

Infotainment Systems Product Development

**Operating Systems Security Specification**

---

Version RC3

**DRAFT**

Version Date: August 02, 2017

UNCONTROLLED COPY IF PRINTED

**FORD CONFIDENTIAL**

The copying, distribution and utilization of this document as well as the communication of its contents to others without expressed authorization is prohibited. Offenders will be held liable for payment of damages. All rights reserved in the event of the grant of a patent, utility model or ornamental design registration.

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Created/Modified By** | **Notes** |
| 5/3/2017 | 0.01 | Justin Mendenhall/jmenden2 | Initial Version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.1.1 Code Signing 3](#_Toc491865244)

[1.1.2 Secure Boot 4](#_Toc491865245)

[1.1.3 Code Integrity Check 4](#_Toc491865246)

[1.1.4 Hardware Security Module 5](#_Toc491865247)

[1.1.5 Memory Protection 5](#_Toc491865248)

[1.1.6 Mandatory Access Controls 6](#_Toc491865249)

[1.1.7 Discretionary Access Controls 6](#_Toc491865250)

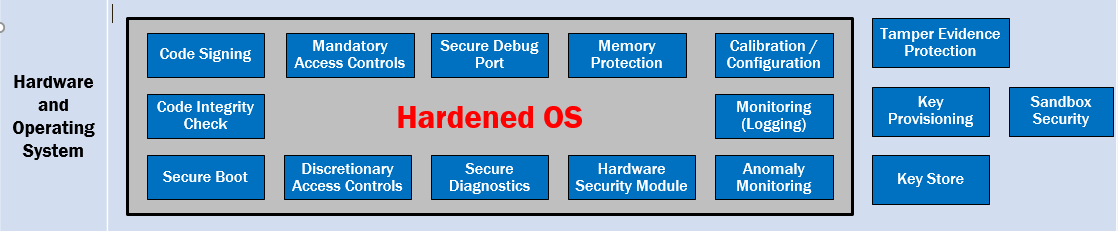
[1.1.8 Secure Debug Port 7](#_Toc491865251)

[1.1.9 Secure Diagnostics 7](#_Toc491865252)

[1.1.10 Calibration and Configuration 8](#_Toc491865253)

[1.1.11 Monitoring and Logging 8](#_Toc491865254)

Security Architecture



### Code Signing

**Title:** Code Signing

**Requirement Text:** All modules shall support code signing and verify signatures for code that is to be installed onto an ECU. Only code signed by Ford shall be accepted for production. Only approved cryptographic libraries shall be used and each instance shall be documented. Each processing domain within an ECU shall have a unique code-signing leaf certificate or Public/Private key pair.

**Definition:** Verification of code, data, scripts and the originator through the use of a chain of trust approved by Ford. Chain of trust mechanism shall use a Ford approved public/private key pair and signature such as: X.509 Certificate PKCS #7, RSA-2048.

Approved cryptographic libraries include but are not limited to: OpenSSL, boringSSL, TomCrypt,

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Verify that only authorized software, applications, and code are installed or executed on the system or module**

**Rationale:** Code Signing verifies that the software or application that is being installed originated from a known and trusted source. Absent a vulnerability that bypasses code signing or permits injection, this feature prevents unauthorized software or applications from being installed on the system.

**Verification Type:** DV

**Applicability: All ECU’s Global**

### Secure Boot

**Title:** Secure Boot

**Requirement Text: :** The system shall have a method to verify that the system software has not been modified at system start-up and that unverified code is not executed. All modules shall verify at start-up that all bootloaders and system software is unmodified.

All keys and certificates required for this operation shall be stored and protected in a tamper resistant IC.

The Operating System shall not execute unverified code. The Operating System may execute verified code prior to the system completing the secure boot process.

**Definition:** Verified code is software or binaries whose signatures have been compared and have been found to be unmodified since its installation.

A tamper resistant IC is an HSM or similar device (Reference HSM section for further details).

**Note:**

Reference NIST 800-147 Appendix A parts 4 and 5, and NIST 800-155.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Verify that the software installed on the system has not been tampered or modified since its installation**

**Rationale:** Secure Boot ensures that the software or data on the hosted the system is authentic at boot-up. If unauthorized software managed to be installed onto the system after a software update, Secure Boot will catch this and prevent the system from booting. Secure Boot requires hardware support to work effectively.

**Verification Type:** DV

**Applicability: All ECU’s with a high-level OS Global**

### Code Integrity Check

**Title:** Code Integrity Check

**Requirement Text:** The OS shall verify that an application, persistent data, and shared libraries are authorized prior to each application execution by comparing the application and data against a known and trusted signature or hash. If the values do not match or the value cannot be trusted, the application shall not execute. This event shall be logged and reported.

**Definition:**

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Prevent tampered or unauthorized applications from executing**

**Rationale:** Applications that have been tampered or modified after system startup will not be caught by secure boot during the current ignition cycle or current system runtime. Checking the application’s signature prior to execution will identify the modification and prevent the application from executing.

This is viewed as the third prong in application integrity verification. The other two prongs are verification at application installation (code signing) and system start (secure boot).

**Verification Type:**  DV or Configuration Review

**Applicability: All Global ECUs with High Level OS**

### Hardware Security Module

**Title:** HSM requirement

**Requirement Text:** Private keys and symmetric keys shall not be directly exposed to the core operating system, applications, or external hardware. A tamper resistant integrated circuit (IC) may be used to protect the keys as necessary if the ECU uses symmetric keys, private keys, or executes secure critical code.

**Definition:** The class of tamper resistant IC required is dependent on how many cryptographic keys are required for a given system. Types of tamper resistant IC are: Hardware Security Module (HSM), Secure Hardware Extension (SHE), Trusted Platform Module (TPM), ARM TrustZone.

Secure critical code is privileged code that needs to execute with full trust and can perform privileged operations.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Secure cryptographic keys and provide HW acceleration of cryptographic functions**

**Rationale:**  Hardware Security Modules and its various subtypes provide hardware based cryptographic acceleration. HSMs essentially provide a mechanism for protecting security keys or cryptographic keys by not directly exposing them. These devices can also be leveraged for secure boot.

**Verification Type:** Hardware Design Review and attestation

**Applicability: All Global ECUs with High Level OS**

### Memory Protection

**Title:** Memory Protections

**Requirement Text:** High level Operating Systems shall implement and enable memory protection mechanisms for the stack and heap of applications. Enabled protections shall be dependent upon support of the Operating System and sign-off from the security team.

**Definition:** Examples of memory protections include but are not limited to: Stack Canaries, Address Space Layout Randomization (ASLR), Position Independent Execution (PIE), Read-Only Relocation (RELRO), No Execute (nX), and Data Execute Prevention (DEP).

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Detect memory modifications and prevent abnormal program execution**

**Rationale:** Memory protection covers a wide area and includes multiple items. Memory protections help detect modification to an application’s stack and/or heap, randomize an application’s address layout, and prevent memory locations from being executable.

**Verification Type:** Design Review and Test Script execution or review of build and configuration system

**Applicability: All Global ECUs with High Level OS**

### Mandatory Access Controls

**Title:** Mandatory Access Controls

**Requirement Text:** Applications shall only be granted the minimum amount of privileges required. Access to resources, including but not limited to system interfaces, data, and message paths, shall be limited and audited. A manifest shall be generated enumerating all applications and their MAC lists. The manifest shall be reviewed by the security team.

**Definition:**

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Restrict access to system interfaces, run applications under Principle of Least Privilege, and minimize data leakage.**

**Rationale:** The system determines and enforces the permissions and access to a given object. The primary utility for enforcement is UID and GID. MAC provides a mechanism to have a rootless system, secure channel, and application sandboxing. Rootless system employs Principle of Least Privilege by where each process or application runs with as few privileges as possible. Secure channel secures the inter-process communication by restricting access to those paths. Additional communication protections can be provided by using cryptographic functions or libraries. Application sandboxing, which also requires appropriate use of Discretionary Access Controls, limits Application A from accessing resources controlled by Application B.

**Verification Type:** Test script and configuration review

**Applicability: All Global ECUs with High Level OS**

### Discretionary Access Controls

**Title:** Discretionary Access Controls

**Requirement Text:** Access to files and executables shall be managed and audited. Permissions in POSIX based systems shall be 744, 755, or hosted in a read-only configuration (e.g. read-only partition). Non-POSIX based systems must use equivalent permissions.

**Definition:** The owner of the file, directory, or process determines the access controls for that object. This protects files and process from unauthorized access and manipulation.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Access to system data, resources, and applications shall be limited, managed, and audited.**

**Rationale:** Applications and users shall not have the ability to access, modify, or execute resources it does not own or is not a member of the owning group. Read access shall only be granted when needed. This prevents data leakage, prevents the system from acting in an unexpected way if configuration files, logs, or other data was modified, and prevents privilege escalation.

**Verification Type:** Test Script and Design Review

**Applicability: All Global ECUs with High Level OS**

### Secure Debug Port

**Title:** Secure Debug Port and/or Services

**Requirement Text:** All debug ports and services shall require a unique authentication credential per ECU for granting enablement or access or be disabled. When an ECU contains multiple micros, each micro shall have unique authentication credentials.

For Hardware Debug Ports, refer to ARL 020667

**Definition:** Debug port access and enablement authentication credentials shall be unique per ECU. Debug port access and enablement events shall be logged and reported.

(Add debug token info)

Unique passwords can be derived from approved random or psuedo-random functions.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: If a debug port or service needs to be accessed or enabled, a secure method is required. The credentials (e.g. password, certificate) shall be unique per each processor.**

**Rationale:** Debug ports should be disabled and removed as this reduces the attack surface. In cases where a debug port needs to be enabled or accessed, the utility, certificate, debug token, or password used to access or enable the debug port, the method shall be unique as this reduces the likelihood of other modules from being compromised through password reuse.

**Verification Type:** DV

**Applicability: All ECUs Global**

### Secure Diagnostics

**Title:** Secure Diagnostics

**Requirement Text: :** Diagnostic sessions shall authenticate the requestor and verify the requestor’s permissions. Each requested secure diagnostic session event shall be logged in an auditable manner.

Diagnostic session shall only be initiated by authenticated and authorized requestor for a given level of access.

**Definition:** Diagnostic session consists of elevated user privileges with the capability to access or modify the ECU or system.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Only authenticated and authorized users shall be able to alter the system configuration**

**Rationale:** Engineers or suppliers require different elevated access to troubleshoot the ECU or module. These credentials and elevated access could be used by malicious actors to modify intended functionality.

**Verification Type:** DV

**Applicability: All ECU’s Global**

### Calibration and Configuration

**Title:** Calibration and Configuration

**Requirement Text:** Write access to calibration and configuration data in memory shall only be permitted to approved accounts, entities, or software update process. All modification events and all attempted modification attempts shall be logged and reported.

**Definition:** Calibration and Configuration data are DIDs, configurable data.

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Ensure the use of authorized calibration and configuration data**

**Rationale:** Using unauthorized calibration and configuration data may cause the system to operate in an unsafe manner, unsecure manner, or enabled unapproved features.

**Verification Type:**

**Applicability: All ECU’s Global**

### Monitoring and Logging

**Title:** Monitoring, Anomaly Detection, and Logging

**Requirement Text**: The system shall monitor and log security related events. Each event shall be written to a system log. The system shall implement an event specific counter, and each event shall increment the counter.

The OS shall have a method to detect unauthorized modification of logs.

Logs shall only be extracted by approved and authorized methods.

A minimum 2MB rotating security log shall be retained.

**Definition:** Security related events include but are not limited to:

* software installation,
* software installation attempts,
* attempts to initiate communication channels,
* attempts to initiate diagnostic sessions,
* attempts to access sensitive or critical data,
* privilege escalation attempts,
* application execution attempts,
* system resets,
* direct memory access attempts,
* events occurring outside of defined behavior

Sensitive or critical data includes certificates, user credentials, passwords, keys, core configuration files.

Approved log extraction methods include: Signed utilities issued by a Ford approved entity (e.g. Hancock)

**Owning CPSC:**

**Requirement Author:** Justin Mendenhall

**Goal: Provide an audit log of security related events**

**Rationale:** Monitoring analyzes various aspects of the system, including but limited to system performance, application execution, access events. Monitoring utilities exist to identify if an application or process exceeds its resource allocation or if it does not have access to its minimum allocation. Monitoring can also identify when an application becomes unresponsive or crashes.

Logging and reporting of this data is crucial to identifying problems or unexpected events. Logs should include a mechanism to identify if an entry is authentic or if it has been modified. This mechanism is generally referred to as Secure Logging. Options should exist to increase or decrease the verbosity of logs. Event counters are useful in the event that the logs roll over, you still know that an event occurred.

**Verification Type:**

**Applicability: All Global ECUs with High Level OS**