boot manager product itself. NOTE 2: For UC-HW, RF-AVAIL is marked "varies" to reflect that the use case with hardware-rooted trust can apply to devices that are critical in compromise impact (RF-IMPACT) without requiring the highest availability expectations. Examples include high-assurance personal computing devices and certain medical devices where compromise impact is severe but continuous operation is not required.			

B.6 Mapping of risk factors to security profiles

B.6.1 General

The security profiles are derived from the risk factor analysis in B.5. This clause documents the rationale for associating each use case with its security profile.

B.6.2 UC-IMM: LOW

UC-IMM products have minimal attack surface (RF-SURFACE: Minimal).

The absence of update, network boot, and configuration capabilities eliminates entire threat categories (T-NET, T-AVAIL-1, T-INTEGRITY-3, T-INTEGRITY-20/21/22).

The residual risk is vulnerability accumulation without remediation capability, accepted because the minimal attack surface limits exploitability. Security profile LOW is appropriate.

B.6.3 UC-VER: MEDIUM

UC-VER products have standard attack surface (RF-SURFACE: Standard).

Verified boot detects and refuses execution of bootkit-installed components covered by T-INTEGRITY-1/2 and T-PERSIST-1/2.

Update capability mitigates long-term vulnerability accumulation but introduces T-INTEGRITY-3 and T-SUPPLY-4/5/7. Logging enables post-incident analysis.

Key provisioning supports trust anchor lifecycle management. The balance of mitigation and introduced risk places UC-VER above LOW.

Environmental factors (RF-NET, RF-AVAIL, RF-IMPACT) vary by deployment but do not change the security profile. They are addressed through functional capability requirements triggered when the corresponding capability is present. Security profile MEDIUM is appropriate.

B.6.4 UC-HW: HIGH

UC-HW products add hardware-backed key protection to UC-VER capabilities. This addresses T-INTEGRITY-10 (software key extraction) which UC-VER cannot mitigate.

Hardware security reduces software-only attack surface against the root of trust to physical attacks requiring specialist equipment.

Typical deployments have critical impact expectations (RF-IMPACT: Critical), while availability expectations vary by deployment context. The combination of elevated impact and the hardware assurance described above justifies the highest assessment depth, independent of the availability expectation. Security profile HIGH is appropriate.

Annex C (informative): Product documentation

C.1 Introduction

Table C.1-1 maps the documentation artifacts described in this annex to the requirements they support and the security profiles for which they are applicable.

Table C.1-1: Documentation to requirements mapping

Artifact	Supports	LOW	MEDIUM	HIGH
Threat modelling	REQ-BM-KEV-001 REQ-BM-KEV-002	X	X	X
Boot flow and trust boundaries	REQ-BM-AMS-001	X	X	X
Cryptographic architecture	REQ-BM-CP-001	X	X	X
Memory architecture	REQ-BM-INT-001 REQ-BM-AMS-001	X	X	X
Interface architecture	REQ-BM-AMS-003	X	X	X
Recovery mechanisms	REQ-BM-AP-012		X	X
Isolation boundaries	REQ-BM-INT-001 REQ-BM-AMS-003	X	X	X
Software Bill of Materials (SBOM)	REQ-BM-MON-004	X	X	X
Measurement process	REQ-BM-MON-001		X	X
Environmental assumptions	REQ-BM-INT-004	X	X	
Configuration security	REQ-BM-AC-002		X	X
Update procedures	REQ-BM-SU-004		X	X
Compensating controls	REQ-BM-KEV-003 REQ-BM-SU-003	X	X	X
Physical security	REQ-BM-AMS-003	X	X	X
End-of-life guidance	REQ-BM-CP-007, REQ-BM-CP-010	X	X	X

NOTE: Security testing evidence applies to all profiles (LOW, MEDIUM, HIGH) and is addressed through the conformity assessment regime in clause 6. Testing evidence requirements are described in clause C.7.

C.2 Security design documentation

Threat modelling: Documented threat modelling covering all threats in Annex B applicable to the product's capabilities.

Boot flow and trust boundaries: Documentation of the complete boot flow from power-on to handoff, identifying trust boundaries between stages and the defence-in-depth approach.

Cryptographic architecture: Documentation of the cryptographic key hierarchy, storage locations, and trust model. This includes key provisioning, lifecycle management, and identification of trust anchors.

Memory and interface architecture: Documentation of memory layout, protection mechanisms, and interfaces exposed at each boot stage.

Recovery mechanisms: Documentation of recovery and failsafe mechanisms, including recovery triggers, failsafe boot paths, and user notification.

Isolation boundaries: Documentation of isolation boundaries between boot components, including the mechanisms employed and failure isolation properties.

Software Bill of Materials (SBOM): Software bill of materials aligned with the requirements of prEN 40000-1-3 [i.4], listing all third-party components with version information.

C.3 Physical security documentation

Documentation of physical security requirements when updates are not supported. This covers physical access control requirements, tamper detection expectations, and compensating controls for lack of update capability.

C.4 Integrity mechanism documentation

Measurement process: Documentation of the measurement process for validation, including measurement algorithm, sequence, storage, and validation method.

Environmental assumptions without hardware security: Documentation of environmental assumptions for physical security when hardware security components are not available, including threat model limitations and deployment restrictions.

C.5 Configuration security documentation

Documentation of security impact for each configuration option, including recommended secure values and warnings for insecure configurations. Applies to boot managers with configuration capability (all profiles).

C.6 Vulnerability handling documentation

Update procedures: Documentation of update procedures, or explicit statement that updates are not supported with rationale.

Compensating controls for non-updateable implementations: Documentation of compensating controls and hardware replacement as the update mechanism when updates are not supported.

End-of-life guidance: End-of-life guidance when vulnerabilities cannot be mitigated for non-updateable implementations, including decommissioning procedures.

Vulnerability mitigation scope: Documentation of which vulnerabilities can be addressed via configuration versus code updates.

C.7 Security testing evidence

Evidence demonstrating the boot manager has undergone appropriate security validation.

Signature verification testing: Evidence of testing valid signatures, invalid signatures, expired signatures, and revoked signatures.

Authentication mechanism testing: Evidence of authentication bypass and privilege escalation testing.

Rollback protection testing: Evidence of rollback attempt testing with older valid images and version counter manipulation.

Secure failure mode testing: Evidence of testing behaviour under verification failure, resource exhaustion, and corrupted input.

Input fuzz testing: Evidence of fuzz testing covering parsers for configuration, images, and certificates.

Memory safety testing: Evidence of buffer overflow and integer overflow testing, including static and dynamic analysis.

Code review: Evidence of security-focused code review.

Architecture review: Evidence of architecture review against the documented threat model.

Binary analysis: Evidence of binary analysis of production builds, including verification that debug symbols and test code are removed.

Supply chain security review: Evidence of supply chain security review covering third-party components and build process.

Test coverage metrics: Documentation of test coverage metrics for code, requirements, and threats.

Vulnerability remediation: Evidence of remediation of vulnerabilities identified during testing.

Annex D (informative): Relation to physical attack resistance standards

D.1 Purpose

D.1.0 Introduction

This annex identifies external standards that address physical-attack-related threats to products with digital elements. The physical-attack threats applicable to boot managers are catalogued in clause B.4.3.4 (T-PHYS). The present document does not specify requirements for physical-attack resistance of boot managers.

D.1.1 Component standards: prEN 50765 and prEN 50766

prEN 50765 [i.7] (CLC TC 47X WG 1) and prEN 50766 [i.8] (CLC TC 47X WG 2) specify security requirements for secure microcontrollers and tamper-resistant security processors, respectively. Clause 8.3.15 (TR_RES) of prEN 50765 [i.7] specifies tamper- and attack-resistance requirements at the component level.

A boot manager whose security relies on a secure microcontroller or a tamper-resistant security processor inherits the physical-attack resistance properties of the underlying hardware. The assumption that hardware platform components meet their applicable component standards is recorded in clause B.3.2 (AS-HW-*).

NOTE: Where no such hardware component is present, the physical-attack resistance of the boot manager is determined by the protections offered by the platform on which it executes.

D.1.2 Assessment standards

EN 17927 [i.6] (SESIP), clause 7.7 (Vulnerability Assessment), and ISO/IEC 15408 [i.9] Common Criteria, vulnerability analysis assurance family (AVA_VAN), specify methodology used in the evaluation of physical-attack countermeasures.

D.2 Threat coverage in the wider landscape

For the threats catalogued in clause B.4.3.4 (T-PHYS), the following external standards address corresponding subject matter.

Table D.2-1: Threat coverage

Threat (B.4.3.4)	External standards
Fault injection	prEN 50765 [i.7], Section 8.3.15
Hardware tampering and component modification	prEN 50765 [i.7], Section 8.3.15 prEN 50766 [i.8]
Flash / ROM extraction	prEN 50765 [i.7] and prEN 50766 [i.8]
Debug interface exploitation	prEN 50765 [i.7] and prEN 50766 [i.8]

NOTE: This table is descriptive of the standards landscape at the date of publication of the present document.

D.3 Exploitation mitigation guidance

The present clause provides non-exhaustive informative guidance on exploitation mitigation techniques that may contribute to fulfilling the requirement in clause 5.13 (REQ-BM-EM-001). Exploitation mitigation aims to ensure that, where a vulnerability is successfully exploited, the consequences remain limited and controlled.

The objective is to prevent an attacker from extending their control, escalating privileges, or accessing components, interfaces, or data beyond those directly affected by the original vulnerability. This can be achieved by designing the boot manager so that:

- containment and isolation are applied, for example through isolation or sandboxing of critical components or processes;
- least privilege and resource minimisation are enforced, by reducing the privileges, accessible memory regions, and data available to each function;
- functionality restriction is in place, by limiting the functionality available to compromised components;
- runtime protection and integrity enforcement are implemented, such as Control Flow Integrity (CFI), pointer authentication, or memory tagging, where supported by the target toolchain and platform;
- memory protection mechanisms are applied, such as stack protection mechanisms or Address Space Layout Randomisation (ASLR), where supported by the target toolchain and platform;
- side-channel mitigation measures are applied, such as differential power analysis (DPA) countermeasures, electromagnetic (EM) shielding, tamper detection, and constant-time implementation of cryptographic operations, where the deployment context warrants and the underlying hardware supports them.

The applicability of each measure to a specific boot manager depends on the toolchain, the underlying platform, and the operational environment. The product-facing requirement in clause 5.13 (REQ-BM-EM-001) does not mandate a specific subset of the above techniques but requires the boot manager to implement exploitation mitigation mechanisms appropriate to the boot-manager execution environment and to the documented threat profile.

NOTE: The exploitation-mitigation guidance in this clause corresponds to control GEC-11 of the generic security framework in prEN 40000-1-2 [i.5] (clause 5.2.2(k)). As prEN 40000-1-2 [i.5] is not a normative reference of the present document, the substantive intent of that control is reproduced here as informative guidance and instantiated as the normative requirement REQ-BM-EM-001 in clause 5.13.

Annex E (informative): Relation to NIST SP 800-193

E.1 Purpose

This annex maps requirements to the Protection/Detection/Recovery (PDR) framework from NIST SP 800-193[i.10] Platform Firmware Resiliency Guidelines.

NOTE 1: NIST SP 800-193 [i.10] defines Roots of Trust (RTU, RTD, RTRec) anchored in immutable hardware. These are out of scope, boot managers consume these roots but do not provide them.

NOTE 2: "Section" references throughout this annex refer to sections within NIST SP 800-193 [i.10], not clauses within the present document.

E.2 Protection (NIST SP 800-193, Section 4.2)

E.2.1 Authenticated update mechanism (NIST SP 800-193, Section 4.2.1.1)

Table E.2.1-1: Authenticated update mechanism

Requirement	Description
REQ-BM-SU-001	Update mechanisms enabling vulnerability remediation
REQ-BM-INT-005	Verify authenticity and integrity of update packages
REQ-BM-SU-005	Execute update operations atomically
REQ-BM-SU-004	Store update verification keys in hardware security component
REQ-BM-INT-017	Enforce rollback protection
REQ-BM-INT-018	Store anti-rollback counters, verify signed version metadata

E.2.2 Integrity protection (NIST SP 800-193, Section 4.2.1.2)

Table E.2.2-1: Integrity protection

Requirement	Description
REQ-BM-INT-002	Verify integrity and authenticity using hashes and signatures
REQ-BM-INT-003	Prevent boot with unverified, invalid, expired, or revoked components
REQ-BM-INT-006	Maintain configurable list of approved signatures
REQ-BM-INT-009	Load components into protected memory before verification
REQ-BM-AC-007	Protect trusted certificate stores from unauthorised modification
REQ-BM-CP-010	Overwrite sensitive data after use
REQ-BM-MON-001	Record measurements in a tamper-evident manner

E.2.3 Non-bypassability (NIST SP 800-193, Section 4.2.1.3)

Table E.2.3-1: Non-bypassability

Requirement	Description
REQ-BM-CP-001	Restrict access to cryptographic key material
REQ-BM-AMS-001	Disable debug interfaces, revoke DMA access before handoff
REQ-BM-INT-001	Enforce privilege boundaries between boot stages
REQ-BM-AC-001	Require physical presence for key/certificate changes
REQ-BM-AC-002	Protect boot order and parameters from unauthorised modification
REQ-BM-AC-003	Require authentication before weakening security defaults
REQ-BM-AC-005	Verify cryptographic signature on security policy changes
REQ-BM-INT-003	Prevent boot continuation with unverified components
REQ-BM-INT-008	Require authentication before allowing verification bypass
REQ-BM-SBD-001	Enable cryptographic signature validation by default
REQ-BM-SBD-002	No default passwords, backdoors, or undocumented access
REQ-BM-AMS-003	Disable debug interfaces and test modes in production

E.2.4 Protection of critical data (Section 4.2.4)

Table E.2.4-1: Protection of critical data

Requirement	Description
REQ-BM-INT-011	Protect sensitive configuration with authenticated encryption
REQ-BM-CP-006	Hash stored credentials

E.3 Detection (NIST SP 800-193, Section 4.3)

E.3.1 Detection of corrupted code (NIST SP 800-193, Section 4.3.1)

Table E.3.1-1: Detection of corrupted code

Requirement	Description
REQ-BM-MON-001	Record measurements in tamper-evident manner.
REQ-BM-MON-002	Provide measurement logs with freshness mechanisms
REQ-BM-INT-004	Check component provenance and version consistency
REQ-BM-INT-002	verify boot components against trust anchor before execution

E.3.2 Detection of corrupted critical data (NIST SP 800-193, Section 4.3.2)

Table E.3.2-1: Detection of corrupted critical data

Requirement	Description
REQ-BM-INT-012	Restore secure defaults when corruption detected
REQ-BM-MON-001	Record measurements of boot components and configuration
REQ-BM-MON-001	Record measurements in a tamper-evident manner
REQ-BM-MON-003	Indicate security-relevant failures and state changes

E.3.3 Notification

Table E.3.3-1: Notification

Requirement	Description
REQ-BM-SBD-005	Provide user-visible indication when security is reduced
REQ-BM-AC-006	Indicate when running with modified policies
REQ-BM-AP-016	Indicate errors with sufficient detail
REQ-BM-MON-002	Provide measurement records in verifiable format
REQ-BM-MON-003	Indicate security-relevant failures

E.4 Recovery (NIST SP 800-193, Section 4.4)

E.4.1 Recovery of mutable code (NIST SP 800-193, Section 4.4.1)

Table E.4.1-1: Recovery of mutable code

Requirement	Description
REQ-BM-AP-014	Halt boot, maintain integrity on failure
REQ-BM-AP-015	Enter recovery mode on security violations
REQ-BM-AP-003	Support redundant boot paths
REQ-BM-AP-005	Enforce timeouts for all operations
REQ-BM-AP-010	Require authentication for recovery mode
REQ-BM-AP-013	Protect recovery from unauthorised modification
REQ-BM-AP-016	Maintain recovery accessibility, preserve across updates
REQ-BM-INT-021	Support revocation of compromised keys

E.4.2 Recovery of critical data (NIST SP 800-193, Section 4.4.2)

Table E.4.2-1: Recovery of critical data

Requirement	Description
REQ-BM-SBD-006	Support restoration of secure defaults
REQ-BM-CP-009	Cryptographically erase sensitive data during disposal
REQ-BM-CP-010	Indicate successful sanitisation completion
REQ-BM-AP-001	Support fallback to known-good configuration
REQ-BM-MON-004	Provide version information accessible to OS

E.5 Firmware categories (NIST SP 800-193)

NIST SP 800-193 [i.10] applies PDR to specific firmware categories. The present document applicability:

Table E.5-1: Firmware categories

SP 800-193 [i.10] Category	The present document Applicability
Platform firmware (BIOS/UEFI)	System firmware
Boot firmware	Primary focus
Option ROMs	when boot-relevant
SMM firmware	when boot-relevant
Management controller	boot-related functions only
Network controller	Network boot only

Annexes F to J:
Void

Annex K (normative): Generic cryptographic requirements and assessment

K.1 Cryptography

K.1.1 Requirement

The product's default configuration shall only use cryptographic mechanisms that meet at least one of the following criteria:

1) ACM-listed: the cryptographic mechanism is listed in the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [1];

2) ACM-extended: the cryptographic mechanism is not listed in the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [reference] and meets at least one of the following conditions:

    a) the cryptographic mechanism is listed in clause K.3.2 as an ACM-extended cryptographic mechanism for the specific product function(s), following consideration of the criteria specified in item 2.b);

    b) where the cryptographic mechanism is not listed in clause K.3.2, the cryptographic mechanism meets all the following criteria:

        i) the cryptographic mechanism, or where applicable the ACM-listed cryptographic mechanism on which it is based, is not deprecated per the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [1];

        ii) the cryptographic mechanism has been specified, developed or maintained through a transparent process by a recognised European, international or sector-specific standards development organisation, or by an industry specification organisation accountable for the relevant specification, including ISO, IEC, IETF, IEEE, ETSI, CEN and CENELEC*;* or the cryptographic mechanism is listed as suitable in a publicly available cryptographic catalogue maintained by a recognised national or governmental cybersecurity authority, where the catalogue is maintained under a documented revision and retirement process, including NIST FIPS and Special Publications, NIST CAVP/ACVP validation programmes, BSI TR-02102 series [i.18], ANSSI RGS Annex B [i.19], CCCS ITSP.40.111 [i.20] and ETSI TS 119 312 [i.21].

        iii) the cryptographic mechanism is described in a valid, publicly available and uniquely referenceable specification;

        iv) the cryptographic properties of the cryptographic mechanism are known;

        v) no known weakness affects the cryptographic mechanism in a way that affects its cryptographic properties;

        vi) the cryptographic mechanism is required for a specific set of product functions;

3) Interoperability-based: the cryptographic mechanism is listed in clause K.4.2 as an interoperability-based cryptographic mechanism for specific product function(s) and external specification(s) or external requirement(s).

NOTE 1: The reference to the product's default configuration is intended to define a clear and assessable baseline, corresponding to the configuration in which the product is placed on the market. The product can provide several configurations that fulfil the requirement.

NOTE 2: For products supplied as hardware platforms or components to be configured by an integrator, references in this annex to the product's default configuration refer to the configuration specified for the intended operational use of the product after integration, including the relevant integration assumptions and configuration constraints.

NOTE 3: The applicable ACM catalogue version is identified in the normative references of the present document. The lifecycle treatment of mechanisms affected by ACM deprecation dates, expiry dates, migration conditions or usage limitations is addressed in clause K.2.

NOTE 4: In this clause, an external specification or external requirement means a specification or requirement that is imposed on the product, and which requires the use of a specific cryptographic mechanism for the product to interoperate with an identified system, platform or operational context. An external requirement can be a regulatory requirement, operational constraint, technical interoperability constraint, or platform compatibility requirement.

NOTE 5: Inclusion of a cryptographic mechanism under the interoperability-based criterion does not classify that mechanism as state-of-the-art cryptography.

EXAMPLE: Examples of product functions include secure communication based on TLS 1.3; authenticated encryption of communicated data based on AES-GCM; storage confidentiality based on AES-XTS; firmware signature verification based on Ed25519; and key derivation based on HKDF.

K.1.2 Assessment of product cryptographic configuration

K.1.2.0 General

The assessment in clause K.1.2 verifies the cryptographic mechanisms used in the product's default configuration against clause K.1.1.

For ACM-extended cryptographic mechanisms, the assessment verifies that the cryptographic mechanism is either listed in clause K.3.2 or fulfils the criteria specified in clause K.1.1 item 2.b. For interoperability-based cryptographic mechanisms, the assessment verifies that the cryptographic mechanism is listed in clause K.4.2. The assessment also verifies that the product uses the cryptographic mechanism in accordance with the relevant characteristics, product functions, use cases where applicable, external specifications or external requirements, and conditions specified in the present document.

K.1.2.1 Assessment of ACM-listed cryptographic mechanisms

K.1.2.1.1 Assessment objective

The purpose of this assessment case is to verify that cryptographic mechanisms claimed to comply with clause K.1.1 item 1) and used in the product's default configuration are listed in the ECCG Agreed Cryptographic Mechanisms (ACM) catalogue [1].

K.1.2.1.2 Assessment preparation

- Preconditions for the assessment: The product's default configuration shall be used for the assessment.
- The documentation shall provide a list of cryptographic mechanisms used in the product's default configuration and claimed under clause K.1.1 item 1), including the related product function, relevant parameters where applicable, and the relevant ACM entry.

K.1.2.1.3 Assessment activities

The assessment shall include verification that:

a) each cryptographic mechanism claimed under clause K.1.1 item 1) is identified by reference to a relevant ACM entry;

b) the ACM entry corresponds to the cryptographic mechanism used in the product's default configuration, including relevant parameters where applicable;

c) where the ACM entry includes lifecycle information, such as expiry date, deprecation date, migration condition or usage limitation, this information is reflected in the documentation.

K.1.2.1.4 Assessment evidence

The assessment evidence shall include, as applicable:

a) description of the product's default configuration;

b) list of cryptographic mechanisms claimed under clause K.1.1 item 1);

c) references to the relevant ACM entries;

d) relevant parameters, profiles, cipher suites or configuration constraints, where applicable;

e) relevant lifecycle information from the ACM catalogue, where applicable.

K.1.2.1.5 Assessment verdict

- The verdict PASS shall be assigned if the required evidence has been provided, is complete, is applicable to the assessed configuration, and demonstrates compliance with clause K.1.1 item 1).
- The verdict FAIL shall be assigned otherwise.

K.1.2.2 Assessment of ACM-extended cryptographic mechanisms

K.1.2.2.1 Assessment objective

The purpose of this assessment case is to verify that cryptographic mechanisms claimed to comply with clause K.1.1 item 2) and used in the product's default configuration are either listed as ACM-extended cryptographic mechanisms in clause K.3.2 or fulfil the criteria specified in clause K.1.1 item 2.b, and are used for the claimed product function(s).

K.1.2.2.2 Assessment preparation

- Preconditions for the assessment: The product's default configuration shall be used for the assessment.
- The documentation shall provide a list of cryptographic mechanisms used in the product's default configuration and claimed under clause K.1.1 item 2), including the related product function, use case where applicable, relevant parameters where applicable, and at least one of the following:

  a) the relevant entry in clause K.3.2; or

  b) evidence that the cryptographic mechanism fulfils all applicable criteria in clause K.1.1 item 2.b.

K.1.2.2.3 Assessment activities

The assessment shall include verification that:

a) each cryptographic mechanism claimed under clause K.1.1 item 2) is either listed in clause K.3.2 as an ACM-extended cryptographic mechanism or supported by evidence demonstrating fulfilment of the applicable criteria in clause K.1.1 item 2.b;

b) the cryptographic mechanism used by the product corresponds to the mechanism listed in clause K.3.2 or described in the evidence provided under clause K.1.1 item 2.b;

c) the product function using the cryptographic mechanism, and the use case where applicable, correspond to the related product function and use case specified in clause K.3.2 or justified under clause K.1.1 item 2.b;

d) the relevant characteristics, parameters, profiles, cipher suites or configuration constraints of the cryptographic mechanism correspond to those specified in clause K.3.2 or justified under clause K.1.1 item 2.b, where applicable;

e) where clause K.3.2 or the justification under clause K.1.1 item 2.b defines conditions or limitations for the use of the cryptographic mechanism, these conditions or limitations are reflected in the product's default configuration;

f) where technically feasible, the cryptographic mechanism, parameters and configuration identified in the documentation correspond to the assessed product configuration.

K.1.2.2.4 Assessment evidence

The assessment evidence shall include, as applicable:

a) description of the product's default configuration;

b) list of cryptographic mechanisms claimed under clause K.1.1 item 2);

c) reference to the relevant entry in clause K.3.2, where the mechanism is listed in clause K.3.2;

d) evidence that the cryptographic mechanism fulfils the criteria in clause K.1.1 item 2.b, where compliance with clause K.1.1 item 2 is claimed on the basis of those criteria;

e) identification of the related product function using the cryptographic mechanism and the use case where applicable;

f) relevant characteristics, parameters, profiles, cipher suites or configuration constraints, where applicable;

g) evidence that the cryptographic mechanism is used in accordance with the conditions or limitations specified in clause K.3.2 or in the justification under clause K.1.1 item 2.b, where applicable.

K.1.2.2.5 Assessment verdict

- The verdict PASS shall be assigned if the required evidence has been provided, is complete, is applicable to the assessed configuration, and demonstrates compliance with clause K.1.1 item 2).
- The verdict FAIL shall be assigned otherwise.

K.1.2.3 Assessment of interoperability-based cryptographic mechanisms

K.1.2.3.1 Assessment objective

The purpose of this assessment case is to verify that cryptographic mechanisms claimed to comply with clause K.1.1 item 3) and used in the product's default configuration are listed as interoperability-based cryptographic mechanisms in clause K.4.2 and are used in accordance with the characteristics, product functions, use cases where applicable, external specifications or external requirements, and conditions specified in clause K.4.2.

K.1.2.3.2 Assessment preparation

- Preconditions for the assessment: The product's default configuration shall be used for the assessment.
- The documentation shall provide a list of cryptographic mechanisms used in the product's default configuration and claimed under clause K.1.1 item 3), including the related product function, use case where applicable, relevant parameters where applicable, the external specification or external requirement requiring its use, and the relevant entry in clause K.4.2.

K.1.2.3.3 Assessment activities

The assessment shall include verification that:

a) each cryptographic mechanism claimed under clause K.1.1 item 3) is listed in clause K.4.2 as an interoperability-based cryptographic mechanism;

b) the cryptographic mechanism used by the product corresponds to the cryptographic mechanism listed in clause K.4.2;

c) the product function using the cryptographic mechanism, and the use case where applicable, correspond to the related product function and use case specified in clause K.4.2;

d) the external specification or external requirement requiring the use of the cryptographic mechanism corresponds to the external specification or external requirement specified in clause K.4.2;

e) the relevant characteristics, parameters, profiles, cipher suites or configuration constraints of the cryptographic mechanism correspond to those specified in clause K.4.2, where applicable;

f) the conditions or limitations specified in clause K.4.2 for the use of the cryptographic mechanism are reflected in the product's default configuration;

g) where technically feasible, the cryptographic mechanism, parameters and configuration identified in the documentation correspond to the assessed product configuration.

K.1.2.3.4 Assessment evidence

The assessment evidence shall include, as applicable:

a) description of the product's default configuration;

b) list of cryptographic mechanisms claimed under clause K.1.1 item 3);

c) reference to the relevant entry in clause K.4.2;

d) identification of the related product function using the cryptographic mechanism and the use case where applicable;

e) identification of the external specification or external requirement requiring use of the cryptographic mechanism;

f) relevant characteristics, parameters, profiles, cipher suites or configuration constraints, where applicable;

g) evidence that the cryptographic mechanism is used in accordance with the conditions or limitations specified in clause K.4.2.

K.1.2.3.5 Assessment verdict

- The verdict PASS shall be assigned if the required evidence has been provided, is complete, is applicable to the assessed configuration, and demonstrates compliance with clause K.1.1 item 3).
- The verdict FAIL shall be assigned otherwise.

NOTE 1: The assessment of the external specification or external requirement itself is outside the scope of this assessment case. The assessment verifies that the external specification or external requirement is identified in clause K.4.2 and that the cryptographic mechanism is used by the product for the related product function.

NOTE 2: Documentation review is a minimum assessment activity under the present annex. Functional and/or resistance testing activities can be added by vertical standards where relevant for the product category, product function or cryptographic mechanism.

K.2 Crypto agility

K.2.1 Requirement

Where the product's default configuration uses a cryptographic mechanism for which the ACM catalogue, or the present document, specifies a deprecation date, expiry date, migration condition or usage limitation falling within the intended lifetime of the product, the product shall provide means for addressing the affected cryptographic mechanism by one or more of the following:

a) updating the cryptographic mechanism;

b) using another cryptographic mechanism that complies with clause K.1.1 and is not subject to the relevant deprecation date, expiry date, migration condition or usage limitation;

c) disabling the use of the affected cryptographic mechanism; or

d) limiting the use of the affected product functions accordingly.

EXAMPLE: Where a security mechanism uses a hybrid cryptographic construction, for example a hybrid key-encapsulation mechanism combining a classical and a post-quantum primitive, the lifecycle status of each constituent primitive is relevant to the lifecycle status of the construction. Hybridisation can be used as part of a planned migration strategy where permitted by the ACM catalogue or by the present document. However, hybridisation does not, by itself, extend the recommended usage lifetime of a constituent primitive that has been deprecated.

NOTE 1: Where a hybrid cryptographic construction includes multiple cryptographic primitives, the lifecycle status of each constituent primitive is relevant to the lifecycle status of the construction. Hybridisation is not assumed to mitigate the deprecation of a constituent primitive unless the ACM catalogue or the present document classifies the hybrid construction as suitable for the corresponding product function.

NOTE 2: Crypto agility supports maintaining appropriate cryptographic protection within the intended lifetime of the product, in addition to the capability of updating cryptographic mechanisms on the product in accordance with secure update and secure communication mechanisms.

NOTE 3: Deprecation information from sources other than the ACM catalogue or the present document can be considered as input to vulnerability management or security risk assessment but does not by itself trigger the requirement in clause K.2.1.

NOTE 4: Where the product provides cryptographic capabilities for use by other products, components or layers, the mechanism required by clause K.2.1 can consist of update, configuration, disabling, deprecation or limitation capabilities usable by the integrating product, system operator or user, where applicable.

NOTE 5: The applicability condition of this requirement can express exceptions based on product characteristics, e.g. cryptographic implementation based on immutable hardware co-processor(s) or hardware-based root of trust, or long-lived silicon used in a long-lived non-updateable environment.

K.2.2 Assessment of crypto-agility

K.2.2.1 Assessment objective

The purpose of this assessment case is to verify whether, where the product's default configuration uses a cryptographic mechanism for which the ACM catalogue, or the present document, specifies a deprecation date, expiry date, migration condition or usage limitation falling within the intended lifetime of the product, the product provides means to address the affected cryptographic mechanism in accordance with clause K.2.1.

K.2.2.2 Assessment preparation

- Preconditions for the assessment: The product's default configuration shall be used for the assessment.
- The documentation shall identify:

  a) the intended lifetime of the product;

  b) the cryptographic mechanisms used in the product's default configuration;

  c) the related product function(s) for each cryptographic mechanism;

  d) the lifecycle information applicable to each cryptographic mechanism, where specified by the ACM catalogue or the present document, including any deprecation date, expiry date, migration condition or usage limitation.

K.2.2.3 Assessment activities

The assessment shall include verification that:

a) for each cryptographic mechanism used in the product's default configuration, the lifecycle information specified by the ACM catalogue or the present document has been identified, where such information is specified;

b) where the ACM catalogue or the present document specifies a deprecation date, expiry date, migration condition or usage limitation for a cryptographic mechanism, this information is reflected in the documentation;

c) where a deprecation date, expiry date, migration condition or usage limitation falls within the intended lifetime of the product, the product provides an applicable mechanism for updating the cryptographic mechanism, using another cryptographic mechanism that complies with clause K.1.1 and is not subject to the relevant deprecation date, expiry date, migration condition or usage limitation, disabling the use of the affected cryptographic mechanism, or limiting the use of the affected product functions accordingly;

d) where another cryptographic mechanism is used, that cryptographic mechanism complies with clause K.1.1 and is not subject to the relevant deprecation date, expiry date, migration condition or usage limitation.

K.2.2.4 Assessment evidence

The assessment evidence shall include, as applicable:

a) documentation of the intended lifetime of the product;

b) list of cryptographic mechanisms used in the product's default configuration;

c) identification of the related product function(s) for each cryptographic mechanism;

d) references to the ACM catalogue entry or to the provision of the present document used to determine lifecycle information, where applicable;

e) identification of any deprecation date, expiry date, migration condition or usage limitation falling within the intended lifetime of the product;

f) description of the means provided by the product, such as update of the cryptographic mechanism, use of another cryptographic mechanism that complies with clause K.1.1 and is not subject to the relevant lifecycle constraint, disabling the use of the affected cryptographic mechanism, or limitation of the use of the affected product functions.

K.2.2.5 Assessment verdict

- The verdict PASS shall be assigned if the required evidence has been provided, is complete, is applicable to the assessed configuration, and demonstrates compliance with clause K.2.1.
- The verdict FAIL shall be assigned otherwise.

K.3 ACM-extended cryptographic mechanisms

K.3.1 Requirement

Where the product's default configuration uses ACM-extended cryptographic mechanisms, these mechanisms shall comply with the ACM-extended criterion in clause K.1.1 item 2.

K.3.2 List of ACM-extended cryptographic mechanisms

No ACM-extended cryptographic mechanisms are specified.

K.4 Interoperability-based cryptographic mechanisms

K.4.1 Requirement

Where the product's default configuration uses interoperability-based cryptographic mechanisms, these mechanisms shall be selected from the interoperability-based cryptographic mechanisms listed in clause K.4.2 for the specified product functions and external specifications or external requirements and shall be used in accordance with the conditions or limitations specified in clause K.4.2.

K.4.2 List of interoperability-based cryptographic mechanisms

No interoperability-based cryptographic mechanisms are specified.

History

Version	Date	Status
V0.1.3	June 2026	SRdAP process EV 20260915: 2026-06-17 to 2026-09-15