24MM Cybersecurity Specification

Version 1.6

Author	TMNA – Cybersecurity-PCG (formerly VST)
Number	24MMSecSpec

Revision History

Version	Date	Description
1.0	August 27, 2021	Initial version for P21MM RFQ.

1.1	January 25th, 2022	• Added section 2.4 Factory Provisioning.
		• Added requirements for TLS 1.2 whitelist (24MM.SEC.PLAT.COM.TLS.10-13) and updated wording for 24MM.SEC.PLAT.COM.TLS.1-2 and 24MM.SEC.APP.COM.EXT.1.
		• Added Qualcomm UIE and FDE support requirements (24MM.SEC.QC.FDE.3-4).
		• Updated wording of 24MM.SEC.PLAT.SB.28,30 regarding timed secure boot.
		• Updated validation requirements for:
		  ○ 24MM.SEC.PLAT.COM.TLS.1-2
		  ○ 24MM.SEC.APP.COM.EXT.1
		  ○ 24MM.SEC.QC.FDE.2
		• Updated Glossary.
		• Removed 24MM.SEC.APP.COM.EXT.2 as it is now covered by 24MM.SEC.APP.COM.EXT.1.
		• Removed 24MM.SEC.PLAT.COM.NFC.1 as NFC is not supported.
		• Removed 24MM.SEC.QC.PER.1-3 as those features are not supported by SA6155/SA8155.

1.2	May 31st, 2022	• Grammatical updates throughout.
		• Removed Verification field and replaced with Evidence field.
		• Added 24MM.SEC.PRJ.PGM.14 to specify CV deadline for delivery of evidences.
		• Updated 24MM.SEC.HW.MEM.1 to broaden acceptable secure element certification methods.
		• Updated Verification for 24MM.SEC.HW.MEM.2.
		• Removed VNC requirements:
		  ○ Updated 24MM.SEC.PRJ.TST.4
		  ○ Removed 24MM.SEC.PLAT.AT.2
		• Clarified 24MM.SEC.PRJ.SW.4 is referring to a “linter.”
		• Updated 24MM.SEC.PLAT.DBG.PROD.2 to make clear the secure debug mechanism is exclusive.
		• Added 24MM.SEC.PLAT.UPD.18-19
		• 24MM.SEC.PLAT.TEE.1: Updated list of use cases for TEE.
		• Update 24MM.SEC.PLAT.COM.TLS.1-2 and 24MM.SEC.APP.COM.EXT.1: Removed TLS 1.2 exception.
		• 24MM.SEC.PLAT.COM.TLS.10-13: Removed TLS
		• MPTCP not supported. Removed requirements (Section 4.13.3).
		• Removed MQTT requirements (Section 4.13.4) as duplicates of Common Communications Spec.
		• Removed 24MM.SEC.PLAT.COM.REV.1, 3- 5, 7, 9-15 as duplicates of Common Communications Spec.
		• Updated 24MM.SEC.PLAT.FDE.1 and 24MM.SEC.QC.FDE.3 to make exception and only encrypt certain boot images with UIE instead of all.

		• Updated 24MM.SEC.PLAT.CRYP.7 to require use of the methods, not implementation of all the methods. • Updated wording of 24MM.SEC.PRJ.PGM.7. • Narrowed language of 24MM.SEC.PLAT.SB.3 to target prevention of U-Boot shells. • Removed 24MM.SEC.QC.SBX.1 as it does not apply to this processor.
1.3	June 18th, 2022	• Updated P21MM to 24MM. • Added 24MM.SEC.APP.COM.WIFI.1. • Restored P21MM.SEC.PLAT.COM.MQTT.1,3. • Restored P21MM.SEC.PLAT.COM.REV.1- 4, 7, 9-15. • Removed 24MM.SEC.PLAT.CRYP.6. • Removed 24MM.SEC.PLAT.STG.6. • Removed P21MM.SEC.PLAT.LOG.GEN.5 • Removed P21MM.SEC.PLAT.LOG.CFG.1-2, 4-5 • Removed P21MM.SEC.PLAT.LOG.ACC.1,3-4 • Removed P21MM.SEC.PLAT.LOG.CI.4 • Removed P21MM.SEC.PLAT.LOG.TR.1-2, 4-5, 7-10 • Removed 24MM.SEC.PLAT.CRT.3. • Removed 24MM.SEC.PLAT.COM.INT.4. • Removed 24MM.SEC.PLAT.PER.5-6. • Removed 24MM.SEC.QC.SOC.1-2. • Changed 24MM.SEC.PLAT.SB.14 from “shall” to “should”. • Changed 24MM.SEC.PLAT.SB.19 from “shall” to “should” and require approval of exceptions. • Changed 24MM.SEC.PLAT.UPD.4 from “shall” to “should”. • Changed 24MM.SEC.PLAT.UPD.9 from “shall” to “should”. • Changed 24MM.SEC.PLAT.DBG.PROD.17 from “shall” to “should”. • Changed 24MM.SEC.PLAT.LOG.CI.2 from “shall” to “should”. • Changed 24MM.SEC.PLAT.COM.TLS.7 from “shall” to “should”. • Changed 24MM.SEC.APP.LOG.5 from “shall” to “should”. • Changed 24MM.SEC.APP.LOG.10 from “shall” to “should”. • Updated 24MM.SEC.PRJ.PGM.13: updated design review list. • Updated 24MM.SEC.PRJ.SW.10: updated wording. • Updated 24MM.SEC.PRJ.SW.11: added example tool. • Updated 24MM.SEC.PRJ.TST.4: Removed test build requirement and moved test facilities list to 24MM.SEC.PLAT.DBG.PROD.4. • Updated 24MM.SEC.HW.SB.1-2: only secure boot and anti-rollback for processors that support those features. • Updated 24MM.SEC.HW.PER.1: removed RTC from trusted peripheral list. • Updated 24MM.SEC.PLAT.CRYP.5: specified RNG. • Updated 24MM.SEC.PLAT.KEY.1: listed exception and that Toyota issues keys for pre-production and production. • Updated 24MM.SEC.PLAT.KEY.2: prefer highest security environment. • Updated 24MM.SEC.PLAT.KEY.4-5: updated reasoning. • Updated 24MM.SEC.PLAT.KEY.7: reference key rotation schedule.

		• Updated 24MM.SEC.PLAT.KEY.9: added note that key import APIs must be accessible from secure world. • Updated 24MM.SEC.PLAT.SB.1: specified secure boot only for supported processors. • Updated 24MM.SEC.PLAT.SB.6: updated wording. • Updated 24MM.SEC.PLAT.SB.15: require only for supported processors. • Updated 24MM.SEC.PLAT.SB.18: added reference to hash strength requirement. • Updated 24MM.SEC.PLAT.UPD.3: require only for SoCs with support for software updates. • Updated 24MM.SEC.PLAT.UPD.7: require only for processors with support for image validation. • Updated 24MM.SEC.PLAT.UPD.12: updated wording. • Updated 24MM.SEC.PLAT.UPD.18: added note. • Updated 24MM.SEC.PLAT.DBG.PROD.1: clarified allowed debug facilities. • Updated 24MM.SEC.PLAT.DBG.PROD.2: clarified “only” specific units. • Updated 24MM.SEC.PLAT.DBG.PROD.3, 6: updated example. • Updated 24MM.SEC.PLAT.DBG.PROD.5: changed HW debug could be re-enabled with debug policy. • Updated 24MM.SEC.PLAT.DBG.PROD.6: update wording. • Updated 24MM.SEC.PLAT.DBG.PROD.6-7: Added reference to 24MM.SEC.PLAT.DBG.PROD.3. • Updated 24MM.SEC.PLAT.DBG.PROD.4: Moved test facilities from 24MM.SEC.PRJ.TST.4. • Updated 24MM.SEC.PLAT.DBG.PROD.10: added “device specific” password. • Updated 24MM.SEC.PLAT.DBG.PROD.12: added “or render unusable”. • Updated 24MM.SEC.PLAT.DBG.PROD.18: clarified wording. • Updated 24MM.SEC.PLAT.DBG.PROD.23: clarified wording. • Updated 24MM.SEC.PLAT.TEE.5: updated wording. • Updated 24MM.SEC.PLAT.OS.SFC.1: clarified for production images. • Updated 24MM.SEC.PLAT.STG.11: updated wording and minimum languages. • Updated 24MM.SEC.PLAT.LOG.GEN.2: removed reference to pre-defined bundles. • Updated 24MM.SEC.PLAT.LOG.CFG.3: Added note regarding default configuration. • Updated 24MM.SEC.PLAT.LOG.ACC.2: added authorized • Updated 24MM.SEC.PLAT.LOG.TR.6: changed from defined period to period approved by Toyota. • Updated 24MM.SEC.PLAT.COM.TLS.4: listed cipher suites. • Updated 24MM.SEC.PLAT.COM.TLS.8-9: updated wording. • Updated 24MM.SEC.PLAT.COM.WIFI.4: updated wording. • Updated 24MM.SEC.APP.COM.BLT.1: clarified expectations. • Updated 24MM.SEC.PLAT.AT.1: changed test images to debug images. • Updated 24MM.SEC.APP.LOG.2: added logging events.

		• Updated 24MM.SEC.APP.LOG.7: clarified encrypted info is OK. • Updated 24MM.SEC.APP.COM.INT.1: clarified exceptions. • Updated 24MM.SEC.APP.HRD.2-7: updated wording. • Updated 24MM.SEC.APP.SBX.4: updated wording. • Updated 24MM.SEC.QC.SB.1: updated wording. • Updated 24MM.SEC.QC.SB.2: clarified wording. • Updated 24MM.SEC.QC.SB.9: removed reference to MSA fuse. • Added 24MM.SEC.PLAT.DBG.PROD.21-23 [DC24-3514] • Added 24MM.SEC.HW.PER.4 [DC24-3514] • Added 24MM.SEC.QC.FDE.5 [DC24-3514]
1.4	July 29th, 2022	• Updated 24MM.SEC.PLAT.TEE.5 : updated wording. [DC24-4072] • Removed 24MM.SEC.PLAT.KEY.1 [DC24-4594] • Updated 24MM.SEC.PLAT.OS.GEN.10 : updated wording. [DC24-6463] • Updated 24MM.SEC.PLAT.DBG.PROD 1-15 17-19 21-23 : added wording that target is SoC only. [DC24-4590] • Updated 24MM.SEC.QC.FDE.1 : updated wording from SA8155P to SA6150/SA8150. [DC24-5209] • Updated 24MM.SEC.PLAT.SB.13 : added wording that the requirement is recommended. [DC24-5116] • Updated 24MM.SEC.APP.SBX.1-2 , 5 , 6 : added wording that the requirement is recommended. [DC24-5739] • Updated 24MM.SEC.HW.PER.4 : added wording that the requirement is recommended. [DC24-6970] • Updated 24MM.SEC.PLAT.PER.2-3 : added wording that the requirements applies only to communication between the SoC and TA100. [DC24-7022] • Updated 24MM.SEC.PLAT.OS.USR.7 : added wording about deletion of files have suid/sgid bits. [DC24-6484] • Updated 24MM.SEC.PLAT.OS.FS.1 : updated wording. [DC24-6900] • Updated 24MM.SEC.PLAT.KEY.5 : updated wording. [DC24-6896] • Updated 24MM.SEC.PLAT.SB.22 : updated wording from toyota to Tier1. [DC24-7025] [DC24-6478] • Updated 24MM.SEC.APP.LOG.2 : added wording that the requirement is recommended. [DC24-6384] • Updated 24MM.SEC.PLAT.LOG.CI.3 : updated wording. [DC24-6379] • Updated 24MM.SEC.QC.FDE.5 : added wording that the requirement is recommended. [DC24-6389] • Updated 24MM.SEC.PLAT.KEY.8 : updated wording. [DC24-3909] • Updated 24MM.SEC.PLAT.COM.FWL.2、24MM.SEC.PLAT.COM.FWL.3 : updated wording. [DC24-6991]
1.5	September 30th, 2022	• Added related document reference to “Common Specification for the Communication”. [AGLSD-2665]
1.6	November 11th, 2022	• Updated 24MM.SEC.QC.FDE.3 : deleted XBL, QSEE/QTEE, QHEE from UIE targets. [AGLSD-3186]

Table of Contents

Related Documents

Issuer	Document Name	Document ID
TMNA PCG	Post 2021 Multimedia Server Cybersecurity Requirements	S_190_NA_EU_AU
TMNA PCG	Cybersecurity Requirements for Mobile Applications	M_190
TMNA PCG	P21MM Design Phase IVI Information Security Threat Analysis and Risk Assessment	P21mmIVISecurityTARA
TMNA CT	191 Specification (TMNA, TME, TMCA) – Post 21MM	Spec 191 P21MM
SCDD	Common Specification for the Communication Interface between In-Car and Out-Car.	
46F	Requirements Specification of Recovery System for Security	SEC-ePF-IDS-REC-REQ-SPEC
46F	Requirements Specification of Center Communication Security	SEC-ePF-CCS-REQ-SPEC
46F	Requirements Specification of Diagnostic Security	SEC-ePF-DGN-REQ_SPEC
46F	Requirements Specification of Cyber Security Event Logging	SEC-ePF-IDS-ANO-REQ-SPEC
46F	Requirements Specification of Host-based IDS for Multi-layered separation	SEC-ePF-IDS-HIM-REQ-SPEC
46F	Requirements Specification of Response Slave of Intrusion Prevention System	SEC-ePF-IDS-IPS-REQ-SPEC
46F	Requirements Specification of QSEvs Creation	SEC-ePF-IDS-QSV-REQ-SPEC
46F	Requirements Specification of Security Information and Event Management system for IDS master	SEC-ePF-IDS-SIEM-REQ-SPEC
46F	Requirements Specification of KMGPKR	SEC-ePF-KMG-PKP-REQ-SPEC
46F	Requirements Specification of KMGPKT	SEC-ePF-KMG-PKP-TST-SPEC
46F	Requirements Specification of Message Filtering	SEC-ePF-MFG-REQ-SPEC
46F	Requirements Specification of Multi-Layered Separation	SEC-ePF-MLS-REQ-SPEC
46F	Requirements of Personal and Privacy Information	SEC-ePF-PPI-REQ-SPEC

46F	Requirements Specification of Standard Reprogramming Security	SEC-ePF-RPR-REQ-SPEC
46F	Requirements Specification of Secure Boot	SEC-ePF-SBT-REQ-SPEC
46F	Terms and Definitions related to Vehicle Cybersecurity and Privacy	SEC-ePF-TRM-GUD-PROC
46F	Requirements Specification to Supplier's Vehicle SIRT	SEC-ePF-VCL-SIRT-REQ-SPEC
46F	Requirements Specification of Common Vulnerability Countermeasure	SEC-ePF-VUL-CMN-REQ-SPEC
46F	Requirements Specification of Vulnerability Countermeasure for ECU	SEC-ePF-VUL-ECU-REQ-SPEC
46F	Requirements Specification of Wireless Communication Security	SEC-ePF-WLS-REQ-SPEC
46F	Requirements of Information Provision for Cybersecurity Examination	SEC-ePF-VCL-EIP-REQ-SPEC
46F	Requirements of Information Provision for Cybersecurity Examination	車載器-センタ間通信標準仕様書_Ver.1.2.pdf

Glossary

Term	Definition
ADAS	Advanced Driver Assistance Systems
AP	Access Point
API	Application Programming Interface
ARP	Address Resolution Protocol
ASLR	Address Space Layout Randomization
BT	Bluetooth
CA	Certificate Authority. A set of root certificates trusted by the device
CAN	Controller Area Network. In-vehicle communication bus
CECU	Central ECU. New ECU for P19ePF that centralizes communications
CMAC	Cipher-based message authentication code
DCM	Data Communications Module. Provides cellular data connectivity
DNS	Domain Name Service
DoIP	Diagnostics over Internet Protocol
DSRC	Dedicated Short Range Communications
ECU	Electronics Control Unit. An electrical component of the vehicle
ETC	Electronic Toll Collection
FTP	File Transfer Protocol
GPS	Global Positioning System
HDMI	High-Definition Multimedia Interface
HLOS	High-Level Operating System. Example: Linux-based OS built using Yocto.
HMAC	Hash-based message authentication code
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
HVAC	Heating, Ventilation, and Air Conditioning
KPI	Key Performance Indicator
MitM	Man-in-the-Middle
MM	Multimedia. Also known as the head unit (HU) or In-Vehicle Infotainment (IVI)
MMU	Memory Management Unit
NFC	Near Field Communication
NIST	National Institute of Standards and Technology
Normal World	Software running outside of the TEE, e.g., Linux operating system
OBD	On-Board Diagnostic
OS	Operating System, e.g., Linux-based Yocto operating system
OS Kernel	Software that implements common services and provides application separation, e.g., Linux
P21MM	Post-21 Multimedia
PCG	Product Cybersecurity Group; formerly VST: Vehicle Security Team
PIE	Position Independent Executable. Required for ASLR.
PKI	Public Key Infrastructure
PPI	Personally Identifiable Information
RFI	Request for Information
RAM	Random Access Memory
ROM	Read-Only Memory
RSE	Rear Seat Entertainment
SD	Secure Digital

SDLC	Secure Development Life Cycle
Secure World	Software running inside of TEE, e.g., OP-TEE and trusted applications.
SoC	System on Chip
Sensitive Information	See section 1.5 Key Terms.
SSH	Secure Shell
SSID	Service Set Identifier
SSL	Secure Sockets Layer
TARA	Threat Analysis and Risk Assessment.
TEE	Trusted Execution Environment, e.g., ARM TrustZone.
USB	Universal Serial Bus
VSOC	Vehicle Security Operations Center

1 Introduction

1.1 Purpose

This document specifies cybersecurity requirements for the 24MM project. These requirements are in addition to the standard cybersecurity specifications produced by TMC 46F. These requirements do not replace the standard cybersecurity specifications. The additional requirements in this document will provide extra implementation details specific to 24MM cybersecurity.

1.2 Scope

The scope of these documents is the 24MM ECU and its links to external entities. These requirements apply globally to all regions.

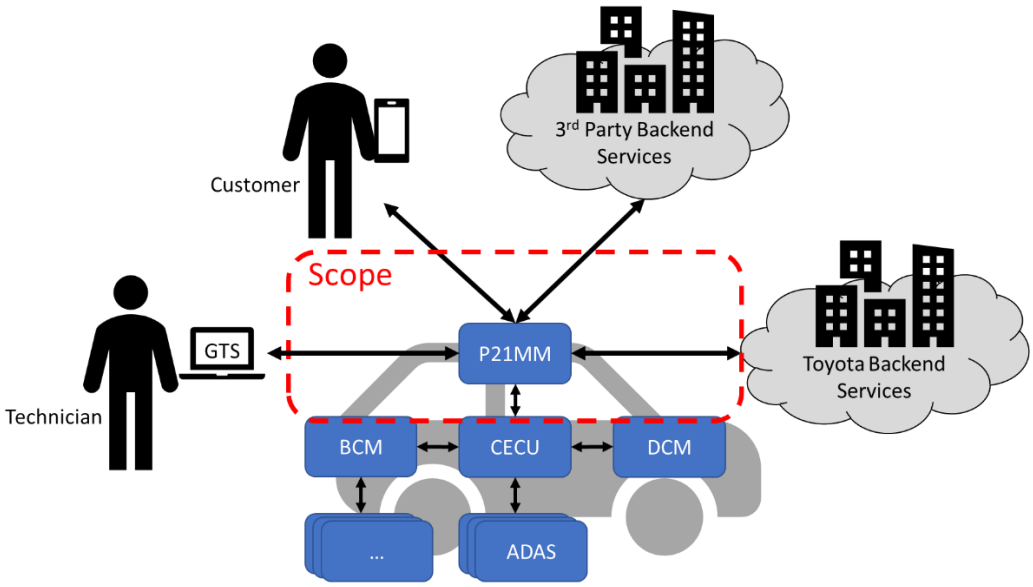

Figure 1 - Scope of this document.

1.3 Structure

This document organizes the cybersecurity requirements according to their level in the ECU architecture. Also, a separate section has been created for requirements specific to the Qualcomm SA6155 / SA8155. There is some overlap in the levels of the architecture, but the goal is that this structure helps the reader find and focus on the requirements most relevant to them depending on their area of responsibility.

The architecture levels used for the document are:

- Project Requirements: Requirements related to the management and operation of the 24MM project.
- Hardware Requirements: Requirements for just the hardware of the ECU.
- Platform Requirements: Requirements on functionality and implementation of cybersecurity features needed to support the complete system, including applications.
- Application Requirements: Requirements for the high-level software that implements the functionality of the ECU.
- SA6155/SA8155 Requirements: Requirements specific to the main application processors chosen for 24MM.

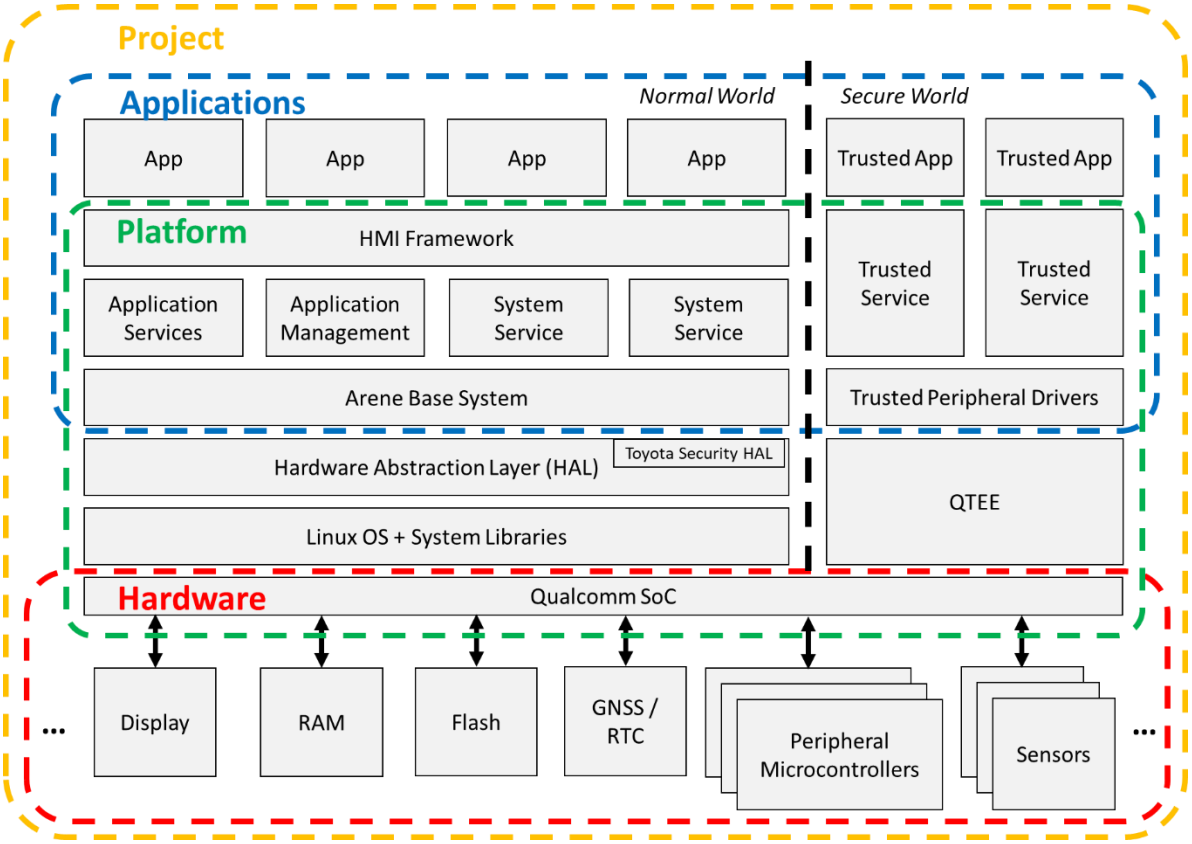

Figure 2 - Rough breakdown of different architecture levels. Note that there is some overlap depending on the specific requirement or function.

Within each architecture section the requirements are organized according to a cybersecurity feature. Many cybersecurity features have requirements in multiple sections, so those subsections are repeated across the architecture sections. From the perspective of a cybersecurity feature, the relationship between the architecture sections is often:

- What hardware support do I need for cybersecurity feature X? Answered in Hardware Requirements section and in SA6155/SA8155 Requirements section.
- How should the cybersecurity feature be implemented X? Answered in Platform Requirements section.
- How should cybersecurity feature X be used? Answered in Application Requirements section.
- Where can I get more background on cybersecurity feature X? Appendix 7.1 Use Cases.

However, the guidelines above do not always apply. Some cybersecurity features may be specific to high-level applications and will only be found in Application Requirements, and similarly with the Platform Requirements.

The appendix contains example use cases that explain the need for certain cybersecurity features.

1.4 Background

The 24MM project is a major update to the Toyota vehicle multimedia system. The update will apply to the head unit as well as the instrument cluster and the rear seat entertainment system. At the same time, major updates to the electrical platform will take place including an updated Data Communications Module (DCM) and a new central ECU. Major software changes may also occur such as the addition of Arene and a new UI system.

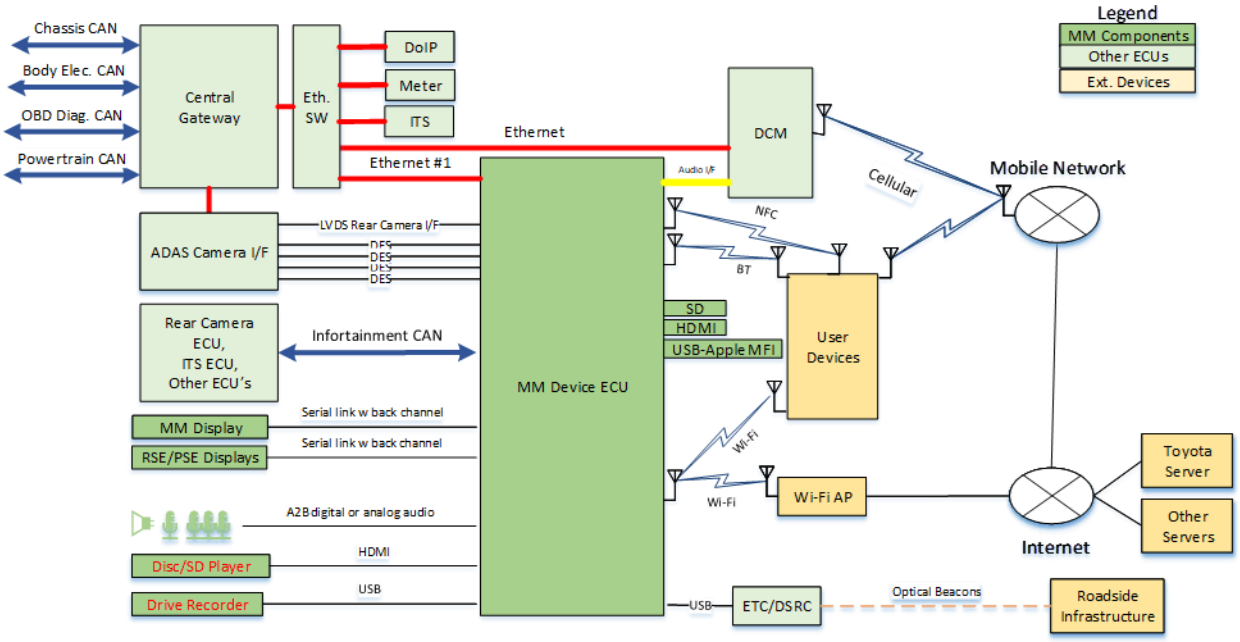

Figure 3 - 24MM High-Level System Architecture

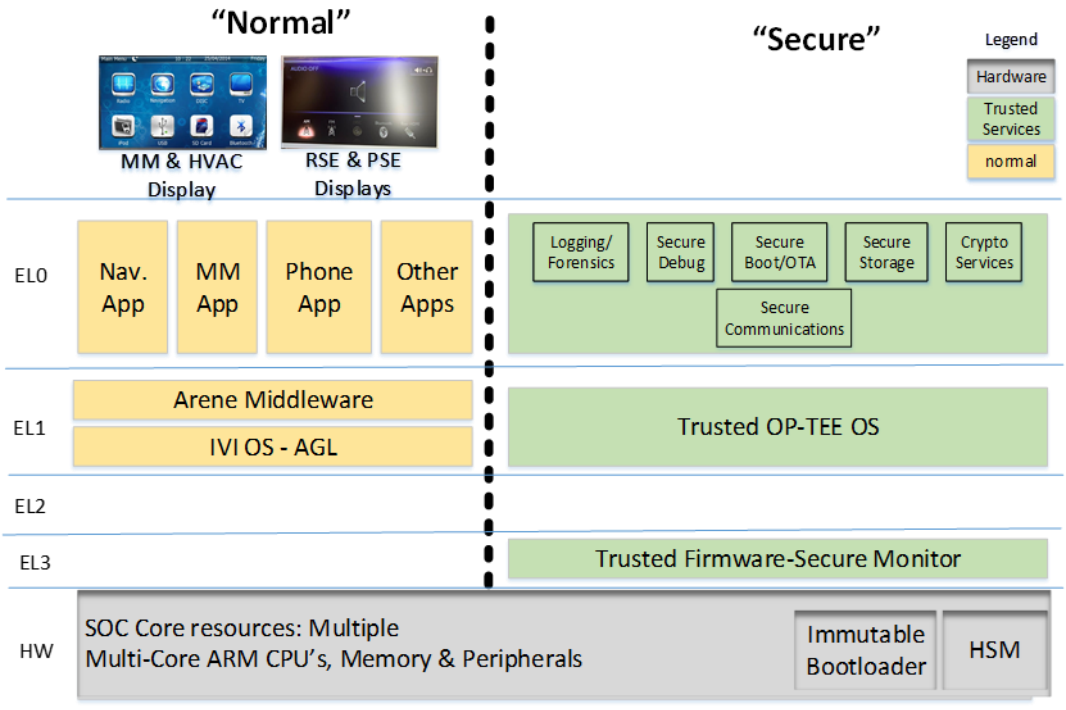

Figure 4 - 24MM High-Level Software Architecture

The addition of these advanced technologies requires equally advanced security measures. These security measures must ensure that each identified threat toward the multimedia system is protected by reliable and redundant security mechanisms that protect against advanced attackers. The TMNA Product Cybersecurity Group (PCG, formerly VST: Vehicle Security Team) has completed a concept-level Threat Analysis and Risk

Assessment (TARA) of 24MM. The TARA details many of the threats in 24MM and suggested security countermeasures.

The design and implementation of cybersecurity and privacy features for 24MM must be a “first order” process. In other words, cybersecurity and privacy must be designed into the 24MM from the beginning, not “bolted on” at the end. This ensures cybersecurity is integrated robustly at all levels of the ECU. IT also allows cybersecurity testing to be conducted early in the development process and avoids significant re-work of features caused by not designing cybersecurity and privacy principles into the system from the beginning.

The “Auto-ISAC Security Development Lifecycle Best Practice Guide” [1] lists some key principles for a secure design that must be considered when designing the 24MM. These principles are:

- Defense in Depth – Layers multiple different controls, so that if one control fails, security remains intact, like the defenses of a castle (e.g., a gateway limiting access to and among ECUs, ECU hardening in case the gateway is compromised, or signed code to prevent unauthorized ECU reprogramming in case ECU hardening is compromised).
- Secure by Default - Usually used in the context of configurations which come from the factory in a secure state but could be modified by the user (e.g., a vehicle's Wi-Fi access point defaults to a secure configuration from the factory but may be modified by the customer). The system may also warn the user if a change in configuration reduces security.
- Open Design (Avoiding Security by Obscurity) - Security is not based upon secrecy of implementation. Instead, use techniques like verification of signed code that are inherently secure even if their use is widely known.
- Fail Securely - When a component fails it defaults to a secure configuration (e.g., if a gateway stops working, then it defaults to denying all traffic rather than allowing all traffic).
- Economy of Mechanism–Designs are kept simple to shrink attack surface (e.g., a component implements only required protocols and unused functionality is removed).
- Separation of Duties (Functions) - This involves separating functional duties (e.g., Wi-Fi is not included on a steering module).
- Separation of Privileges – A single privilege is broken into multiple pieces (e.g., workflow with multiple approvers is followed to provision a certain vehicle component).
- Principle of Least Privilege - Limiting privileges to the least necessary to perform a required operation (e.g., internet facing entities have no CAN bus privileges, services that do not require root access do not run under the root account).
- Zero Trust Model – Since most components can be individually compromised, trust is limited (e.g., validate inputs).
- Continuous Update – A secure system deployed today may not remain secure forever. Note that other principals like defense in depth, when applied well, may mitigate, or reduce the impact of components found to have vulnerabilities later.
- End-to-End Security - This relates to safeguarding information in an information system from point of origin to point of destination.

Customer privacy is also a critical concern for the 24MM and other ECUs. “Privacy by Design: The 7 Foundational Principles” [2] provides a list of principles that must be considered when designing the 24MM. These principles are:

1. **Proactive not Reactive; Preventative not Remedial.** The Privacy by Design (PbD) approach is characterized by proactive rather than reactive measures. It anticipates and prevents privacy invasive events before they happen. PbD does not wait for privacy risks to materialize, nor does it offer remedies for resolving privacy infractions once they have occurred — it aims to prevent them from occurring. In short, PbD comes before-the-fact, not after.
2. **Privacy as the Default Setting**. We can all be certain of one thing — the default rules! PbD seeks to deliver the maximum degree of privacy by ensuring that personal data are automatically

protected in any given IT system or business practice. If an individual does nothing, their privacy remains intact. No action is required on the part of the individual to protect their privacy — it is built into the system, by default.

3. **Privacy Embedded into Design**. Privacy by Design is embedded into the design and architecture of IT systems and business practices. It is not bolted on as an add-on, after the fact. The result is that privacy becomes an essential component of the core functionality being delivered. Privacy is integral to the system, without diminishing functionality.
4. **Full Functionality — Positive-Sum, not Zero-Sum**. PbD seeks to accommodate all legitimate interests and objectives in a positive-sum "win-win" manner, not through a dated, zero-sum approach, where unnecessary trade-offs are made. Privacy by Design avoids the pretense of false dichotomies, such as privacy vs. security, demonstrating that it is possible to have both.
5. **End-to-End Security — Full Lifecycle Protection**. PbD, having been embedded into the system prior to the first element of information being collected, extends securely throughout the entire lifecycle of the data involved — strong security measures are essential to privacy, from start to finish. This ensures that all data are securely retained, and then securely destroyed at the end of the process, in a timely fashion. Thus, PbD ensures cradle to grave, secure lifecycle management of information, end-to-end.
6. **Visibility and Transparency — Keep it Open**. PbD seeks to assure all stakeholders that whatever the business practice or technology involved, it is in fact, operating according to the stated promises and objectives, subject to independent verification. Its component parts and operations remain visible and transparent, to users and providers alike. Remember, trust but verify.
7. **Respect for User Privacy — Keep it User-Centric**. PbD requires architects and operators to keep the interests of the individual uppermost by offering such measures as strong privacy defaults, appropriate notice, and empowering user-friendly options. Keep it user centric.

1.5 Key Terms

These terms are used throughout the document and their definition expanded here.

Software	This includes the first image loaded by the immutable ROM of the SoC and extends to all application software, including Linux user space applications and libraries. It also includes software images for peripherals or microcontrollers sometimes referred to as "firmware." Finally, shell scripts and scripts for interpreted languages are also included.
Static Data	This is data that does not change between updates. This includes OS configuration files, Mandatory Access Control policies, read-only databases, device tree blobs, etc.
Sensitive Information	Data whose confidentiality must be ensured. This includes: • Personal Information and Sensitive Personal Information as defined by "191 Specification (TMNA, TME, TMCA) - Post 21MM" • Financial information • Cryptographic keys • Wi-Fi Passphrase • Bluetooth Link Key • Plaintext passwords (not password hashes properly salted). • Authentication tokens
Toyota Servers	These are servers controlled and operated by Toyota. This would include backend services such as TSC, TSP, CTP, etc.

3rd Party Servers	These are servers that Toyota has influence over but does not directly control. This would typically be servers operated by suppliers that Toyota has a contractual relationship with such as navigation or music streaming providers.

1.6 General Guidance

The requirements are written generically as they apply to any ECU with a High-Level Operating System (HLOS). For this document, interpret “ECU” as referring to the 24MM.

Unless otherwise stated, all requirements are intended for the production version of the ECU hardware and software. The Secure Debug sections have requirements specific to debug versions of the ECU hardware and software, and to debugging production ECUs. A “production” version of the ECU is the final hardware and software delivered to customers. An initial “production” version of the ECU must be delivered at the CV timing with all cybersecurity features implemented so cybersecurity testing can be conducted (see 24MM.SEC.PRJ.PGM.5).

References to milestones such as CV refer to the earliest version of the milestone across the different regions. For example, if the CV date for the initial 24MM North American vehicle is earlier than the CV date for the initial Japan vehicle, the CV milestone is the CV for the North America vehicle.

If a requirement cannot be met for any reason, the supplier must seek an exception to the requirement from Toyota. The supplier must provide the reason the requirement cannot be met and a remediation plan that addresses the open security issues.

If a requirement prevents or limits a desired use case, please consult with the security team to understand if the use case goals can still be met by clarifying cybersecurity requirements or by achieving the same level of security through alternate means.

Suppliers should understand all sections of this document. However, depending on your role, you may want to focus on certain areas:

- Hardware engineers: Focus on hardware requirements and SA6155 / SA8155 requirements.
- Platform / BSP engineers: Focus on platform requirements.
- Engineers for functional applications / 3rd party integrations: Focus on application requirements.

Everyone should also be familiar with the Project Requirements section to understand expected evidence and software development requirements.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2 Project Requirements

2.1 Program Management

Goal: Cybersecurity is integrated at all levels of the ECU from the beginning of development. Prevent and catch problems early during development and enable Toyota to perform thorough testing.

ID	24MM.SEC.PRJ.PGM.1
Requirement	The supplier's activities shall be compliant with ISO/SAE 21434. Note: Tier-2+ suppliers must also comply with ISO/SAE 21434 engineering process and industry best practices for cybersecurity engineering of products
Reasoning	Toyota must develop new components according to ISO/SAE 21434.
Evidence	A summary of all ISO/SAE 21434 cybersecurity activities and deliver the corresponding work products to Toyota for review (e.g., Appendix A.2 Overview of cybersecurity activities and work products)
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.2
Requirement	Suppliers shall provide a cybersecurity record of capability according to Section 7.4.1 of ISO/SAE 21434. ISO/SAE 21434 lists components of the supplier record of capability: a) Evidence of the organization's capability concerning cybersecurity (e.g., cybersecurity best practices from development, post-development, governance, quality, and information security) b) evidence of continuous cybersecurity activities and cybersecurity incident response c) summary of previous cybersecurity assessment reports
Reasoning	Supplier record of capability needed to meet ISO/SAE 21434 RQ-07-01 which requires evaluation of supplier's capability to develop and perform post-development activities.
Evidence	Evidence described in requirement.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.3
Requirement	Suppliers shall establish, document, and implement initiatives in-line with commonly accepted industry standards and practices to build security into the software development process. Such initiatives that build security within all phases of the development lifecycle can include but are not limited to: • Risk assessment/Threat modeling process • Documented security requirements • Secure coding guidelines and checklists

	• Secure design/architecture review • Source code review • Security testing All initiatives and documentation shall be made available to Toyota as needed.
Reasoning	Secure development practices must be established at the beginning of the program to ensure cybersecurity is designed into the ECU holistically.
Evidence	A summary of all industry standards and best practices initiatives and documentation that describes how security is being built into the software development process.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.PGM.4
Requirement	Suppliers shall designate a security specialist as a manager responsible for conducting risk assessments
Reasoning	Ensure efficient resolution of cybersecurity decisions by clearly identifying supplier person responsible for providing Toyota needed risk assessments.
Evidence	Name(s) and contact information of the designated security specialist(s) Toyota can coordinate with for conducting and reviewing supplier risk assessments.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.5
Requirement	Suppliers shall prioritize development of cybersecurity so that all cybersecurity functions and requirements are completed by the Confirmed Vehicle (CV) milestone. This includes secure boot, Mandatory Access Control policies, secure update, secure storage, etc.
Reasoning	Completing cybersecurity features by CV provides enough time to perform cybersecurity requirement validation. Ensures security features are not developed late in the program where unexpected side effects of security features could lead to workarounds that lessen the security of the device (e.g., MAC policies are more permissive than necessary due to insufficient time to investigate problems). Ensures all security issues identified after CV timing are legitimate issues (not planned to be fixed by an unreleased security feature) and there is enough time to fix them.
Evidence	Development schedules that include all cybersecurity features.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.6
Requirement	Suppliers shall provide a feature rollout plan (FROP) for all cybersecurity features within

	three months of the start of development. Any updates to the feature schedule shall be communicated to Toyota immediately. For each cybersecurity feature, the FROP shall list the software release milestone targeted for completion of the feature, along with the dates applicable for North America.
Reasoning	Vulnerability testing is more efficient when it is known which security features are complete and available for testing.
Evidence	Supplier shall provide a FROP that includes all cybersecurity features.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.7
Requirement	The supplier shall only use software that receives support by one of the following methods: • Commercial software supported by a company. • Open-source software with an active community of open-source developers. • Direct support by the supplier. Note: Support means having the knowledge and capability to add features as requested and to fix security vulnerabilities as they are discovered
Reasoning	Old projects without maintainers may have security issues that will never be fixed due to the lack of maintenance.
Evidence	SBOM shall list entity providing support.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.PGM.8
Requirement	The supplier shall not include different implementations of the same functionality. Example: Most applications use the Linux TCP/IP stack, but a few applications perform TCP/IP using a separate library. This is not allowed. Note: Some applications depend on specific libraries to implement a given feature. In situations where different applications require different libraries to implement the same functionality, this may be permissible with the approval of Toyota. However, most applications will support a common implementation of their dependent features.
Reasoning	Reduce opportunity for vulnerabilities by minimizing software that is used.
Evidence	SBOM
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.PGM.9
Requirement	Suppliers shall provide Toyota a list of all 3rd party and open-source libraries and applications used as part of the ECU along with their versions. Otherwise known as a Software Bill of Materials (SBOM). Note: This includes libraries that are statically compiled into another piece of software.
Reasoning	When vulnerabilities are disclosed in outside software, Toyota can quickly determine if the ECU is affected.
Evidence	SBOM and unencrypted flash images.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.10
Requirement	Suppliers shall provide Toyota with a list of all hardware components and their part numbers used as part of the ECU. Otherwise known as a Hardware Bill of Materials (HBOM).
Reasoning	When hardware vulnerabilities (particularly in programmable chips) are disclosed for any product, even non-automotive products, Toyota can quickly determine if the ECU is affected.
Evidence	HBOM.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.11
Requirement	All Bill of Materials (HBOM and SBOM) documents shall be machine-readable. Preferred format is CycloneDX followed by SPDX.
Reasoning	Machine-readability is necessary for automated verification systems.
Evidence	SBOM and HBOM.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.12
Requirement	An accurate hardware and software bill of materials shall be included with every release or update delivered to Toyota.
Reasoning	The Bill of Materials (BOM) shall be updated to reflect changes so that Toyota can quickly and easily identify component changes between releases.

Evidence	SBOM, HBOM, unencrypted flash images.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.13
Requirement	Suppliers shall provide Toyota (including TMNA PCG) design reviews of critical cybersecurity functions, including: • Secure Boot • Secure Updates • Secure Debug • Trusted Execution Environment • Full Disk Encryption • Logging • Application Sandboxing Framework • Key Management • Secure Communications • Certificate revocation and management • Peripheral binding implementation Design reviews shall consist of live presentations and question-and-answer sessions. Design reviews of cybersecurity functions shall be conducted as part of any program "gates" where other feature design reviews take place. Cybersecurity design reviews shall be conducted before development of the functions and shall meet Toyota approval before suppliers proceed.
Reasoning	Toyota gains a complete understanding of critical cybersecurity functions that form the security foundation of the ECU.
Evidence	Design review materials (presentations, datasheets, etc.)
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.PGM.14
Requirement	Evidences for each requirement shall be supplied by the CV milestone.
Reasoning	Gives Toyota time to review materials and request changes.
Evidence	N/A
Threat Scenarios	N/A

2.2 Software Development

Goal: Developers identify and fix potential security issues early in the development process before vulnerability testing and deployment.

ID	24MM.SEC.PRJ.SW.1
Requirement	Supplier developed software shall comply with the applicable SEI (Software Engineering Institute) CERT Coding Standard.
Reasoning	SEI CERT Coding Standard contains many of the best practices and requirements that can be used to avoid implementing vulnerable software
Evidence	Document describing how SEI CERT standards are utilized as part of the development practices and code reviews.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.2
Requirement	Supplier developed software shall be written in compliance with a coding style guide that documents indentation, brace style, naming convention, etc.
Reasoning	Following a style guide makes the code easier to read and easier to identify problems during code review
Evidence	Coding style specification used for software development.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.3
Requirement	Supplier shall conduct internal code reviews for all supplier developed software. Code reviews shall look for logical errors or flaws that could lead to a cybersecurity vulnerability and shall verify that software cybersecurity requirements are implemented.
Reasoning	Code reviews are a fundamental activity to ensure high code quality and identify cybersecurity issues before the code is released.
Evidence	Dates code reviews were completed for each software component.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.4
Requirement	Supplier developed ECU software shall be scanned for violations of the coding style by a static analysis tool (i.e., “linter”). Any violations shall be fixed before the code is integrated into the main codebase.
Reasoning	Enforcing the coding style as part of the build process allows code reviewers to focus on identifying security vulnerabilities and other flaws and avoid spending time on style issues.

Evidence	Linter tool configuration file.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.5
Requirement	Supplier developed ECU software shall be scanned by a static analysis tool that looks for cybersecurity vulnerabilities. Any violations shall be resolved before the software release.
Reasoning	Enforcing vulnerability scanning tools as part of the build process fixes some issues early and allows code reviewers to focus on identifying harder to detect issues.
Evidence	Vulnerability scan configuration file.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.6
Requirement	All violations detected by the static analysis tool shall be resolved before the software containing the violation is released. Resolved means either documenting why a fix is not necessary (e.g., false positive) or fixing the software to remove the violation.
Reasoning	Ensure all known vulnerabilities are resolved as soon as they are identified to avoid accumulating a backlog of known vulnerabilities
Evidence	Summary report of static analysis results that confirms no violations are left unresolved.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.7
Requirement	Static analysis tool configuration shall be shared with Toyota for review. Configuration changes requested by Toyota shall be applied
Reasoning	Ensure static analysis is thorough and is not misconfigured so that vulnerabilities are missed.
Evidence	Updated static analysis tool configuration files after applying changes requested by Toyota.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.8
Requirement	Software provided by the supplier shall be compiled with warnings enabled that could catch programmer mistakes such as:

	• Illogical or unusual comparisons. • Misleading indentation. • Pointer vs. array misuse. The list of options that shall be enabled are: • -Wall • -Wextra • -Wconversion • -Wsign-conversion
Reasoning	Emitting extra warnings can catch subtle errors that may lead to security issues
Evidence	List of compiler flags used.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.9
Requirement	Software provide by the supplier shall fail to compile if any warnings are detected. Note: This feature can be enabled using the -Werror flag.
Reasoning	Halting the build on warnings ensures potential security issues are resolved early in development and not ignored or hidden.
Evidence	List of compiler flags used.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.10
Requirement	The supplier should update all supplied software to the latest stable version with long-term support at most two months prior to pre-CV timing. This includes: • Open-source software • Distribution software like the boot loader and operating system kernel. • Hardware-vendor firmware and microcode. • Supplier provided platform and application software. Any outdated software that is used shall be documented by the suppliers and reviewed and approved by Toyota.
Reasoning	Old software with known vulnerabilities is a primary mechanism for an adversary to escalate privilege and do harm. Minimize these risks by updating software, including the kernel and bootloader, to the latest version.
Evidence	SBOM shall list version of software used.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.11
Requirement	The supplier shall confirm all known vulnerabilities for each software component of the SBOM are fixed. Vulnerability databases shall be consulted to identify known vulnerabilities, including the Common Vulnerabilities and Exposures (CVE) system operated by MITRE, Japan Vulnerability Notes (JVN), and GitHub Advisory Database. Note: Where available, tools that provide automatic notification of vulnerable libraries/dependencies should be used, e.g., GitHub Dependabot, Yocto cve-check, etc.
Reasoning	Must ensure all known issues are fixed in software that is deployed.
Evidence	List of CVEs for each software component and “yes/no” status if that CVE is fixed in the version of the software component that is used.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.SW.12
Requirement	The supplier shall protect the development and production environments (e.g., developer endpoints, development tools, libraries, source code, documentation, test data, debug, and diagnostic hardware) from unauthorized access.
Reasoning	Development versions of hardware and software can be used to glean sensitive design information that may enable an attacker to find vulnerabilities in production systems.
Evidence	Documentation on the security of their development and production environments provided within three months of the start of development.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39, TS74

ID	24MM.SEC.PRJ.SW.13
Requirement	Log level guidance documents shall be written that software developers can use to determine the proper log severity level of each log message. The same guidance document can apply to multiple software components, and multiple guidance documents can be written for distinct types or groupings of software. The severities shall follow or map to the syslog severities provided below: • Emergency: system is unusable • Alert: action must be taken immediately • Critical: critical conditions • Error: error conditions • Warning: warning conditions • Notice: normal but significant condition • Informational: informational messages
Reasoning	It is important to ensure consistent use of log severity levels to enable proper identification of important log messages during real-time analysis and forensics.
Evidence	Log level guidance provided within three months of the start of development.

Threat Scenarios	TS37

2.3 Vulnerability Testing

Goal: Suppliers provide Toyota with all necessary equipment and information to conduct thorough and timely vulnerability testing of the ECU.

ID	24MM.SEC.PRJ.TST.1
Requirement	Supplier shall provide Toyota with all materials and documentation necessary to reproduce all cybersecurity requirements verification testing and penetration testing conducted by the supplier. Materials and documentation shall be provided by the CV milestone.
Reasoning	Enables Toyota to verify all testing performed by the supplier.
Evidence	Evidence described in requirement.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.2
Requirement	Suppliers shall provide all necessary wiring harnesses, test hardware, and documentation to enable vulnerability testing of the device. This includes artifacts for: • Diagnostic modes. • Internal debug interfaces such as JTAG, serial ports, Ethernet, etc. • Internal diagnostic or analysis tools. • Test environments and emulators. • System internals documentation necessary for validation, to be specified by Toyota.
Reasoning	Access to all debug, diagnostic, and analysis software and hardware enables the vulnerability testing team to determine if those interfaces are appropriately disabled and secured. They can also be used to ensure there are no other issues in other parts of the system.
Evidence	Evidence described in requirement.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.3
Requirement	Suppliers shall provide all hardware and software tools necessary to mock or simulate external devices to conduct testing of the ECU.
Reasoning	Often suppliers create simple utilities or devices to mimic an external device so the ECU can be fully tested. Toyota also needs access to these utilities or devices to conduct their own testing.

Evidence	Evidence described in requirement.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.4
Requirement	Every software release provided to Toyota during development, both pre-CV and post-CV, shall include a “production” version. Every production version of software shall: • Remove or disable all debug and development facilities such as serial consoles, software debuggers, etc. • Use production configurations of all features and facilities. • Activate all implemented security features including secure boot, Mandatory Access Controls, secure updates, secure storage, etc.
Reasoning	Vulnerability testing is most valuable on production versions of software, so any identified issues are known to be valid. Avoids wasting time documenting and reporting issues that are present only for debug and development purposes
Evidence	Production software builds.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.5
Requirement	Toyota shall be provided with the necessary documentation and tools such that it can install updates independently from the supplier during development and testing.
Reasoning	Waiting for units to be upgraded by a supplier is inefficient and delays test activities.
Evidence	Evidence described in requirement.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.6
Requirement	Toyota shall be provided full update packages at each update. The full update package shall allow the unit to be upgraded to the most recent version no matter the current version of the unit.
Reasoning	Tracking many differential updates between specific versions during development is error prone and can lead to broken units.
Evidence	Full update packages.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.7
Requirement	Suppliers shall provide Toyota with a full, unencrypted image of non-volatile storage for each version of software and for every peripheral and processor.
Reasoning	Access to full images allow vulnerability testing and security requirements validation to be more effective
Evidence	Unencrypted flash images.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.8
Requirement	Every update delivered to Toyota shall include a change log that documents and describes the changes made to any security features since the last update.
Reasoning	Enables vulnerability testing efforts to focus on new functions and features, increasing the efficiency and completeness of testing activities.
Evidence	Change log for each update with any changes to security functions listed.
Threat Scenarios	N/A

ID	24MM.SEC.PRJ.TST.9
Requirement	TOYOTA shall be given access to review all secure boot implementations on the ECU. This includes detailed design documentation and the source code for secure boot functionality of all boot stages, including SoC ROM software. Information and access shall be granted prior to the pre-CV milestone of ECU. Note: TOYOTA can support source code reviews at supplier facilities if source code cannot be shared directly with TOYOTA.
Reasoning	The secure implementation of the secure boot software is critical, particularly for the SoC ROM software since it forms the root of trust for secure boot and cannot be updated. The ROM shall be analyzed to ensure it is secure and that no vulnerabilities are present that could be exploited by an attacker.
Evidence	Evidence described in requirement.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PRJ.TST.10
Requirement	TOYOTA shall be given access to review all TEE implementations on the SoC. This includes design documentation and source code of TEE functionality and trusted applications. Information and access shall be granted prior to the pre-CV milestone of ECU.

	Note: TOYOTA can support source code reviews at supplier facilities if source code cannot be shared directly with TOYOTA.
Reasoning	The security of the TEE is especially important as it forms the root of trust for secure software and operations. The TEE shall be analyzed to ensure it is secure and that no vulnerabilities are present that could be exploited by an attacker.
Evidence	Evidence described in requirement.
Threat Scenarios	TS1-5

ID	24MM.SEC.PRJ.TST.11
Requirement	All external and in-vehicle interfaces of the ECU shall be fuzz tested. This includes wired and wireless interfaces, such as: • Wi-Fi • Bluetooth • NFC • CAN • Ethernet • USB
Reasoning	Ensure parsing of messages is secured against buffer overflow attacks, integer overflow attacks, etc.
Evidence	Fuzz test report showing no detected problems.
Threat Scenarios	TS47-TS57, TS59-67

ID	24MM.SEC.PRJ.TST.12
Requirement	Fuzz tests shall be conducted both before and after authentication, as applicable. Example: Bluetooth fuzzing shall be conducted on both an unpaired and a paired connection.
Reasoning	Vulnerabilities only available after authentication can still be used by an attacker connecting with a compromised device or to gather information about the device for reverse-engineering.
Evidence	Fuzz test report showing no detected problems.
Threat Scenarios	TS47-TS57, TS59-67

ID	24MM.SEC.PRJ.TST.13
Requirement	Fuzz tests shall incorporate the structure of the associated protocols when creating the test data to increase the coverage of the test. Fuzz tests shall not be “dumb” where fuzzing is only performed using unstructured, random data.

Reasoning	Vulnerabilities only available after authentication can still be used by an attacker connecting with a compromised device or to gather information about the device for reverse-engineering.
Evidence	Fuzz test configuration.
Threat Scenarios	TS47-TS57, TS59-67

ID	24MM.SEC.PRJ.TST.14
Requirement	The configuration of the fuzz tests shall be supplied to Toyota for review and approval.
Reasoning	Toyota must be able to confirm the depth of the fuzz testing meets expectations.
Evidence	Fuzz test configuration.
Threat Scenarios	TS47-TS57, TS59-67

2.4 Factory Provisioning

The 24MM device will require provisioning at the factory with Critical Security Parameters (CSPs). This includes client authentication keys and keys needed for updates and secure transfer of information. Most of the CSPs will be provisioned at the security chip factory and then the security chip installed at the 24MM factory. The security chip provisioning process is described in separate documents.

However, certain trusted operations must also be performed at the 24MM factory:

1. Loading and execution of sensitive provisioning software for one-time use.
2. Establish binding between the Qualcomm processor and the security chip.

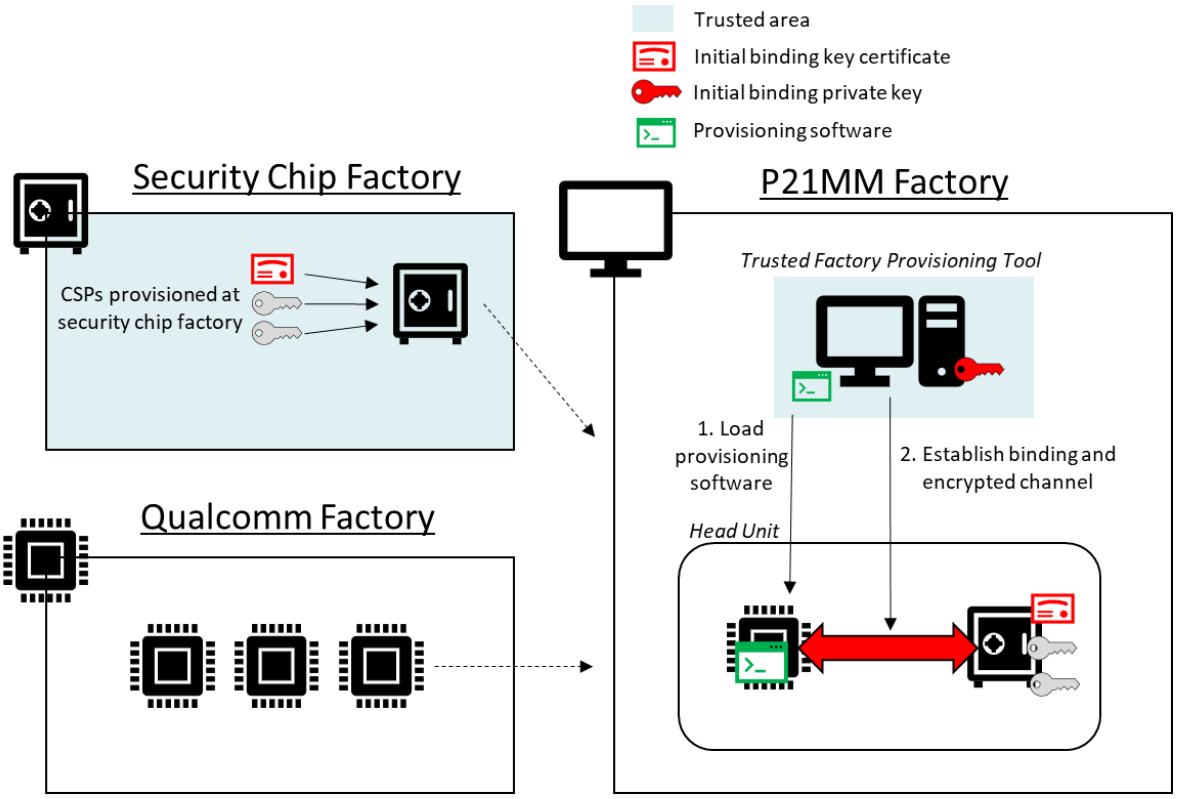

Figure 5 - Factory Provisioning

Below are high-level requirements for the trusted area in the 24MM factory. Additional requirements will be added with input from Toyota ISM.

ID	24MM.SEC.PRJ.FCT.1
Requirement	24MM factory shall support a trusted area that secures: 1. Cryptographic keys used for binding the Qualcomm processor to the security chip. 2. Sensitive provisioning software.
Reasoning	A trusted area of the factory is needed to secure information necessary for binding and provisioning steps.
Evidence	Document describing the security controls of the trusted area.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PRJ.FCT.2
Requirement	Physical access to the trusted factory provisioning tools and computing hardware shall be limited to authorized personnel only.
Reasoning	Reduce the risk that unauthorized personnel can leak sensitive provisioning software or binding key.

Evidence	Document describing the security controls of the trusted area.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PRJ.FCT.3
Requirement	Only authorized personnel shall be able to login and use the trusted factory provisioning tool.
Reasoning	Reduce the risk that unauthorized personnel can leak sensitive provisioning software or binding key.
Evidence	Document describing the security controls of the trusted area.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PRJ.FCT.4
Requirement	The trusted factory provisioning tool shall be standalone and isolated from any network.
Reasoning	Prevents a remote attack that can leak sensitive software or keys.
Evidence	Document describing the security controls of the trusted area.
Threat Scenarios	TS20, TS21

3 Hardware Requirements

3.1 Secure Boot

ID	24MM.SEC.HW.SB.1
Requirement	Every processor with support for secure boot shall implement secure boot. Processors that do not support secure boot shall be documented by the suppliers and reviewed and approved by Toyota. Secure boot is defined as cryptographically verifying the authenticity of all software and static data before the software is executed or the static data is used. Note: “Processor” includes every programmable processor within every microprocessor, microcontroller, DSP, and other SoC on the ECU.
Reasoning	All software images must be validated to ensure an attacker has not modified them. Verifying only the initial boot images still allows an attacker to modify the device behavior by modifying or replacing software later in the boot chain.
Evidence	Materials from secure boot design review.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.HW.SB.2
Requirement	Every processor with support for anti-rollback shall implement anti-rollback. Processors that do not support anti-rollback shall be documented by the suppliers and reviewed and approved by Toyota. Anti-rollback is defined as a hardware-based mechanism that permanently prevents loading software images older than a specified security version. Note for RFQ: This may require the SoC ROM to support verifying a software security version stored in eFuses or another similar mechanism.
Reasoning	Any software image has the potential for vulnerabilities that could cause a serious impact to the ECU. Therefore, all software images must be protected from rollback.
Evidence	Materials from secure boot design review shall describe anti-rollback mechanism.
Threat Scenarios	TS1-5, TS9-13

3.2 Communications

3.2.1 Wi-Fi

ID	24MM.SEC.HW.COM.WIFI.1

Requirement	The Wi-Fi chipset shall support WPA2 and WPA3 modes.
Reasoning	WPA3 provides the strongest security for standard Wi-Fi connections.
Evidence	Datasheet that lists supported WPA modes.
Threat Scenarios	TS59-64, 68

ID	24MM.SEC.HW.COM.WIFI.2
Requirement	The Wi-Fi chipset shall allow the main SOC to configure new MAC addresses.
Reasoning	Enables random MAC addresses to minimize privacy tracking concerns.
Evidence	Datasheet or other confirmation of ability to change MAC address.
Threat Scenarios	TS29

3.2.2 Bluetooth

ID	24MM.SEC.HW.COM.BLT.1
Requirement	The Bluetooth chipset shall support Bluetooth 4.2 or greater.
Reasoning	Latest and strongest security features are available with Bluetooth 4.2.
Evidence	Datasheet that lists supported Bluetooth version.
Threat Scenarios	TS59-64, 68

3.3 Peripherals

ID	24MM.SEC.HW.PER.1
Requirement	The following peripherals shall connect to the SoC in such a way that they can be configured as trusted peripherals that can only be accessed from TrustZone: • Security chips (e.g., TPM, HSM). Note: This may require these peripherals to be connected via I2C, SPI, or UART to allow control from TrustZone.
Reasoning	Some peripherals will perform only security related functions and should be inaccessible from outside TrustZone
Evidence	Schematic showing the SoC bus that the peripheral is connected to and the TrustZone peripheral configuration.

Threat Scenarios	TS40-44, 46

ID	24MM.SEC.HW.PER.2
Requirement	Unused GPIO and peripheral pins on all SOCs shall be disabled and should be grounded as appropriate.
Reasoning	Prevent an attacker from using unused but enabled peripherals to affect the behavior of the receiver or camera.
Evidence	Board schematics, PCB routing diagrams and list of GPIO functions / peripheral usage.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.HW.PER.3
Requirement	Wireless charging interfaces shall not have any communication data sent between the external device being charged and any ECU processors.
Reasoning	Ensure there is no attack surface from wireless charging devices sending malicious communications.
Evidence	Board schematics that confirm no communication signals run between the wireless charger and any processors on the ECU.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.HW.PER.4
Requirement	HDCP shall be enabled either via a fixed hardware mechanism or via a software mechanism controlled by trusted software in the TEE. Note: if HDCP is implemented by a serializer/deserializer chip, that chip must enforce HDCP using one of the following mechanisms: 1. Fixed configuration that always enforces HDCP. 2. HW pin that is fixed to the setting to enforce HDCP. 3. Trusted software in the TEE. This implies the bus used to configure the chip (I2C, SPI, etc.) is controlled exclusively by the TEE. This requirement is recommended.
Reasoning	Level 1 DRM controls require hardware-level controls to protect the content.
Evidence	Board schematics and chip documentation that shows how HDCP is enforced.
Threat Scenarios	TS29, TS46

3.4 Storage and Memory

ID	24MM.SEC.HW.MEM.1

Requirement	The ECU shall support a tamper-resistant, certified environment for storing and processing client certificates for mutual-TLS.
	
	The environment shall be certified with one of the following methods:
	
	• Common Criteria EAL4+ with AVA_VAN.4 or higher.
	• Resistance to attackers with attack potential of Moderate according to a protection profile accepted by Toyota. Evaluation Shall be done by a SOGIS (Senior Officials Group Information Systems Security) qualified/authorizing participant (i.e., SOGIS accredited lab). Accepted protection profiles include:
	  ○ JIL Application of Attack Potential to Smartcards and Similar Devices
	  ○ GlobalPlatform TEE Protection Profile

Reasoning	If extracted by an attacker, the client certificate could be used to access Toyota backend services maliciously. The security of the client certificate must be preserved
Evidence	Supporting documentation of the certifications.
Threat Scenarios	TS18-21

ID	24MM.SEC.HW.MEM.2
Requirement	Flash storage for the main SoC shall support RPMB.
Reasoning	RPMB is useful for support rollback prevention. For instance, an attacker may try to flash old, vulnerable software that has been already fixed.
Evidence	Datasheet for the flash storage of the main SoC which lists RPMB support.
Threat Scenarios	TS1-5, TS9-13, TS18, TS22, TS23, TS25, TS26, TS30, TS31, TS35, TS36, TS70

ID	24MM.SEC.HW.MEM.3
Requirement	All RAM modules shall be soldered in place. RAM modules shall not be installed in sockets or in any way that allows easy removal.
Reasoning	Prevents RAM modules from being removed for a cold-boot attack
Evidence	N/A – Confirmed through physical analysis.
Threat Scenarios	TS18, TS20, TS21, TS22, TS24, TS25, TS26, TS29

ID	24MM.SEC.HW.MEM.4
Requirement	RAM bus signals shall be routed such that an attacker cannot connect to test points, external IC pins, or surface traces in a way that allows them to impersonate a memory controller and access RAM contents.
Reasoning	Prevent attachment of rogue memory controller to RAM while device is in suspended state.
Evidence	PCB layout for the data flow associated with the RAM.
Threat Scenarios	TS18, TS20, TS21, TS22, TS24, TS25, TS26, TS29

4 Platform Requirements

4.1 Cryptographic Algorithms

Goal: Use only established cryptographic algorithms and parameters so that cracking encryption or forging signatures is infeasible.

ID	24MM.SEC.PLAT.CRYP.1
Requirement	Digital signatures shall satisfy FIPS PUB 186-4 and use key lengths of security strength 128 or greater according to NIST SP 800-57 Part 1 Rev. 5. All other signature algorithms shall be approved by Toyota. Examples: RSA-4096; ECDSA-256; ECDSA-384; ED25519. Note: Use cases that require digital signatures include Secure Boot and Secure Updates.
Reasoning	Use established signature algorithms to ensure an attacker cannot forge a signature to modify a software image.
Evidence	List of all uses of digital signatures and the algorithms and key lengths used.
Threat Scenarios	TS20

ID	24MM.SEC.PLAT.CRYP.2
Requirement	Only Message Authentication Codes (MACs) approved for use by NIST SP 800-131A Rev. 2 shall be used.
Reasoning	Use NIST guidance to ensure MAC algorithms are resistant to attack for a long -time window.
Evidence	List of all uses of MACs and the algorithms and key lengths used.
Threat Scenarios	TS20

ID	24MM.SEC.PLAT.CRYP.3
Requirement	Hash functions shall be of security strength 128 or greater according to NIST SP 800-57 Part 1 Rev 5. All other hashing algorithms used shall be approved by Toyota. For example: SHA-256, SHA-384. Note: “Security strength” is different from the number of bits in the hash result. A security strength of 128 does not mean a SHA-256 hash can be truncated to 128 bits. The full hash result shall be used. Note: Use cases that require hash functions include Secure Boot and Secure Updates.
Reasoning	Must ensure the software image hash is resistant to collision attacks.
Evidence	List of all uses of hashes and the algorithms used.
Threat Scenarios	TS20

ID	24MM.SEC.PLAT.CRYP.4
Requirement	AES-128, AES-192, and AES-256 shall be the only symmetric encryption algorithms used. Implementations of AES shall use a mode approved in the NIST SP 800-38 series. However, ECB mode shall not be used unless given approval by Toyota. Note: XTS is the preferred AES mode for bulk storage encryption use cases. GCM is the preferred AES mode for other use cases.
Reasoning	Use NIST guidance to ensure encryption algorithms are resistant to attack for a long - time window.
Evidence	List of all uses symmetric encryption and the algorithms and key lengths used.
Threat Scenarios	TS20

ID	24MM.SEC.PLAT.CRYP.5
Requirement	If an Initialization Vector (IV) is needed for an AES mode of operation, the IV shall be generated randomly using a TRNG or PRNG seeded by a TRNG.
Reasoning	Must ensure resistance to dictionary and chosen plaintext attacks. See CWE-329 [3] for more details.
Evidence	List of all uses of IVs and the source of the IV value.
Threat Scenarios	TS20

ID	24MM.SEC.PLAT.CRYP.7
Requirement	When performing symmetric key derivation, the ECU shall use one of the following methods described in the following NIST guidelines: • SP 800-56C Rev. 2 • SP 800-108 • SP 800-132 • SP 800-135 Rev. 1 Or Elliptic Curve Integrated Encryption Scheme.
Reasoning	Symmetric key derivation functions can be used to securely generate temporary keys based on a pre-shared secret.
Evidence	List of all uses of symmetric key derivation and the algorithms used.
Threat Scenarios	TS20

4.2 Key Management

Goal: Protect cryptographic material so that it is not disclosed to an attacker.

Note: The design should minimize the number of pre-shared and stored cryptographic keys on the ECU. Instead, dynamic keys should be used wherever possible, leveraging techniques such as Butterfly Keys. This reduces the maintenance burden of maintaining many keys for each ECU.

ID	24MM.SEC.PLAT.KEY.2
Requirement	Pre-shared and stored cryptographic secrets (e.g., private asymmetric keys, symmetric keys) shall be stored and managed by the Trusted Execution Environment (TEE) or a secure element (SE; e.g., HSM, security processor). All cryptographic operations using the cryptographic secrets shall be performed within the TEE or SE. Whenever possible, the TEE or SE with the highest security shall be used. The cryptographic secrets shall never be accessible to normal world software. However, cryptographic operations using cryptographic secrets can be initiated by normal world software.
Reasoning	Must ensure cryptographic secrets are not disclosed to an attacker. By restricting access to secrets to the TEE, it becomes more difficult for an attacker to obtain access.
Evidence	List of every key provisioned on the device and the entity that stores and uses it.
Threat Scenarios	TS18, TS19, TS20, TS21

ID	24MM.SEC.PLAT.KEY.3
Requirement	Access control shall be implemented such that only the intended applications may access or use cryptographic material.
Reasoning	Prevents an attacker broadening their access to the system by compromising an application and using it to perform cryptographic operations on keys that should only be used by a different application.
Evidence	Design document for key management that describe access control.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.KEY.4
Requirement	Cryptographic secrets (e.g., asymmetric private keys, symmetric keys) shall be confidentiality protected using a mechanism backed by the TEE or secure element. The mechanism shall ensure that the cryptographic material cannot be read by any software outside of the TEE or secure element or by an attacker probing external communication buses.
Reasoning	Ensure there is high resistance to an attacker attempting to read secret keys.
Evidence	Design document for key management that describes confidentiality protections.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.KEY.5
Requirement	Cryptographic material shall be integrity protected using a mechanism backed by the TEE or secure element. The mechanism shall ensure that the cryptographic material cannot be modified outside of the intended update process. As Tier1's responsibility, integrate the application provided by TMNA that is operated correctly under SecurityHAL.
Reasoning	Must ensure an attacker cannot modify key information without detection.
Evidence	Design document for key management that describes integrity protections.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.KEY.6
Requirement	Cryptographic material shall be protected from rollback attacks using a mechanism backed by the TEE or secure element.
Reasoning	Must ensure an attacker cannot use old or compromised keys.
Evidence	Design document for key management that describes rollback protections.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.KEY.7
Requirement	Cryptographic material shall be updateable according to a defined key rotation schedule. Cryptographic material that is not planned for rotation must be reviewed and approved by Toyota.
Reasoning	It is best practice to rotate key material periodically to minimize impact of disclosure.
Evidence	Procedures for generating key updates.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.KEY.8
Requirement	The cryptographic material update process shall encrypt and authenticate the data using special keys used only for key updates, not the same keys used for general updates. The special key used only for key updates refers to A014 (public key for key updates) in the Appendix C+D Key Formats.
Reasoning	The cryptographic material update process must be highly secured. Using special keys reserved for that purpose alone reduces the risk those keys will be compromised.
Verification	The supplier can validate this requirement by performing a key update service using the assigned keys from the Toyota server. All traces ang logs shall be provided as evidence
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.KEY.9
Requirement	The cryptographic material update process shall be performed with the TEE or secure element. Note: Cryptographic material, like new key values, must never be accessible by normal world software. Therefore, APIs that import cryptographic material by wrapping with a HW-based key (e.g. Qualcomm Keymaster) must be accessible from secure world.
Reasoning	Must ensure normal world software can never access or manipulate keys, since normal world software has a higher risk of exploitation by an attacker.
Evidence	Design document for key management that describes key update implementation.
Threat Scenarios	TS18, TS20, TS21

ID	24MM.SEC.PLAT.KEY.10
Requirement	No references to old cryptographic material shall exist after the cryptographic material update process.

Reasoning	Must ensure there is no possibility of mistakenly using an old encryption key.
Evidence	Design document for key management that describes deletion of old keys.
Threat Scenarios	TS20, TS21

4.3 Secure Boot

The pieces of software and static data protected by secure boot are collectively referred to as “software images” hereafter.

ID	24MM.SEC.PLAT.SB.1
Requirement	Every processor with support for secure boot shall verify the authenticity of all software and static data before the software is executed or the static data is used. This process is called “secure boot.” This verification shall be done cryptographically using the processes described in this section. Processors which do not support secure boot shall be documented by the suppliers and reviewed and approved by Toyota.
Reasoning	All software images must be validated to ensure an attacker has not modified them. Verifying only the initial boot images still allows an attacker to modify the device behavior by modifying or replacing software later in the boot chain.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.2
Requirement	Complete secure boot shall be the only boot mode supported. There shall be no mechanism to bypass or disable secure boot at any stage of the secure boot process. Example: Disabling secure boot based on an external pin or a configuration value in modifiable flash is not permitted.
Reasoning	Prevents an attacker changing the boot mode so that secure boot is disabled.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.3
Requirement	Secure boot shall not include facilities that can halt boot-up and enter a general-purpose command-line or other mode that is not designed to prevent secure boot bypass. Example: A U-Boot shell that can be entered by pressing ‘Enter’ during boot is not permitted.

Reasoning	Shells and other facilities that affect the boot process are attack surfaces with potential for exploitation by an attacker.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.4
Requirement	At power-up the SoC shall always execute the first instruction from an immutable ROM embedded within the SoC. Note: For devices without a mask ROM that implements the initial boot stage, an internal flash that is marked One-Time Programmable (OTP) can be used to satisfy this requirement.
Reasoning	Prevents an attacker from bypassing the secure boot mechanism by modifying the first boot stage.
Evidence	Datasheet of processors that describes possible boot methods and design document that describes how ROM mode is only mode allowed.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.5
Requirement	Any facilities that can modify the SoC ROM execution shall require authentication before installing modifications. Example: ROM patching mechanisms.
Reasoning	Prevents an attacker from abusing a ROM patching mechanism to bypass secure boot.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.6
Requirement	There shall not be a mechanism to disable any Toyota approved modifications to the SoC ROM execution after installation.
Reasoning	Prevents an attacker from removing or disabling a ROM patch.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.7
Requirement	Approval by Toyota shall be received prior to deploying any modifications to the SoC ROM.
Reasoning	Toyota must fully understand the secure boot process and any changes made to it.
Evidence	Written confirmation from SoC supplier that they will comply.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.8
Requirement	When performing secure boot processing, the SoC ROM code shall only use volatile memory internal to the SoC or an encrypted and integrity protected region of external memory.
Reasoning	Prevent an attacker from modifying the secure boot processing or gleaning information about secret values used during secure boot.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.9
Requirement	SoC ROM code shall operate within a secure processing environment that restricts access to the volatile memory used for secure boot processing to only code within the secure processing environment. Example: ARM TrustZone.
Reasoning	Prevent less trusted software from modifying the secure boot processing or gleaning information about secret values used during secure boot.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.10
Requirement	SoC ROM code shall start operation at the most privileged level of processing/exception level segmentation.
Reasoning	Helps ensure that low privileged code cannot modify or tamper with highly privileged code.

Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.11
Requirement	Later stage bootloaders shall not operate at a privilege level higher than the highest privilege of the image portion they are booting.
Reasoning	A vulnerability in an unprivileged bootloader stage should not put secure processes at risk.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.12
Requirement	The mask ROM shall execute all cryptographic operations using the hardware cryptographic engine.
Reasoning	Ensures any tampering or side-channel analysis protections that are built into the cryptographic engine are used. Also, it should result in faster processing.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.13
Requirement	All software and static data shall have a chain of trust that links back to the hardware-based root of trust. This chain of trust shall be cryptographically verified before the software is executed or the static data is used. Note: This includes the SoC ROM boot stage, which shall verify the authenticity of all the software images it loads. The above verification may be performed in background. Note: This includes Linux application software and static configuration files. For example, this requirement is not satisfied if an attacker can replace a systemd unit file to prevent or alter the execution of an application.
Reasoning	Verifying every boot stage before execution creates a chain of trust. Starting this chain from the immutable SoC ROM till the last boot stage creates a chain of trust that prevents an attacker from modifying any software images at any point in the boot chain.

Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.14
Requirement	Boot stages after the ROM code should still use the ROM code to perform the authenticity check.
Reasoning	Having a single, highly trusted implementation of the authenticity check minimizes the chances of errors that could be exploited by an attacker.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.15
Requirement	Whenever supported by the SoC, the authenticity of a software image shall be verified by the SoC that will use the software image (the “target SoC”). Processors which do not verify the authenticity of their software images must be documented by the suppliers and reviewed and approved by Toyota. Note: If the software image is sent to the target SoC by another SoC (the “sender SoC”), the target SoC should still verify the authenticity of the software image before using it. The sender SoC should not be the only SoC to verify the authenticity of the software image.
Reasoning	If the target SoC does not perform its own verification, an attacker with control of the sender SoC could compromise the target SoC by sending it a malicious software image.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.16
Requirement	The authenticity of a software image shall be confirmed in two steps: • Verifying the digital signature of a secure boot certificate containing the length and cryptographic hash of the software image. • Verifying the hash of the software image matches the hash contained in the secure boot certificate. Note: The strength of the algorithms for this process are defined in the following requirements.
Reasoning	Ensures that an attacker cannot modify the software image or the secure boot certificate so they can execute a malicious software image.

Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.17
Requirement	The keys used for verifying digital signatures shall be verified to be part of a public key certificate chain that ends with a trusted root key.
Reasoning	Ensure an attacker does not replace a secure boot signing key with their own key that allows them to sign malicious software.
Evidence	Secure boot design review materials.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.SB.18
Requirement	The trusted root key use for secure boot signature verification shall be stored immutably in the SoC hardware. Note: The hash of the trusted root key may be stored immutably in the SoC hardware instead of the full key. The hash must comply with 24MM.SEC.PLAT.CRYP.3.
Reasoning	Link all secure boot keys to a hardware-based root-of-trust ensures an attacker cannot replace the root key to allow them to sign their own malicious software images.
Evidence	Secure boot design review materials.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.SB.19
Requirement	All secure boot certificates should implement the X.509 v3 format. Alternate certificate formats shall be documented by suppliers and shared with Toyota for review and approval.
Reasoning	Use industry standard formats to improve interoperability and prevent vulnerabilities common to proprietary, unanalyzed formats.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.20

Requirement	If an HMAC or CMAC is used to verify authenticity, the secret key shall be stored immutably in the SoC hardware.
Reasoning	Ensure an attacker cannot replace the secret key so they can generate authentication codes for their own malicious software images.
Evidence	Secure boot design review materials.
Threat Scenarios	TS18

ID	24MM.SEC.PLAT.SB.21
Requirement	If an HMAC or CMAC is used to verify authenticity, the secret key shall be stored in such a way that no software can read it. Only the cryptographic engine performing the computations may access the secret key.
Reasoning	Minimizes the risk of compromised software leaking the secret key. An attacker with the secret key could generate valid HMACs or CMACs for their own malicious software images.
Evidence	Processor datasheet describing compliance or written confirmation from SoC supplier that they satisfy the requirement.
Threat Scenarios	TS21

ID	24MM.SEC.PLAT.SB.22
Requirement	All kernel modules loaded by the Linux operating system shall be signed with a key issued by Tier1. Implement the contents described in the URL below. https://www.kernel.org/doc/html/v5.0/admin-guide/module-signing.html
Reasoning	Prevent an attacker with root privileges from loading malicious kernel images not issued by Toyota.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.23
Requirement	Revocation of keys used for secure boot shall be done in a way that is irreversible.
Reasoning	If key revocation is reversable an attacker may reverse it, and use known vulnerable key material.
Evidence	Secure boot design review materials.

Threat Scenarios	TS18

ID	24MM.SEC.PLAT.SB.24
Requirement	Whenever supported by the SoC, the secure boot process shall support a mechanism that permanently prevents loading software images older than a specified security version. This mechanism is hereafter referred to as the “anti-rollback mechanism.” Secure boot processes which do not support an anti-rollback mechanism shall be documented by suppliers and reviewed and approved by Toyota.
Reasoning	Prevent an attacker from “rolling back” to an old, valid software image that has known vulnerabilities.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.25
Requirement	The anti-rollback mechanism shall support increasing the security version through updates.
Reasoning	When a vulnerability is identified, ECUs that are already produced must prevent an attacker from rolling back to the old, vulnerable version after an update.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.26
Requirement	There shall be an anti-rollback mechanism to support every software image. Note: This includes the first software image loaded by the SoC ROM. This may require the SoC ROM to support verifying a security version stored in hardware monotonic counters implemented using One-Time Programmable (OTP) memory or eFuses.
Reasoning	Any software image has the potential for vulnerabilities that could cause a serious impact to the ECU. Therefore, all software images must be protected from rollback.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.27
Requirement	Before entering a Suspend-to-RAM state, the SoC shall store all sensitive information into secure storage and wipe sensitive information from external RAM.
Reasoning	Minimizes damage caused by an attacker dumping RAM contents while the SoC is in a suspended state
Evidence	Secure boot design review materials.
Threat Scenarios	TS18, TS20, TS21, TS22, TS24, TS25, TS26, TS29

ID	24MM.SEC.PLAT.SB.28
Requirement	The secure boot process shall be monitored by a watchdog or a securely loaded sub-processor of the SoC. The boot process shall be aborted if it does not load and validate all secure boot images in the expected time window.
Reasoning	A long boot time may be because of an attack and the boot should be aborted to break the attack.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.29
Requirement	Secure boot verification code shall be constructed to be resistant to bypass by power and clock glitch attacks. Example: Software mitigations could include duplicate checks separated by non-deterministic intervals of time.
Reasoning	Glitch attacks represent an increasingly common form of attack against embedded systems. High impact verification checks must be resistant to such attacks.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.30
Requirement	If software authentication fails or the secure boot time monitor aborts, the ECU shall: • Abort bootup. • Create a persistent log entry indicating the failure. • Boot from alternate boot side. • If failure, repeat steps 1-3 up to 2 additional times. • If bootup is still unsuccessful, transition into a recovery mode.
Reasoning	Ensure that secure boot does not continue to load software modules that are not verified.

Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.31
Requirement	If the ECU is unable to boot, the ECU shall enter a recovery mode that allows supplier and Toyota personnel to analyze and restore the head unit. Note: Analysis and restoration can be accomplished by allowing new software or debug configuration to be loaded from an alternate media such as USB or serial. Note: Recovery mode shall not automatically enable debug ports or special debug software. It shall only allow authenticated software images or authenticated debug configuration to be loaded. Those images must still be verified as being issued by Toyota according to the requirements in this section.
Reasoning	Allow supplier personnel to securely diagnose and recover a malfunctioning or corrupted ECU.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.SB.32
Requirement	The recovery mode should provide basic functionality if possible. The basic functionality shall provide functionality essential for vehicle operation. Note: The basic functionality must still be executed through a secure boot process.
Reasoning	If full boot fails, it may be possible to fall back to basic functionality to allow continued vehicle operation.
Evidence	Secure boot design review materials.
Threat Scenarios	TS6, TS14

4.4 Secure Updates

Updating software and other data is one of the most important features needed by computing systems. Updates are needed to provide new features to customers and most importantly apply fixes to vulnerabilities that get discovered after production. Updates can be delivered through multiple mediums, for example they can be delivered locally by loading on a USB flash drive and then inserting into the MM device. Updates can also be delivered remotely over the cellular network or even through Wi-Fi at the owner's home as depicted in Figure 6.

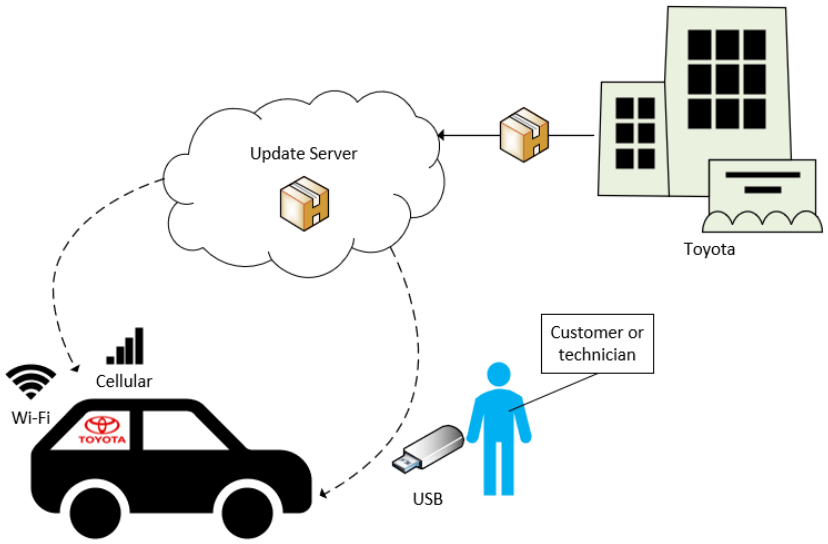

Figure 6. Possible transports for updates to vehicle

Ensuring confidentiality, integrity, and authenticity of updates should be independent of the transport medium used to deliver the update to the target ECU. Integrity and authenticity ensure the update has arrived at the ECU unmodified (integrity) and it came from the proper author (authenticity). Confidentiality ensures an update in transit cannot be captured and reverse engineered. With all three concepts in place and implemented well we can be assured updates are authentic and desired. This concept is paired with Secure Boot, which ensures that only authentic code is run on a processor no matter how it got there.

Requirements:
The pieces of software and data that can be updated are collectively referred to as “software images” hereafter. Software includes the first image loaded by the immutable ROM of the SoC and extends to all application software, including Linux user space applications and libraries. It also includes software images for peripherals or subcomponents sometimes referred to as “firmware.”

Data that can be updated is data that does not change between updates. This includes OS configuration files, Mandatory Access Control policies, map data, read-only databases, device tree blobs, etc. However, data could include any data on the ECU, including mutable data such as modifiable databases, caches, etc.

The information that must be transmitted to perform an update, including metadata and the update data itself, is collectively referred to as an “update package.”

ID	24MM.SEC.PLAT.UPD.1
Requirement	The ECU device shall only apply updates that have first been authenticated as being issued by Toyota using a key that chains to a Toyota root-of-trust key.
Reasoning	Ensure an attacker cannot modify the device software through a malicious update.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.2
Requirement	The ECU device shall not execute any component of an update package that has not first been authenticated as being issued by Toyota using a key that chains to a Toyota root-of-trust key.
Reasoning	Ensure an attacker cannot execute malware through a malicious update.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.3
Requirement	For every SoC that support software updates, the SoC shall implement a software update mechanism for every software image. Processors that do not support software updates shall be documented by suppliers and reviewed and approved by Toyota.
Reasoning	Any piece of software could have a cybersecurity vulnerability that must be addressed. All software must be updateable to address any potential issues.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.4
Requirement	The ECU should be compliant with the AUTOSAR Adaptive Platform update process.
Reasoning	AUTOSAR Adaptive Platform provides a framework that helps to secure updates to ECUs in vehicles.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.5
Requirement	The update mechanism shall support local and remote distribution of update packages.
Reasoning	Supporting full remote updatability allows security issues to be addressed rapidly and completely. However, local updates are also sometimes necessary.
Evidence	Secure update design review materials and update procedures for local and remote updates.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.6
Requirement	Update packages shall be signed, encrypted, and signed again.
Reasoning	Performing an internal signature ensures authenticity of the software plaintext while the outer signature can increase performance by ensuring the integrity of the ciphertext without decrypting the package.
Evidence	Secure update design review materials.

Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.7
Requirement	For every processor that can support image validation, the processor shall validate update images by itself, even if the image was validated by an external processor. Processors that do not support image validation shall be documented by suppliers and reviewed and approved by Toyota.
Reasoning	Prevent an attacker from injecting a malicious update directly to another microcontroller or SoC.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.8
Requirement	Secure update verification code shall be constructed to be resistant to bypass by power and clock glitch attacks. Example: Software mitigations could include duplicate checks separated by non-deterministic intervals of time.
Reasoning	Glitch attacks represent an increasingly common form of attack against embedded systems. High impact verification checks must be resistant to such attacks.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.9
Requirement	The ECU should include a mechanism to verify a software package is linked to the entire vehicle update campaign.
Reasoning	Updates to ECUs should not be independent, instead updates should be linked to the entire update process to all ECUs to ensure correct versioning. This can be accomplished with AUTOSAR's concept of vehicle packages.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.10
Requirement	The ECU shall include a mechanism to verify the size of a software image before installing. The specified size shall be protected with the authentication mechanism. For example: A manifest file can specify a hash and size for a corresponding image and the manifest file is then digitally signed.
Reasoning	Specifying a size during verification phase ensures attackers cannot override the size by sending an infinite amount of data until target ECU runs out of storage and crashes.

Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.11
Requirement	The ECU shall not allow software to be rolled back to old versions.
Reasoning	Ensure an attacker cannot downgrade to an old software version with known vulnerabilities to attack the head unit.
Verification	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.12
Requirement	Each processor with an update capability shall contain two memory partitions, one containing current software and the other for validating new software. The processor shall activate new software after successful validation and health check following bootup and then switch default boot from old to new software . This is often referred to as “A/B storage.”
Reasoning	Maintaining two partitions for the update process allows for validation of updates while preserving current working software.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-6, TS9-14, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.13
Requirement	An update for a processor must be encrypted in transit and must remain encrypted until in the processor’s memory.
Reasoning	Update packages may contain Toyota IP (Intellectual Property) or details an attacker can use to attack the system.
Evidence	Secure update design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS29, TS34

ID	24MM.SEC.PLAT.UPD.14
Requirement	An update for a processor in the ECU shall be encrypted when stored in external non-volatile memory from the target processor.
Reasoning	Update packages may contain Toyota IP or details an attacker can use to attack the system.
Evidence	Secure update design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS29, TS34

ID	24MM.SEC.PLAT.UPD.15
Requirement	An update for a processor shall carry with it a mechanism to verify the integrity of the update package using a hashing mechanism compliant with 24MM.SEC.PLAT.CRYP.3.

Reasoning	Corrupted images (intentionally or unintentionally) can cause issues with customer experience, safety or security if loaded.
Evidence	Secure update design review materials. Toyota shall tamper with the hash of a software update package and verify that the target ECU rejects the software package. Analysis of logs can help to verify if the correct behavior was observed.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.16
Requirement	The ECU shall not allow any method (such as a backdoor) to bypass, disable, or circumvent the secure update procedure. For Example: Mechanisms which disable cryptographic assurances of secrecy, integrity, or authenticity are not allowed. This includes any mechanisms that accept images not signed by a Toyota issue key or accept images not signed at all.
Reasoning	Bypasses to the update procedure could be used by an attacker to load malicious software images.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.17
Requirement	The ECU shall inform Toyota server if an update fails.
Reasoning	Toyota must be able to analyze update failures to detect malicious behavior.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.UPD.18
Requirement	No update files or software images shall persist on the device after an update is successfully applied. This includes firmware images for peripheral processors. Note: firmware that must be loaded onto peripheral processors during boot may persist. Note: alternate boot side images are OK to persist.
Reasoning	Peripheral firmware left cached on the device can be used by an attacker to reverse-engineer the peripheral and identify vulnerabilities.
Evidence	Secure update design review materials.
Threat Scenarios	TS8, TS17

ID	24MM.SEC.PLAT.UPD.19
Requirement	The software application that applies an update to the device shall be provided along with the update. The software update application shall not persist on the device.

	Note: This includes software update applications for peripheral processors.
Reasoning	Prevents an attacker from reverse-engineering the software update application to bypass software update security and apply a malicious update.
Evidence	Secure update design review materials.
Threat Scenarios	TS1-7, TS9-16

4.5 Secure Debug

4.5.1 General

ID	24MM.SEC.PLAT.DBG.GEN.1
Requirement	All debug and diagnostic interfaces and services of the ECU shall require authentication and shall check that the user is authorized to use that interface. Note: This applies for debug and production devices and all software builds.
Reasoning	If a debug unit were to be obtained by an attacker, the information contained on the debug unit must still be protected by requiring authorization. Also prevent an unauthorized test device or a compromised ECU from gathering information or modifying data of the ECU.
Evidence	Secure debug design review materials, including a list of all available debug interfaces and the corresponding authentication mechanism.
Threat Scenarios	TS1, TS3, TS6, TS9, TS11, TS14, TS18-26, TS30, TS33-36, TS38, TS39, TS44, TS46, TS47

4.5.2 Debug during development

ID	24MM.SEC.PLAT.DBG.DEV.1
Requirement	No production cryptographic keys or other secrets shall be installed on a debug unit or built into debug software.
Reasoning	If an attacker was able to obtain a debug unit, they may extract secret data. Must ensure compromise of data used in debug cannot be used to attack production units.
Evidence	Secure debug design review materials, including a description of how production keys are separated from debug units and software.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.DBG.DEV.2
Requirement	A debug unit shall only be issued credentials that clearly indicate it is a debug unit.
Reasoning	This allows production backend systems to deny access to debug units.
Evidence	Schema for debug identity credentials.

Threat Scenarios	TS59-64

4.5.3 Debug of production devices

ID	24MM.SEC.PLAT.DBG.PROD.1
Requirement	All debug and diagnostic facilities and services shall be removed or disabled on production units. The exceptions are: • Debug or diagnostic services necessary for dealer or service technicians. • Retrieving logs for analysis. • Secure debug mechanism (see below). Note: If the supplier must perform deeper debug or diagnostics on a production unit, they must use the secure debug mechanism. This requirement is intended for SoC.
Reasoning	Debug and diagnostic services offer a rich attack surface that could be exploited by an attacker or used to gather information for finding potential vulnerabilities.
Evidence	Board schematics and PCB routing showing locations of debug interfaces.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.2
Requirement	An exclusive secure debug mechanism shall be implemented that allows debug facilities to be added to only specific production devices based on a unique identifier tied to hardware. Note: Suppliers shall not implement any other debug mechanism for production devices. This requirement is intended for SoC.
Reasoning	Provide access and capabilities needed for internal test and validation procedures.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.3
Requirement	The secure debug mechanism shall use a hardware-based, cryptographic authentication mechanism (e.g., Qualcomm debug policy) to verify that Toyota issued credentials have enabled secure debug. This requirement is intended for SoC.
Reasoning	Provide access and capabilities needed for internal test and validation procedures based directly on a production image.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.4
Requirement	The secure debug mechanism shall support loading and executing new software applications, including all facilities listed below. All added facilities shall be loaded from an externally supplied debug image. The debug facilities shall not be pre-existing on the production device, even in encrypted or deactivated form. This requirement is intended for SoC. Note: Toyota must have the ability to independently create, sign, and load their own debug images. Supported facilities that can be added by secure debug support include: • Access an authenticated Linux shell running as the root user with full privileges. The shell shall have access to Linux utilities that allow testers the necessary internal access for cybersecurity requirement validation. • Dump an unencrypted image of external storage • Disable Mandatory Access Control. • Load new root certificate authorities. • Install a script that is run as root at Linux bootup. • Collect TCP/IP logs and traces. • Read eFuses needed for validation of SoC specific requirements. • Read physical memory. • Mount USB drives and NFS shares.
Reasoning	Provide access and capabilities needed for internal test and validation procedures. Ensures that on production devices no debug facilities exist that can be used by an attacker.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.5
Requirement	Only hardware debug interfaces necessary for the allowed debug facilities of 24MM.SEC.PLAT.DBG.PROD.1 shall be enabled on the SoC configuration. All other hardware debug interfaces, including JTAG, shall be disabled by the SoC configuration. This requirement is intended for SoC.
Reasoning	Ensure an attacker cannot use JTAG or other interfaces to reverse engineer or control the ECU.
Evidence	Secure debug design review materials, including eFuse or other configuration values that enforce disablement.
Threat Scenarios	TS1, TS3, TS6, TS9, TS11, TS14, TS18-26, TS30, TS33-36, TS38, TS39, TS44, TS46, TS47

ID	24MM.SEC.PLAT.DBG.PROD.6
Requirement	Any hardware debug interfaces not permanently disabled shall only be enabled after verification by the mechanism of 24MM.SEC.PLAT.DBG.PROD.3. Note: If cryptographic authentication before using a debug interface is not possible, the debug interface must be permanently disabled.

	Example mechanism: Qualcomm Debug Policy which specifies allowed debug capabilities and whose signature is verified by the mask ROM of the SoC. This requirement is intended for SoC.
Reasoning	A hardware-based authentication mechanism provides a secure root-of-trust to enable debug services.
Evidence	Secure debug design review materials.
Threat Scenarios	TS1, TS3, TS6, TS9, TS11, TS14, TS18-26, TS30, TS33-36, TS38, TS39, TS44, TS46, TS47

ID	24MM.SEC.PLAT.DBG.PROD.7
Requirement	All SoC configuration settings that enable debug functions (e.g., SPIDEN, SUIDEN, etc.) shall be disabled. It is acceptable that these debug functions can be re-enabled via the mechanism of 24MM.SEC.PLAT.DBG.PROD.3 using credentials tied to the specific device. However, if the debug functions can be re-enabled via another mechanism, the debug functions must be permanently disabled. This requirement is intended for SoC.
Reasoning	Ensure debug functions of the SoC cannot be activated by an attacker on production hardware or software.
Evidence	Secure debug design review materials
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39.

ID	24MM.SEC.PLAT.DBG.PROD.8
Requirement	Secure debug verification code shall be constructed to be resistant to bypass by glitch attacks. This requirement is intended for SoC.
Reasoning	Glitch attacks represent an increasingly common form of attack against embedded systems.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.9
Requirement	Any processors with internal flash shall disable reading internal flash from a debugger. This requirement is intended for SoC.
Reasoning	Prevent an attacker with a debugger from reading internal flash of a microcontroller.
Evidence	Secure debug design review materials, including eFuse or other configuration values that enforce disablement.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.10
Requirement	Any processors with internal flash shall only allow writes to internal flash via an authenticated mechanism, such as a certificate or device specific password managed and issued by Toyota. This requirement is intended for SoC.
Reasoning	Prevent an attacker with a debugger from modifying internal flash of a microcontroller.

Evidence	Secure debug design review materials, including eFuse or other configuration values that enforce authentication.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.PLAT.DBG.PROD.11
Requirement	Any processors with internal RAM shall disable reading and writing internal RAM from a debugger. This requirement is intended for SoC.
Reasoning	Prevent an attacker with a debugger from reading or modifying internal RAM of a microcontroller.
Evidence	Secure debug design review materials, including eFuse or other configuration values that enforce disablement.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.12
Requirement	Loading a debug software image onto a production unit shall wipe or render unusable the production secret cryptographic keys and other secrets on the device, including certificates or other information that authenticates the device as a production unit. This requirement is intended for SoC.
Reasoning	Must not allow a unit running debug software to appear to be a production device to backend services or allow production secrets to leak through debug mechanisms.
Evidence	Secure debug design review materials, including a list of keys that are wiped.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.DBG.PROD.13
Requirement	The ECU shall encrypt exported log files and debug information so that only a privileged user can access the information. This requirement is intended for SoC.
Reasoning	Ensures an attacker or unprivileged technician cannot access log files and debug information without authorization.
Evidence	Secure debug design review materials.
Threat Scenarios	TS39

ID	24MM.SEC.PLAT.DBG.PROD.14
Requirement	All debug and diagnostic authentication attempts shall be logged according to section 4.11 Logging. This requirement is intended for SoC.
Reasoning	Toyota must be able to analyze debug authentication to detect malicious behavior.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.15
Requirement	The ECU shall not allow any sensitive information or software to be read or written via diagnostic communications. This requirement is intended for SoC.

Reasoning	Access to sensitive information must be available only to authorized parties.
Evidence	Secure debug design review materials.
Threat Scenarios	TS1-5, TS8-13, TS17, TS18, TS20, TS21, TS24, TS29-31, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.16
Requirement	The ECU shall not support raw reading or writing of volatile or non-volatile memory through diagnostic communication.
Reasoning	Access to raw memory should never be provided as it provides an avenue for abuse by an attacker to gain information or change behavior of unintended components of the system.
Evidence	Secure debug design review materials.
Threat Scenarios	TS1-5, TS8-13, TS17, TS18, TS20, TS21, TS24, TS29-31, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.17
Requirement	UDS services should only support SID 0x29 for authentication using certificates. SID 0x27 should not be supported. This requirement is intended for SoC.
Reasoning	Certificate based authentication is stronger authentication than security access method and allows chaining to root of trust certificates.
Evidence	Secure debug design review materials.
Threat Scenarios	TS40-TS46

ID	24MM.SEC.PLAT.DBG.PROD.18
Requirement	The ECU shall not initiate diagnostic communications with other ECUs. This requirement is intended for SoC.
Reasoning	Diagnostic communications can be abused to perform unwanted actions on the vehicle. If the ECU is compromised, it must not be able to perform unwanted vehicle actions through diagnostics.
Evidence	Secure debug design review materials.
Threat Scenarios	TS40-TS46

ID	24MM.SEC.PLAT.DBG.PROD.19
Requirement	Uses of Diagnostics over IP (DoIP) shall be restricted to the internal Ethernet network. Access to DoIP services shall not be allowed via Wi-Fi, Bluetooth, or other communication interfaces. This requirement is intended for SoC.
Reasoning	Access to diagnostic functions opens a significant attack surface that must be inaccessible externally.
Evidence	Secure debug design review materials.
Threat Scenarios	TS40-TS46

ID	24MM.SEC.PLAT.DBG.PROD.20
Requirement	Any diagnostic function that can cause a change in the function or behavior of the ECU or vehicle shall require authentication by an authorized entity or tool before the diagnostic function can be used.
Reasoning	Prevent abuse of diagnostic functions by unauthorized parties.
Evidence	Secure debug design review materials.
Threat Scenarios	TS40-TS46

ID	24MM.SEC.PLAT.DBG.PROD.21
Requirement	The secure debug mechanism shall support loading and executing a new signed Linux kernel image, initramfs, device tree binary, and kernel boot parameters. This requirement is intended for SoC.
Reasoning	Allows kernel debugging mechanisms to be re-enabled and deeper analysis to be performed.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.22
Requirement	The secure debug mechanism shall support using new or modified startup configuration files. Examples: systemd unit files, MAC policies, logging configuration, files in /etc, etc. This requirement is intended for SoC.
Reasoning	Allows disabling or reconfiguring security mechanisms to for debug and test.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.DBG.PROD.23
Requirement	Secure debug images shall be encrypted. This requirement is intended for SoC.
Reasoning	Prevent reverse-engineering of device software or configuration if the debug image is leaked.
Evidence	Secure debug design review materials.
Threat Scenarios	TS8, TS17

4.6 Trusted Execution Environment

ID	24MM.SEC.PLAT.TEE.1
Requirement	A TEE shall be used whenever possible for performing authentication operations and for managing sensitive information. Some uses of the TEE that shall be implemented are: • Secure storage. • Secure boot services. • Secure debug services. • Secure update services. • Digital Rights Management (DRM) protection. • Drivers for secure peripherals. • Management of cryptographic secrets, including client private keys for mutual TLS.
Reasoning	Ensure sensitive information and security operations are protected at the highest level.
Evidence	Trusted Execution Environment design review materials.
Threat Scenarios	TS1-5, TS18, TS30, TS31

ID	24MM.SEC.PLAT.TEE.2
Requirement	The TEE shall implement an authentication mechanism to validate the identity of normal world applications that access TEE services.
Reasoning	Secure services such as secure storage must be able to identify the normal world application to ensure that only that application can access its data.
Verification	Supplier to provide the design of their authentication mechanism to Toyota for review. Verify that only authenticated applications from normal world can access security functions residing in Trust Zone (Secure world).
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.TEE.3
Requirement	TrustZone shall be enabled for all Arm Cortex®-A microprocessors supporting the TrustZone architecture extension.
Reasoning	Ensure a TEE is available to provide security services.
Evidence	Trusted Execution Environment design review materials.
Threat Scenarios	TS1-5, TS18, TS30, TS31

ID	24MM.SEC.PLAT.TEE.4
Requirement	TrustZone shall be enabled for all SoC Arm Cortex®-M microprocessors supporting the optional TrustZone architecture extension.
Reasoning	Ensure a TEE is available to provide security services
Evidence	Trusted Execution Environment design review materials.
Threat Scenarios	TS1-5, TS18, TS30, TS31

ID	24MM.SEC.PLAT.TEE.5
Requirement	. Sensitive processing shall only take place in the TEE regardless of power state.
Reasoning	Ensure the same level of platform security is maintained and an attacker cannot compromise the vehicle software while processing sensor inputs, updating on-board software and/or other functions while in low power, ignition off mode.
Evidence	Trusted Execution Environment design review materials.
Threat Scenarios	TS1-5, TS18, TS30, TS31

4.7 Operating System

The Linux operating system provides many facilities and configurations to enable flexibility across a variety of workloads and to allow efficient diagnosis of issues. However, these facilities can be abused by an attacker to gain access to a system or easily broaden their access from an initial exploit. These requirements seek to remove unneeded facilities and configure the Linux kernel and operating system to minimize the attack surface. Some Linux security functionality is activated.

4.7.1 General

ID	24MM.SEC.PLAT.OS.GEN.1
Requirement	System startup parameters shall be integrity protected as part of secure boot to prevent modification. Startup parameters are any configuration values that can be provided during boot up to modify the operation of the ECU, including Linux kernel parameters, U-Boot environment, etc.
Reasoning	Altering system startup parameters can disable some security functions and ease adversarial access to the subject ECU.
Evidence	Supplier shall document how configuration of different boot stages is configured and secured from tampering.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.2
Requirement	The supplier shall analyze system settings under '/proc/sys' and set the most restrictive defaults possible.
Reasoning	Limits the scope and capability of attacks to be within the settings under '/proc/sys.'
Evidence	Secure boot design review materials.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.3
Requirement	The default Linux resource limits shall be set to the most restrictive values possible. Note: Resource limits are restrictions Linux places on the quantity of a variety of resources a process can consume. They can be adjusted using the 'ulimit' command, or the 'setrlimit' system call.

	Note: Finding the most restrictive values will require analysis by the supplier to identify what resources are required and the necessary margin.
Reasoning	Resource limits can reduce an attacker's scope and capability by forcing them to operate within the resources constraints already needed by the target process.
Evidence	Documentation of the rationale for the resource limits.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.4
Requirement	All core dumps shall be disabled with the operating system. Note: Multiple configurations may be needed to prevent core dumps by user applications, as noted above. Some configurations include: • limits.conf: Use: "* hard core 0" • sysctl.conf: "fs.suid_dumpable = 0" • profile: "ulimit -S -c 0 > /dev/null 2>&1"
Reasoning	Core dumps provide a wealth of information useful to an attacker when attempting to exploit a system. Removing core dumps may make exploitation more difficult.
Evidence	N/A
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.5
Requirement	The operating system shall support the Linux audit framework.
Reasoning	The audit framework could be used by the Vehicle Security Operations Center (VSOC) to log the occurrence of specific security events.
Evidence	Confirm the Linux kernel CONFIG_AUDIT options is on. Confirm the auditd daemon and related configuration files are present.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.6
Requirement	The Linux kernel pseudo random number generator (PRNG) shall be initialized using the SoC true random number generator (TRNG) on each boot. The Linux operating system and applications shall use the PRNG wherever possible.
Reasoning	Do not rely on transferring entropy between boot cycles. Instead, seed the PRNG using a strong random source at every boot.
Evidence	Documentation of the RNG initialization process.
Threat Scenarios	TS24

ID	24MM.SEC.PLAT.OS.GEN.7
Requirement	Any passwords or PINs shall be stored as a salted hash and not in plaintext.
Reasoning	Prevent an attacker from finding credentials through privileged filesystem access.
Evidence	List of all passwords or PINs and the hashing algorithm used to secure them.

Threat Scenarios	TS29

ID	24MM.SEC.PLAT.OS.GEN.8
Requirement	The operating system and kernel shall be configured to enable full randomization of user space using Address Space Layout Randomization (ASLR). Note: This applies ASLR to the user space software. Note: ASLR allows the Operating System to randomize the various parts of the application that are loaded into memory. Each time the program is run, the application is loaded into a different memory address. Note: Full randomization with ASLR can be configured by setting /proc/sys/kernel/randomize_va_space to 2.
Reasoning	Enabling ASLR can prevent an attacker from predicting the addresses of target software. This increases the difficulty or prevents an attacker from exploiting the software.
Evidence	N/A
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.9
Requirement	The operating system shall implement Kernel Address Space Layout Randomization (KASLR). Note: This applies ASLR to the kernel software.
Reasoning	KASLR makes it more difficult for an attacker to predict the addresses of kernel software. This increases the difficulty for an attacker to exploit the kernel.
Evidence	N/A
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.10
Requirement	The PRNG used for KASLR shall be seeded from a value derived by a TRNG.
Reasoning	A weak seed for KASLR could allow the attacker to predict the seed and thereby the memory mappings used by the kernel, bypassing KASLR. Note: The device tree ‘kaslr-seed’ value can be used to pass an initial seed from the bootloader.
Evidence	Documentation of the KASLR RNG source.
Threat Scenarios	TS24

ID	24MM.SEC.PLAT.OS.GEN.11
Requirement	Mandatory access control (MAC) shall be supported. Note: PCG (formerly VST) prefers SELinux for MAC due to its high degree of flexibility.

Reasoning	Can be used to restrict access to only the files, communications channels, and other resources necessary to perform the function. This limits an attacker's ability to widen the damage caused by an attack to other components of the system.
Evidence	N/A
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.GEN.12
Requirement	Mandatory access control shall operate in enforcing mode. Enforcing mode ensures the MAC policy is followed and not just analyzed for deviations.
Reasoning	If enforcing mode is not used, then MAC offers no protection, and an attacker may be able to access resources not needed by the target process to broaden the impact of an attack.
Evidence	N/A
Threat Scenarios	TS9-13

4.7.2 Minimize Attack Surface

ID	24MM.SEC.PLAT.OS.SFC.1
Requirement	All debug and development utilities and unused components shall be removed from all production software images. This includes: • Interpreters • Compilers • Integrated development environments (IDEs) • Debuggers • Assemblers • Protocol Analyzers • Programming analysis tools • Decompilers • Disassemblers • Port-scanners • Vulnerability analyzers • System analysis tools • Network diagnostic tools • Editors • Unused shells. • Unused application environments (Example: windowing environment in a non-GUI ECU) • Unused networking servers • Unused networking clients • Unused applications or libraries • Unused drivers • Unused protocol implementations • Unused kernel components • Unused kernel modules • Unused configuration • Unused debug facilities or services

	• Unused diagnostic facilities or services
Reasoning	Debug utilities left on the device may be helpful for an attacker when reverse-engineering the device or for furthering access to the ECU after exploitation.
Evidence	SBOM and unencrypted flash images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.SFC.2
Requirement	Live kernel patching tools shall be disabled and removed. Note: This requirement removes user space support for live-kernel patching. Kernel hardening requirements remove kernel-based support for live-kernel patching.
Reasoning	Live kernel patch tools can perform modifications to the running kernel. This can be used to introduce changes in the kernel that are not reflected in the Toyota and supplier approved firmware and verified in the secure boot process.
Evidence	SBOM and unencrypted flash images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.SFC.3
Requirement	All software binaries shall be stripped to remove unnecessary information. This includes debug information, note sections, etc.
Reasoning	Debugging information provides adversaries with extra, unneeded diagnostics concerning software design.
Evidence	Unencrypted flash images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.SFC.4
Requirement	All software binaries shall have all symbols removed that are not needed for relocation. Note: For libraries, non-public symbols must be removed. The compiler switch -fvisibility=hidden can be used to suppress non-public symbols.
Reasoning	Non-public symbols, for example C++ private methods revealing implementation details, must be suppressed.
Evidence	Unencrypted flash images.
Threat Scenarios	TS9-13

4.7.3 Users and Groups

ID	24MM.SEC.PLAT.OS.USR.1
Requirement	The Linux root user shall only be used when necessary. At all other times, processes and applications shall run as normal, non-root users. There shall be no other privileged users such as “admin”. Note: If a process or application requires elevated privileges, Linux capabilities can be used to grant the minimal set of privileges required.
Reasoning	Using non-root users forces an evaluation of the minimum capabilities needed for each process. It also provides more possibilities for MAC policies.
Evidence	List of all processes running as root and reason it is necessary.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.2
Requirement	The supplier shall remove unused users.
Reasoning	User accounts by themselves grant certain privileges in a system. The removal of unused accounts improves system security by applying defense in depth.
Evidence	List of all users and purpose for each.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.3
Requirement	The supplier shall remove unused groups.
Reasoning	Groups, like user accounts, convey certain privileges in a system. The removal of unused groups improves system security by applying defense in depth.
Evidence	List of all groups and purpose for each.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.4
Requirement	The supplier shall lock (disable login) all user accounts by listing an invalid password hash. Note: The impossible hash should be recognizable as a broken hash to the human eye, for example, the ‘!’ character.
Reasoning	There must not be the potential for interactive usage in the production configuration.
Evidence	Unencrypted filesystem dump.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.5
Requirement	The supplier shall lock (disable login) all groups.
Reasoning	There must not be the potential for interactive usage in the production configuration.
Evidence	Unencrypted filesystem dump.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.6
Requirement	The supplier shall change valid login shells to invalid shells. Note: This typically involves changing a shell from a normal shell like /bin/bash to a binary like /bin/false.
Reasoning	There must not be the potential for interactive usage in the production configuration.
Evidence	Unencrypted filesystem dump.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.USR.7
Requirement	The supplier shall remove binaries and scripts with built-in privilege escalation via suid and sgid facilities. Note: There are a small handful of binaries which may require privilege escalation by suid and sgid facilities. These are to be well documented and protected by the mandatory access control requirements of Software Sandboxing. In the evaluation phase, if TMC/TMNA detects that the file has the suid/sgid bit specified and is instructed to delete it, Teri1 should perform it.
Reasoning	If a component with built-in privilege escalation has a vulnerability, it can be used by an attacker to elevate privileges and increase the scope of the attack.
Evidence	List of all suid/sgid applications and reason it's necessary.
Threat Scenarios	TS9-13

4.7.4 Filesystem

ID	24MM.SEC.PLAT.OS.FS.1
Requirement	All filesystem mounts, except for the mount for "/dev", shall use the 'nodev' mount option. Note: This always applies to all user provided filesystems, such as removable media. All device files must be placed under "/dev".
Reasoning	Character and block-special files can be used to escalate privileges by an unprivileged user. Eliminate this attack path by disabling these special filetypes on filesystems that do not need them.
Evidence	Unencrypted filesystem images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.FS.2
Requirement	All filesystem mounts shall use the 'nosuid' mount option if suid/sgid applications should not exist. Note: This always applies to all user provided filesystems, such as removable media.

Reasoning	The nosuid mount option prevents suid (set-user-id) and sgid (set-group-id) binaries on the filesystem under discussion. Because suid binaries do not have the permission of the invoking user/group, and instead have the permission of the owner/owner-group, they must be maximally restricted to prevent privilege escalation by an adversary.
Evidence	Unencrypted filesystem images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.FS.3
Requirement	All filesystem mounts that should not have executable binaries shall restrict executable support using the 'noexec' mount option. Note: This always applies to all user provided filesystems, such as removable media.
Reasoning	Prevents binary execution on data-only filesystems.
Evidence	Unencrypted filesystem images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.FS.4
Requirement	The default umask shall be set to the most restrictive possible. Note: The recommended setting for umask is 027.
Reasoning	Having a restrictive umask makes new file creation secure by default, rather than relying upon an additional system call or shell command to change permissions.
Evidence	Documentation of the rationale for the umask setting.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.FS.5
Requirement	Files and directories shall be protected with Discretionary Access Control (DAC) using permissions that are as restrictive as possible. Note: Finding the most restrictive permissions will require analysis by the supplier to identify which files and directories must be accessible to each process.
Reasoning	In accordance with the principle of least privilege, discretionary access control must be used to provide a first line of defense against adversarial actors.
Evidence	Unencrypted filesystem images.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.FS.6
Requirement	The supplier shall apply system partitioning to separate logically separate areas of the filesystem, according to industry best practices. Note: https://refspecs.linuxfoundation.org/FHS_3.0/fhs-3.0.pdf

Reasoning	Separate partitions are an industry best-practice and contribute to higher security. They directly give additional security enforcement levers, and limit potential damage from aberrant processes (whether intentional or not).
Evidence	Unencrypted filesystem images.
Threat Scenarios	TS9-13

4.7.5 Linux Kernel Hardening

ID	24MM.SEC.PLAT.OS.KRN.1
Requirement	The Linux kernel shall be configured to use the Lockdown feature in both integrity and confidentiality modes. This configuration shall be set so Lockdown cannot be bypassed. (e.g., set configuration options SECURITY_LOCKDOWN_LSM_EARLY, LOCK_DOWN_KERNEL_FORCE_INTEGRITY and LOCK_DOWN_KERNEL_FORCE_CONFIDENTIALITY to True.).
Reasoning	Disable several mechanisms for modifying the running kernel image.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.2
Requirement	The Linux kernel shall be configured to activate the YAMA Linux security module. The module shall only support mode 3: “no attach”. Note: This will require modifying the Linux kernel source to set the default mode to 3.
Reasoning	Prevent the ability of processes to use ptrace to dump memory and manipulate execution of other processes.
Evidence	Final kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.3
Requirement	The Linux kernel shall be configured to only load signed kernel modules.
Reasoning	Prevent an attacker with root privileges from loading malicious kernel images not issued by Toyota.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.4
Requirement	The Linux kernel configuration shall disable COREDUMP. Note: The configuration is found in fs/Kconfig.binfmt.
Reasoning	Core dumps contain sensitive information from system memory that can be exploited by an adversary.

Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.5
Requirement	The Linux kernel configuration shall disable the HAVE_AOUT configuration. Note: The configuration is found in fs/Kconfig.binfmt.
Reasoning	The binary format a.out is deprecated and will not be present on the subject ECU. Eliminating the support for this format decreases the attack surface with no loss in functionality.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.6
Requirement	The Linux kernel configuration shall disable the BINFMT_MISC configuration. Note: The configuration is found in fs/Kconfig.binfmt.
Reasoning	The BINFMT_MISC configuration is used to support arbitrary executable formats (handled by user space applications). This configuration is not needed for production, and its elimination reduces the attack surface.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.7
Requirement	The Linux kernel configuration shall disable the DEBUG_KERNEL configuration. Note: The configuration is found in init/Kconfig.
Reasoning	The DEBUG_KERNEL configuration would add support for debugging to the kernel. Many low-level debugging constructs within the kernel depend on the DEBUG_KERNEL configuration. Eliminate this configuration to eliminate much debugging support in the kernel.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.8
Requirement	The Linux kernel configuration shall disable the USELIB configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Remove support for the uselib syscall; unused after libc5, unused with glibc.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.9
Requirement	The Linux kernel configuration shall disable the SYSFS_DEPRECATED configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable deprecated sysfs features for older user space tools.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.10
Requirement	The Linux kernel configuration shall disable the SYSFS_DEPRECATED_V2 configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable deprecated sysfs features for older user space tools.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.11
Requirement	The Linux kernel configuration shall disable the RELAY configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable this specialty purpose kernel-space to user-space memory copy facility, unused on the subject ECU, to reduce the attack surface.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.12
Requirement	The Linux kernel configuration shall disable the SYSCTL_EXCEPTION_TRACE configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable the /proc/sys/debug/exception-trace facility.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.13
Requirement	The Linux kernel configuration shall disable the SGETMASK_SYSCALL configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Remove this deprecated support for sys_sgetmask and sys_sgetmask system calls.
Evidence	Consolidated kernel configuration.

Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.14
Requirement	The Linux kernel configuration shall disable the SYSFS_SYSCALL configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Remove this deprecated support for the sys_sysfs system call in libc.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.15
Requirement	The Linux kernel configuration shall disable the ELF_CORE configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable elf format core dumps.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.16
Requirement	The Linux kernel configuration shall disable the PCSPKR_PLATFORM configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable internal pc-speaker support.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.17
Requirement	The Linux kernel configuration shall disable the KALLSYMS configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable loading debugging symbols into the kernel.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.18
Requirement	The Linux kernel configuration shall disable the USERFAULTFD configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable page fault handling in user space by the userfaultfd system call.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.19
Requirement	The Linux kernel configuration shall disable the PROFILING configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable extended profiling support mechanisms used by profilers.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.20
Requirement	The Linux kernel configuration shall disable the TRACEPOINTS configuration. Note: The configuration is found in init/Kconfig.
Reasoning	Disable empty function calls at tracepoints.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.21
Requirement	The Linux kernel configuration shall disable all trace configuration. Note: The configuration is found in kernel/trace/Kconfig.
Reasoning	Remove tracing support from the kernel to remove the kernel's ability to perform this type of tracing/profiling. It is an unacceptable security risk, and its removal decreases the attack surface.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.22
Requirement	The Linux kernel configuration shall disable the DEBUG_FS configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Disable this general-purpose debugging filesystem that can store arbitrary kernel data structures in a format accessible from the filesystem.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.23
Requirement	The Linux kernel configuration shall disable the DYNAMIC_DEBUG configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Disable kernel-level debug messages from inclusion in the subject ECU.

Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.24
Requirement	The Linux kernel configuration shall disable the DEBUG_TIMEKEEPING configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Disable timekeeping sanity checks.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.25
Requirement	The Linux kernel configuration shall disable the DEBUG_OBJECTS configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Disable kernel memory debugging support (unavailable without DEBUG_KERNEL).
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.26
Requirement	The Linux kernel configuration shall disable the STACKTRACE configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Disable kernel back trace support.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS9-13

ID	24MM.SEC.PLAT.OS.KRN.27
Requirement	The Linux kernel configuration shall disable the DEVMEM configuration. Note: The configuration is found in drivers/char/Kconfig.
Reasoning	The ‘/dev/mem’ provide access to physical memory, allowing an attacker with such access the ability to bypass many protections and read/write the memory of the kernel and other processes. This should not be necessary if drivers are properly implemented.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS8-13, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.OS.KRN.28
Requirement	If an exception is granted such that DEVMEM is enabled, the Linux kernel configuration shall enable the STRICT_DEVMEM configuration.

	Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Limit what types of memory '/dev/mem' can access.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS8-13, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.OS.KRN.29
Requirement	If an exception is granted such that DEVMEM is enabled, the Linux kernel configuration shall enable the IO_STRICT_DEVMEM configuration. Note: The configuration is found in lib/Kconfig.debug.
Reasoning	Enable IO_STRICT_DEVMEM to protect I/O ranges in-use by drivers from access by the root user.
Evidence	Consolidated kernel configuration.
Threat Scenarios	TS8-13, TS17, TS20, TS21, TS24, TS29, TS34, TS39

4.8 HLOS Secure Storage

ID	24MM.SEC.PLAT.STG.1
Requirement	HLOS secure storage shall encrypt the data it is given for storage using algorithms and key management according to sections 4.1 Cryptographic Algorithms and 4.2 Key Management. Note: This means that they HLOS secure storage keys are only accessible to the TEE, not to software in the HLOS. Note: This means that secure storage contents are "doubly encrypted", once by the secure storage mechanism, and then again by Full Disk Encryption (FDE). FDE alone is not sufficient to implement the encryption necessary for secure storage.
Reasoning	Must ensure cryptographic operations are secure so that data in HLOS secure storage is further protected from leakage.
Evidence	Secure storage design review materials.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.2
Requirement	HLOS secure storage shall implement rollback protection that prevents an attacker from replacing the encrypted data with a previous version. Note: This could be implemented using UFS/eMMC RPMB or similar means.
Reasoning	Must prevent attackers from using old data. An attacker may try to use old data to load vulnerable software or configuration that has been fixed.

Evidence	Secure storage design review materials.
Threat Scenarios	TS18, TS22, TS25, TS26

ID	24MM.SEC.PLAT.STG.3
Requirement	Items which cannot fit in RPMB sections shall be hashed and the hash stored either directly in the RPMB or with a group of hashes which is then hashed, and the group hash stored in RPMB.
Reasoning	Replay protected memory may have limited storage but a hash can provide assurances that the data is correct.
Evidence	Secure storage design review materials.
Threat Scenarios	TS1-5, TS9-13, TS18, TS22, TS23, TS25, TS26, TS30, TS31, TS35, TS36

ID	24MM.SEC.PLAT.STG.4
Requirement	The root key that protects secure storage contents shall be based on a cryptographic key unique to that device.
Reasoning	Prevent class attacks where disclosure of a secure storage key enables access to multiple devices.
Evidence	Secure storage design review materials.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.5
Requirement	The root key that protects secure storage contents shall be accessible only by hardware and cannot be read by any software.
Reasoning	Prevent attacker from obtaining or deriving secure storage key.
Evidence	Secure storage design review materials and SoC datasheets.
Threat Scenarios	TS21

ID	24MM.SEC.PLAT.STG.7
Requirement	The key used to encrypt data shall be unique for each application using the HLOS secure storage service.
Reasoning	Must ensure separation of sensitive information from between applications.
Evidence	Secure storage design review materials.
Threat Scenarios	TS21

ID	24MM.SEC.PLAT.STG.8
Requirement	Sensitive information in secure storage shall only be accessible to the Linux application that created it. The access control mechanism shall prevent a compromised application from spoofing its identity and accessing another application's sensitive information. Example: The media application uses OAuth tokens to access 3rd party streaming services. The messaging application also uses OAuth tokens to access 3rd party messaging services. Both the media and messaging application store their OAuth tokens into secure storage. However, the media application can only access its OAuth tokens and cannot access the OAuth tokens stored by the messaging application, and vice-versa.
Reasoning	Prevent a compromised application from accessing sensitive information from another application.
Evidence	Secure storage design review materials, including description of the access control mechanism.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.9
Requirement	The HLOS secure storage implementation shall determine the identity of the application calling the API internally. It shall not rely on an identity provided by the application through the API.
Reasoning	If an application is trusted to provide its own identity, then an attacker could compromise an application and spoof the identity information to access the sensitive information of another application.
Evidence	Secure storage design review materials, including description of the access control mechanism.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.10
Requirement	HLOS secure storage shall expose a common application programming interface (API) that all applications can use to secure sensitive information.
Reasoning	A system-wide implementation of HLOS secure storage will provide more robustness and security. Multiple implementations can lead to one or more insecure implementations.
Evidence	Secure storage design review materials and API.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.11
Requirement	The HLOS secure storage API shall support a compatible programming language for all the software that needs to secure sensitive information. Note: This includes, at a minimum, a C API.
Reasoning	Having native API support for each language will prevent insecure workarounds or alternatives.
Evidence	Secure storage design review materials and API.
Threat Scenarios	TS20, TS21, TS24, TS29

ID	24MM.SEC.PLAT.STG.12
Requirement	The HLOS secure storage API shall expose as few configuration and security settings as possible and instead determine secure settings itself.
Reasoning	Remove as much opportunity for misconfiguration as possible, which could lead to insecure use of the API and leakage of sensitive information.
Evidence	Secure storage design review materials and API.
Threat Scenarios	TS20, TS21, TS24, TS29

4.9 Full Disk Encryption

ID	24MM.SEC.PLAT.FDE.1
Requirement	All data on the ECU written to any external, persistent storage shall be encrypted. As soon as possible, SoC Full Disk Encryption facilities shall be used to implement encryption. Note: Until SoC Full Disk Encryption can be activated, some early boot images may have unencrypted headers or metadata. This is acceptable as long as that information does not contain sensitive information and the contents of the image are encrypted. Exception: Qualcomm UIE is too slow to encrypt all boot images covered by UIE. See 24MM.SEC.QC.FDE.3.
Reasoning	Full disk encryption increases the difficulty of reverse-engineering and provides a layer of protection to sensitive information mistakenly stored outside of secure storage.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.FDE.2
Requirement	Full disk encryption shall encrypt all data stored externally to the SoC in a manner that is transparent to general software applications.

Reasoning	Making encryption transparent decreases the chances of some data mistakenly bypassing encryption.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.FDE.3
Requirement	Inline encryption features of the SoC shall be used to implement full disk encryption where available. Alternative implementations include dm-crypt [4]. Note: Use of in-line encryption may require specific types of flash such as UFS.
Reasoning	Inline encryption provides the best performance while still meeting the security goals.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.FDE.4
Requirement	The performance impact of full disk encryption shall be less than 3%.
Reasoning	Performance must still be acceptable to meet functional goals.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.FDE.5
Requirement	The full disk encryption key shall be unique for each unit.
Reasoning	Prevent class attacks where disclosure of a common encryption key enables compromise of multiple devices.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS21

ID	24MM.SEC.PLAT.FDE.6
Requirement	The full disk encryption algorithm shall be AES-XTR or AES-GCM.
Reasoning	Encryption algorithms used for full disk encryption must not have low diffusion. Using low diffusion algorithms on substantial amounts of data makes cryptanalytic attacks more feasible.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.FDE.7
Requirement	The full disk encryption key length shall be at least 128 bits and the maximum length that still meets performance targets.
Reasoning	Protect data as long as possible from cryptanalytic attacks.
Evidence	Full Disk Encryption design review materials.

Threat Scenarios	TS20, TS21

ID	24MM.SEC.PLAT.FDE.8
Requirement	The full disk encryption key(s) shall be managed according to section 4.2 Key Management.
Reasoning	Ensure an attacker cannot read or modify the encryption key.
Evidence	Full Disk Encryption design review materials.
Threat Scenarios	TS21

4.10 SoC Memory Access Configuration

ID	24MM.SEC.PLAT.SOC.1
Requirement	The SoC shall be configured to restrict each processor's access to only the intended memory-mapped peripherals. Example: if a real time processor is intended to use a serial port, then the application cores shall not be allowed to access that serial port's peripheral memory range.
Reasoning	If an attacker gains privileged access to a processor, they should not be able to interact with peripherals or SoC functions not intended for them. This limits the damage an attacker can perform to the system.
Evidence	Listing of each peripheral and which processors can access it.
Threat Scenarios	TS43, TS45, TS46, TS50, TS54-56, TS58, TS62, TS65, TS68, TS69, TS74

ID	24MM.SEC.PLAT.SOC.2
Requirement	The SoC shall be configured to restrict each CPU's access to only the intended memory ranges. For example, an application processor shall not be able to read or write the private memory of a real time processor.
Reasoning	If an attacker gains privileged access to a CPU, they should not be able to interact with memory not intended for them. This limits the damage an attacker can perform to the system.
Evidence	Listing of memory ranges accessible by each processor.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.SOC.3
Requirement	The SoC shall be configured to restrict the memory access of peripherals with bus master capabilities. The peripherals shall only be allowed to access the necessary memory range.
Reasoning	Prevent PCIe or other peripherals from accessing private memory not intended for that peripheral's use.
Evidence	Listing of each peripheral that has bus master capabilities and the allowed memory ranges.

Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.SOC.4
Requirement	The SoC memory controls shall be modified or updated only by the Trusted Execution Environment or by normal world software operating with privileges above the rich OS (e.g., Linux).
Reasoning	Prevents an attacker with privileged access to normal world, such as a Linux kernel exploit, from removing the memory controls.
Evidence	Documentation of the implementation and configuration mechanisms for SoC memory controls.
Threat Scenarios	TS9-13

4.11 Logging

Some terms and their definitions as used in these requirements are:

Term	Description
Logging Ecosystem	Entire vehicle logging architecture. This includes generating log records on the vehicle, shipping logs to the backend database, and distributing allowed logs to authorized entities.
Logging Subsytem	One of potentially several logging frameworks on an ECU. Examples: syslog, journald, Python logging, log4cxx, or custom log frameworks. Sometimes these frameworks are combined. For example, log4cxx can send logs directly to a file or into the syslog daemon.
Backend Log Database	An out-vehicle, Toyota backend service for accumulating logs from vehicles and distributing them to authorized parties.
Logging Configuration	Collection of settings that control how the logging subsystems behave and what data is collected.

4.11.1 General

`

ID	24MM.SEC.PLAT.LOG.GEN.1
Requirement	Every software component on the ECU shall have access to a logging subsystem that meets the requirements of this document. This includes software or scripts written in different programming languages including C, C++, Shell scripts, Python, and Java.
Reasoning	All necessary logs must be collected, no matter the software component that is generating them. Avoid situations where a valuable log is not generated because a log subsystem is not available for that type of software.
Evidence	Listing of the log subsystem used by each software component.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.PLAT.LOG.GEN.2
Requirement	The ECU log architecture shall be designed so that it can never take up so much processing or IO resources that it degrades the performance of core ECU functionality or critical functions such as safety and vehicle operations. The user shall not be able to observe any differences in performance (UI lag, storage space, etc.) due to a change in the log configuration. Note: This includes handling scenarios where an attacker repeatedly performs an action to flood the logging system with log messages. Note: This may require rate-limiting and deduplication of logs.
Reasoning	Must prevent an attacker from repeatedly performing a malicious action to flood the logging subsystem and cause a denial or degradation of service.
Evidence	Log design review materials.
Threat Scenarios	TS7, TS15, TS16

ID	24MM.SEC.PLAT.LOG.GEN.3
Requirement	All log appenders, also known as ‘sinks’ or ‘destinations’, shall be disabled or removed if they are not used in production.
Reasoning	For embedded development, external devices including system consoles, displays and serial consoles and auxiliary log files are often used to facilitate development and debugging processes. The requirement to disable or remove these configurations for production will protect against an adversary easily gaining access to diagnostic information that should otherwise remain protected or confidential.
Evidence	Unencrypted flash images.
Threat Scenarios	TS39

ID	24MM.SEC.PLAT.LOG.GEN.4
Requirement	The ECU log subsystems shall all use a time source accurate to at least 1 millisecond. Note: To prevent timing drift, the ECU will likely need to synchronize periodically with a well-known time source, such as an NTP server or a GPS receiver.
Reasoning	It is critical during forensic analysis to have an accurate time that each log event took place. This will allow proper ordering of events across ECUs and out-vehicle systems.
Evidence	Access to read current time from the same time source used for logging.
Threat Scenarios	TS35

4.11.2 Configuration

ID	24MM.SEC.PLAT.LOG.CFG.3
Requirement	The logging ecosystem shall be designed so that changes to the logging configuration can only occur through an intended mechanism. A malicious software component cannot affect the log configuration of another software component on the ECU or the global log configuration of the ECU. Where possible, each software component should also not be able to modify its own logging configuration. Note: This may require the settings in the logging configuration to be set explicitly in a file-based configuration that is integrity protected by secure boot. This prevents parameters based on default values being overridden by a malicious configuration file. Note: This may require disabling dynamic logging configuration interfaces. Note: If the updateable logging configuration is corrupted, the logging configuration must default to a known default configuration and not disable logging altogether.
Reasoning	A malicious software component should not be able to manipulate the logging subsystems to hide evidence of its behavior. By limiting log configuration changes to only authorized mechanisms, discrepancies between the actual log configuration and expected log configuration can be prevented.
Evidence	Logging design review materials.
Threat Scenarios	TS35

ID	24MM.SEC.PLAT.LOG.CFG.6
Requirement	The ECU logging architecture shall be designed so that log configuration changes can never leave the logging subsystem configuration in an unknown or unapproved state.
Reasoning	It is important that the logging state of the vehicle is always known and is accurate. Otherwise log events may be missed or too much data may be collected.
Evidence	Logging design review materials.
Threat Scenarios	TS35

4.11.3 Access Control

ID	24MM.SEC.PLAT.LOG.ACC.2
Requirement	The logging ecosystem shall be designed so that only authenticated and authorized users can view log records.
Reasoning	Ensure attackers cannot access sensitive log data.
Evidence	Logging design review materials.
Threat Scenarios	TS39

4.11.4 Confidentiality and Integrity

ID	24MM.SEC.PLAT.LOG.CI.1
Requirement	Log records shall be encrypted at rest. Note: The encryption mechanism may use the full disk encryption implementation specified in 2.3.2 Encrypted Flash.
Reasoning	By its nature, logging contains confidential information which is proprietary and may contain trade secrets. The disclosure of this data itself could be damaging to Toyota and its suppliers and affiliates. Moreover, this data could be a vector to additional intrusion into the vehicle.
Evidence	Logging design review materials.
Threat Scenarios	TS39

ID	24MM.SEC.PLAT.LOG.CI.2
Requirement	All log records should be stored using a cryptographic integrity protection mechanism. This mechanism shall allow an authorized user to examine the stored logs records and determine if log entries have been removed or modified.
Reasoning	Must ensure log records can be trusted and that we can always confirm no logs have been dropped or modified.
Evidence	Logging design review materials.
Threat Scenarios	TS35

ID	24MM.SEC.PLAT.LOG.CI.3
Requirement	All log records shall be protected from unauthorized reading and modification on the running system. No application or software process shall be able to read, modify, or delete the logs of another application or process. Note: This can be implemented using OS access control mechanisms. For example: • Filesystem Discretionary Access Control (DAC) • SELinux policy • Filesystem namespaces
Reasoning	By its nature, logging contains confidential information which is proprietary and may contain trade secrets. The disclosure of this data itself could be damaging to Toyota and its suppliers and affiliates. Moreover, this data could be a vector to additional intrusion into the vehicle.
Evidence	Logging design review materials.
Threat Scenarios	TS39

4.11.5 Transfer and Retention

ID	24MM.SEC.PLAT.LOG.TR.3
Requirement	Log records shall be transferred within a mutual-TLS session between the vehicle and the backend log database according to section 4.13.1 TLS.

Reasoning	Log data is sensitive and must be secured in transit over the Internet.
Evidence	Documentation, software, and hardware necessary to perform upload of log records to the backend log database.
Threat Scenarios	TS39

ID	24MM.SEC.PLAT.LOG.TR.6
Requirement	Log records shall be retained for a maximum period approved by Toyota. After the retention period, log records shall be deleted from the ECU at the first opportunity.
Reasoning	Limit aggregation of sensitive information on the ECU to minimize impact of disclosure.
Evidence	Logging design review materials.
Threat Scenarios	TS39

4.12 Certificate Management

ID	24MM.SEC.PLAT.CRT.1
Requirement	The certificate manager shall maintain a certificate authority (CA) store that establishes the root of trust for certificate verification.
Reasoning	Need list of root CAs to perform certificate verification.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.CRT.2
Requirement	The CA store maintained by the certificate manager shall be limited to the minimum set required to enable necessary connectivity.
Reasoning	The more root CAs that are trusted, the more likely that one of the Root CAs could be compromised and used by an attacker to create malicious network connections. Minimize trusted root CAs to minimize likelihood one of them is compromised.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.CRT.4
Requirement	The certificate manager shall provide a common implementation of certificate revocation verification according to section 4.13.5 Certificate Revocation Checks that shall be used by all applications. This could be implemented as a common software library that is included in each software application or as a system service that is available to every software application. Note: this common implementation must be used by all applications including Chromium.
Reasoning	Providing a common certificate revocation verification solution eliminates security gaps in different implementations and maximizes caching benefits.

Evidence	Certificate management design review materials.
Threat Scenarios	TS59-64

4.13 Communications

4.13.1 TLS

ID	24MM.SEC.PLAT.COM.TLS.1
Requirement	The ECU shall support only TLS versions 1.3.
Reasoning	Most recent TLS version removes insecure ciphers and enhances security to protect against attackers from tampering with network communication.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.TLS.2
Requirement	The ECU shall not support TLS or SSL versions prior to TLS 1.3.
Reasoning	TLS downgrade attacks should be protected against to prevent earlier insecure versions of TLS which can lead to communication tampering.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.TLS.3
Requirement	Authentication of the TLS endpoint shall always be enforced by validating that the certificate chains to a root certificate in the trusted certificate store.
Reasoning	Mutual authentication for TLS prevents attackers from performing man-in-the-middle attacks to tamper with network communication.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.TLS.4

Requirement	TLS handshakes shall restrict cipher suites to the following:
	• TLS 1.3
	  ○ TLS_AES_256_GCM_SHA384
	  ○ TLS_CHACHA20_POLY1305_SHA256

Reasoning	Restricting cipher suites prevents attackers from using insecure or outdated cipher suites to read and/or tamper with sensitive information.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.TLS.5
Requirement	Each unit shall have a unique client certificate that is used for mutual TLS.
Reasoning	Must be able to uniquely identify each unit to allow fine-grained permissions and revocation.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.TLS.6
Requirement	The client certificate shall be issued by Toyota and signed by a certificate chain ending with a Toyota Certificate Authority.
Reasoning	Must be able to authenticate that the client certificate is trusted by tracing to a Toyota Certificate Authority.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.TLS.7
Requirement	The ECU should support replacing the client certificate and keys by generating new keys and a corresponding Certificate Signing Request (CSR) that is sent to Toyota for signing.
Reasoning	Sometimes it is necessary to rotate or update keys in the field. By generating the keys on the device, the chance of leaking the key is minimized since the private portion never has to be transported.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.TLS.8
Requirement	Any client key and client CSR generation and management shall be performed in the TEE or SE.
Reasoning	Must ensure private key information is protected from disclosure by using the TEE.
Evidence	Client key generation design review materials.
Threat Scenarios	TS21

ID	24MM.SEC.PLAT.COM.TLS.9
Requirement	Any Certificate Signing Requests (CSRs) sent to Toyota for signing shall conform to Toyota Private Certificate Authority (TPCA) attribute and value definitions. Note: Definition of the TPCA shall be provided in a future appendix to this specification or the S-190 specification.
Reasoning	Must ensure certificates are issued with a common configuration that includes all needed fields and values.
Evidence	N/A
Threat Scenarios	TS59-64

24MM.SEC.PLAT.COM.TLS.10-13 removed. TLS 1.2 fallback no longer supported.

4.13.2 DNS

ID	24MM.SEC.PLAT.COM.DNS.1
Requirement	The ECU shall support authenticated DNS using DNSSEC according to RFCs 4033, 4034, 4035.
Reasoning	Protects manipulation of DNS records on untrusted networks.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.DNS.2
Requirement	The ECU shall support DNS over HTTPS according to RFC 8484.
Reasoning	Protects observation of hostnames contacted by ECU.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.DNS.3
Requirement	The ECU shall use authenticated DNS when resolving Toyota or 3rd party backend services.
Reasoning	When using untrusted network interfaces such as Wi-Fi, an attacker may try to manipulate DNS responses to send the network traffic to a malicious destination.
Evidence	N/A
Threat Scenarios	TS59-64

4.13.3 MPTCP

24MM.SEC.PLAT.COM.MPTCP.1-2 removed. MPTCP not supported.

4.13.4 Publish/Subscribe Systems (MQTT)

ID	24MM.SEC.PLAT.COM.MQTT.1
Requirement	The ECU shall authenticate with the messaging broker using mutual TLS 1.3 for any Publish and Subscribe system such as MQTT.
Reasoning	Identifying the subscriber is key to protecting the MQTT broker and the topics.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.MQTT.3
Requirement	The ECU shall implement end to end encryption for the payload of any Publish and Subscribe system such as MQTT.
Reasoning	End to end encryption provides confidentiality of the message from publisher to subscriber and any third party in between such as the broker cannot read the message.
Evidence	N/A
Threat Scenarios	TS68

4.13.5 Certificate Revocation Checks

It is important to verify that certificates have not been revoked. The following flow diagram shows the high-level actions and fallback steps.

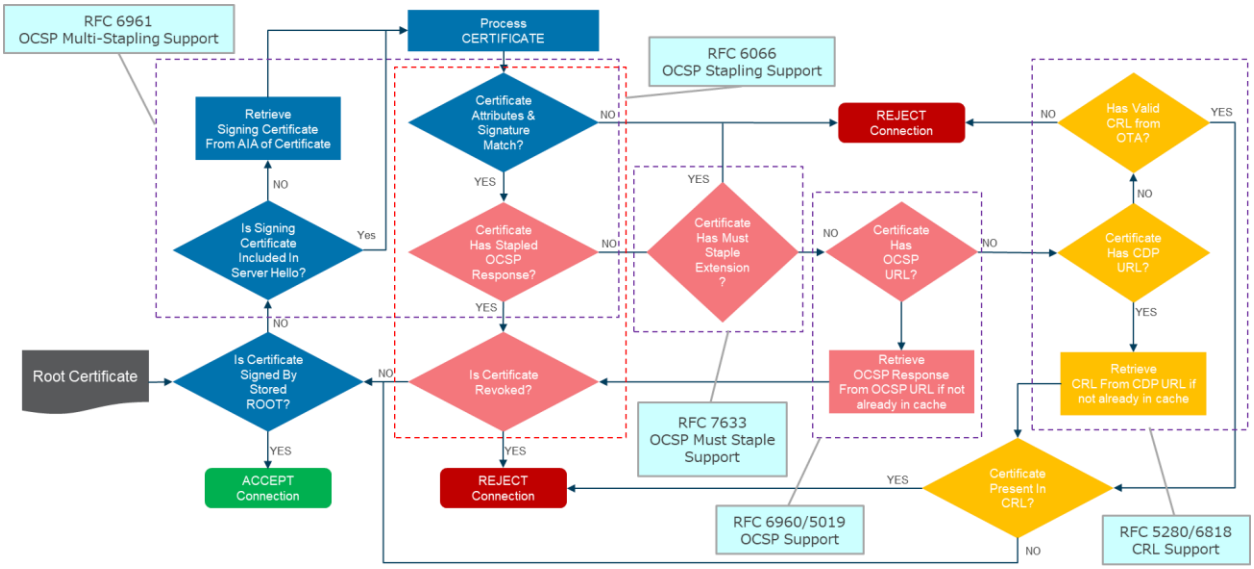

Figure 7 - Certificate verification flow

This section seeks to be compliant with the certificate revocation checks specified in the Certificate Management section in the Common Specification for the Communication Interface between In-Car and Out-Car.

ID	24MM.SEC.PLAT.COM.REV.1
Requirement	The ECU shall follow the overall certificate verification flow shown in Figure 7.
Reasoning	The certificate verification flow must support backup verification methods in case the server does not support the primary verification mechanisms.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.2
Requirement	The ECU shall use Online Certificate Status Protocol (OCSP) according to RFC 6960 and RFC 5019 as the primary certificate revocation checking mechanism.
Reasoning	OCSP is a well-supported and efficient (see requirements below) mechanism for checking the status of a certificate.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.3
Requirement	The ECU shall support OCSP stapling (RFC 6066). The ECU shall always include the status_request extension during the TLS handshake. The ECU shall validate the OCSP

	response is valid and the certificate is good. The ECU shall always use the stapled OCSP response if one is supplied.
Reasoning	OCSP stapling is the most efficient mechanism for verifying the certificate status.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.4
Requirement	The ECU shall support OCSP must staple (RFC 7633). The ECU shall enforce that an OCSP response must be supplied if the must staple certificate option is present and hard fail if it is not present.
Reasoning	OCSP must staple can be used to determine if a stapled OCSP response is being maliciously removed during the TLS handshake.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.6
Requirement	The ECU shall validate every certificate in the certificate chain to roll up to the root certificate stored in the trusted certificate store. This includes the server certificate and all intermediate certificates
Reasoning	Intermediate certificates could also be revoked. If so, their revocation must be detected.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.7
Requirement	When stapled OCSP is not supported by a server for one or more certificates in the certificate chain, the ECU shall perform online collection and validation of OCSP responses directly from OCSP responders (“direct” OCSP).
Reasoning	Not all servers support stapled OCSP. Fallback to direct OCSP.
Evidence	N/A.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.8
Requirement	The ECU shall support multiple OCSP URLs when performing “direct” OCSP. If the connection to an OCSP URL fails (e.g., times out) the ECU shall retry using the next OCSP URL in the list until a valid response is collected or the list is exhausted.
Reasoning	If multiple OCSP URLs are provided, other URLs should be used if there is an error connecting to the first URL to provide robustness to failures or outages.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.9
Requirement	The ECU shall cache OCSP responses for use by all applications.
Reasoning	Maximize performance of revocation checking by creating a common cache of OCSP responses usable by all applications.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.10
Requirement	The ECU shall support the Cache-Control HTTP header to inform how long to cache an OCSP response (RFC 5019). Note: Cache-Control information includes the max-age time. It must be ensured that max-age requirement is met.
Reasoning	This gives the server more control on how long devices will cache OCSP responses
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.11
Requirement	The ECU shall cache an OCSP response no longer than the minimum of the cache-control max-age directive and the nextUpdate field of the OCSP response
Reasoning	Check for OCSP updates as often as specified by the OCSP responder.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.12
Requirement	The ECU shall support the ETag HTTP header to only download updated OCSP responses when necessary (RFC 5019). Note: E-Tag is a string that identifies a particular version of the associated data. This is the ASCII Hex representation of SHA1 hash of OCSP response.
Reasoning	Reduces load on the OCSP responder by only needing to download updated OCSP responses.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.13
Requirement	When OCSP is not supported by a server, the ECU shall support validating against revocation lists already installed on the ECU. Note: These revocation lists are installed and maintained separately from application CRL queries. These revocation lists could include typical CRLs, Google's CRLSet, or similar mechanism as specified by Toyota.
Reasoning	Some CRL lists may be installed at the factory and updated via the update mechanism. If applicable, these CRL lists shall be used.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.14
Requirement	If installed revocation lists are not applicable, the ECU shall download and verify CRLs.
Reasoning	Some servers only support CRLs.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.REV.15
Requirement	The ECU shall cache CRLs for use by all applications.
Reasoning	Maximize performance of revocation checking by creating a common cache of CRLs usable by all applications.

Evidence	N/A
Threat Scenarios	TS59-64, TS68

4.13.6 Firewall

Network firewall refers to an IP-based firewall such as IPTables in Linux.

ID	24MM.SEC.PLAT.COM.FWL.1
Requirement	The ECU shall block all inbound network connections from outside the vehicle.
Reasoning	*Open networks ports can be utilized by attackers to gain access to the ECU*
Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.2
Requirement	The network firewall shall employ a default DROP policy for all built-in chains. Note: This is the Linux iptables DROP target for INPUT, OUTPUT, and FORWARD chains. This requirement is recommended.
Reasoning	A default drop policy silently drops packets that have not been authorized. This policy maximally protects the subject ECU and does not provide additional information useful to adversaries.
Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.3
Requirement	The network firewall shall only permit inbound and outbound traffic rules necessary to implement system requirements / user stories. Note: This is the Linux iptables ACCEPT target. This requirement is recommended.
Reasoning	To prevent remote attacks, a restrictive firewall policy is necessary. By restricting inbound and outbound traffic to the minimum set needed to implement requirements / user stories, network-based attack paths are minimized.

Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.4
Requirement	The network firewall shall only permit traffic from expected IP addresses and shall drop all others. This shall be configured specifically for each network service/port.
Reasoning	Prevent a compromised ECU with an unchanged IP address from using services or other ECUs it is not intended to use.
Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.5
Requirement	The network firewall shall support stateful packet inspection. Note: This is the Linux netfilter -> iptables -> conntrack functionality.
Reasoning	By enabling stateful packet inspection, the subject ECU can make more sophisticated determinations of packet travel validity. This functionality eliminates entire classes of network vulnerabilities based on IP protocol exploits.
Evidence	Linux kernel configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.6
Requirement	The network firewall shall omit support for multi-port TCP/UDP protocols like FTP. Note: This is the Linux netfilter -> iptables -> conntrack -> RELATED. Also note, this rule does not prohibit the RELATED functionality for ICMP protocol errors, if this functionality is required
Reasoning	FTP is not a supported protocol, due to lack of security. Other protocols on the subject ECU do not use multiple ports, apart from ICMP protocol errors.
Evidence	Linux kernel configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.7
Requirement	The network firewall shall not blanket allow any of the following: • Address ranges • Port ranges • Network interfaces • Protocols.
Reasoning	Ranges in the firewall configuration may expand access beyond the minimum set required to implement use cases.
Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

ID	24MM.SEC.PLAT.COM.FWL.8
Requirement	The network firewall shall drop ICMP requests.
Reasoning	Dropping ICMP requests prevents attackers from determining IP address of the ECU.
Evidence	Default firewall configuration.
Threat Scenarios	TS47-53, TS59-64

4.13.7 Wi-Fi

ID	24MM.SEC.PLAT.COM.WIFI.1
Requirement	The Wi-Fi interface shall support WPA2 and WPA3 modes.
Reasoning	WPA3 provides the strongest security for standard Wi-Fi connections. WPA3 uses strong encryption standards and supports forward secrecy. Enabling the WPA2 and WPA3 mode can prevent the attacker from recovering the passwords from data or cracking the password by guessing. Thus, mitigates the brute-force attacks.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.WIFI.2
Requirement	The Wi-Fi interface shall not support TKIP modes.
Reasoning	Note: There are known vulnerabilities with TKIP modes and industry guidance is to disable it. [5] Ensuring that the TKIP modes are disabled can prevent an attacker from getting network access and decrypting the data. Thus, it mitigates the data confidentiality from being compromised.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.WIFI.3
Requirement	The Wi-Fi interface shall be configured with a random MAC address at every boot based on a PRNG seeded by a TRNG. Note: This can be done using the SoC true random number generator (TRNG) on each boot.
Reasoning	Enabling this feature can prevent the attacker from tracking or identifying the customer through Wi-Fi MAC and thus provides an additional layer of privacy.
Evidence	Documentation describing how MAC address randomization is implemented.
Threat Scenarios	TS29

ID	24MM.SEC.PLAT.COM.WIFI.4
Requirement	The Wi-Fi default passphrase shall be created using at least 64-bits of effective randomness based on a PRNG that is seeded by a TRNG.
Reasoning	Enabling this feature makes it difficult for attacker to guess the default passphrase and prevents from using passphrase from one device to connect to another
Evidence	Documentation describing how default passphrase is generated.
Threat Scenarios	TS29

ID	24MM.SEC.PLAT.COM.WIFI.5
Requirement	The Wi-Fi interface shall store SSID, passphrase, and any secret cryptographic material in HLOS secure storage as described in section 4.8 HLOS Secure Storage.
Reasoning	Ensures an attacker cannot access Wi-Fi authentication information or other cryptographic material through physical access to the ECU.
Evidence	N/A
Threat Scenarios	TS20, TS21

4.13.8 Bluetooth

ID	24MM.SEC.PLAT.COM.BLT.1
Requirement	The Bluetooth interface shall support Security Mode 4 Level 4 for Bluetooth BR, EDR and High Speed (HS) connections. Note: Security mode 4 and Service level 4 mandates MitM protection with strong encryption using a 128-bit key generated using FIPS-approved Advanced Encryption Standard (AES) encryption.

Reasoning	Ensuring that the Bluetooth BR, EDR and HS connections uses the security mode 4 and service level 4 mitigate eavesdropping, unauthorized access, and Man-in-the-Middle attacks.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.BLT.2
Requirement	The Bluetooth interface shall support Security Mode 1 Level 4 for Bluetooth LE connections. Note: Bluetooth LE Security mode 1 Service level 4 mandates MITM protection with strong encryption. This feature requires authenticated low energy Secure Connections pairing with Elliptic Curve Diffie-Hellman (ECDH) based encryption.
Reasoning	Ensuring that the Bluetooth LE uses security mode 1 and service level 4 mitigate the Man-in-the-Middle attacks.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.PLAT.COM.BLT.3
Requirement	The Bluetooth interface shall NOT support the Just Works pairing mode. The ECU shall fail the pairing request for Just Works pairing. If that is not possible, the ECU shall forcibly disconnect within 1 second after Just Works pairing is completed without accepting any communications. Note: The main feature of Just Works pairing mode is that it does not require any authentication to complete a pairing procedure.
Reasoning	Ensuring that the Just Works pairing mode of Bluetooth is disabled helps prevent attackers from intercepting and manipulating all communications. Thus, mitigates Man-in-the-Middle attacks.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.PLAT.COM.BLT.4
Requirement	The Bluetooth interface shall support resolvable private Bluetooth addresses and shall rotate addresses every 15 minutes or less.
Reasoning	Enabling this feature can prevent the attacker from tracking or identifying the customer through Bluetooth MAC and thus provides an additional layer of privacy.
Evidence	N/A

Threat Scenarios	TS29

ID	24MM.SEC.PLAT.COM.BLT.5
Requirement	The Bluetooth interface shall store Bluetooth Link Keys in HLOS secure storage as described in section 2.5.2
Reasoning	Enabling this feature makes it difficult for an attacker to access Bluetooth Link Keys through physical access to the MM device.
Evidence	N/A
Threat Scenarios	TS20, TS21

4.13.9 NFC

Requirements removed.

4.13.10 In-Vehicle Communications

In-Vehicle communication encompasses wired communication performed inside the vehicle over CAN, USB, and Ethernet.

The following table provides high level security requirements for communication over these wired interfaces.

ID	24MM.SEC.PLAT.COM.INT.1
Requirement	The USB interface shall only support devices needed for production use cases. Example: The USB interface shall not support USB to ethernet adapters.
Reasoning	Reasoning: Limiting support for USB devices prevents attackers from gaining unauthorized access to the ECU through unnecessary features.
Verification	List of USB devices supported by ECU.
Threat Scenarios	TS50-53, TS57

ID	24MM.SEC.PLAT.COM.INT.2
Requirement	The CAN interface shall only transmit and receive the necessary CAN IDs to perform the intended functions of the ECU. This shall be enforced by transmit and receive whitelists for CAN IDs.
Reasoning	CAN whitelisting features provide another layer of defense against attackers trying to use one compromised component to attack another device.
Evidence Evidence	List of CAN frames the ECU can send.

Threat Scenarios	TS47-53

ID	24MM.SEC.PLAT.COM.INT.3
Requirement	The whitelist feature of the CAN interface shall be implemented in every SoC in the CAN transmission and reception path.
Reasoning	Prevent an attacker from transmitting on the CAN interface to unintended CAN IDs in the event the main SoC is compromised.
Evidence	CAN whitelist.
Threat Scenarios	TS47-53

4.14 Peripherals

Peripheral communication encompasses circuit board level communication performed between SoCs, SoCs and external memory components, and SoCs and external debugging hardware. It also includes sensors or IO devices connected to the ECU.

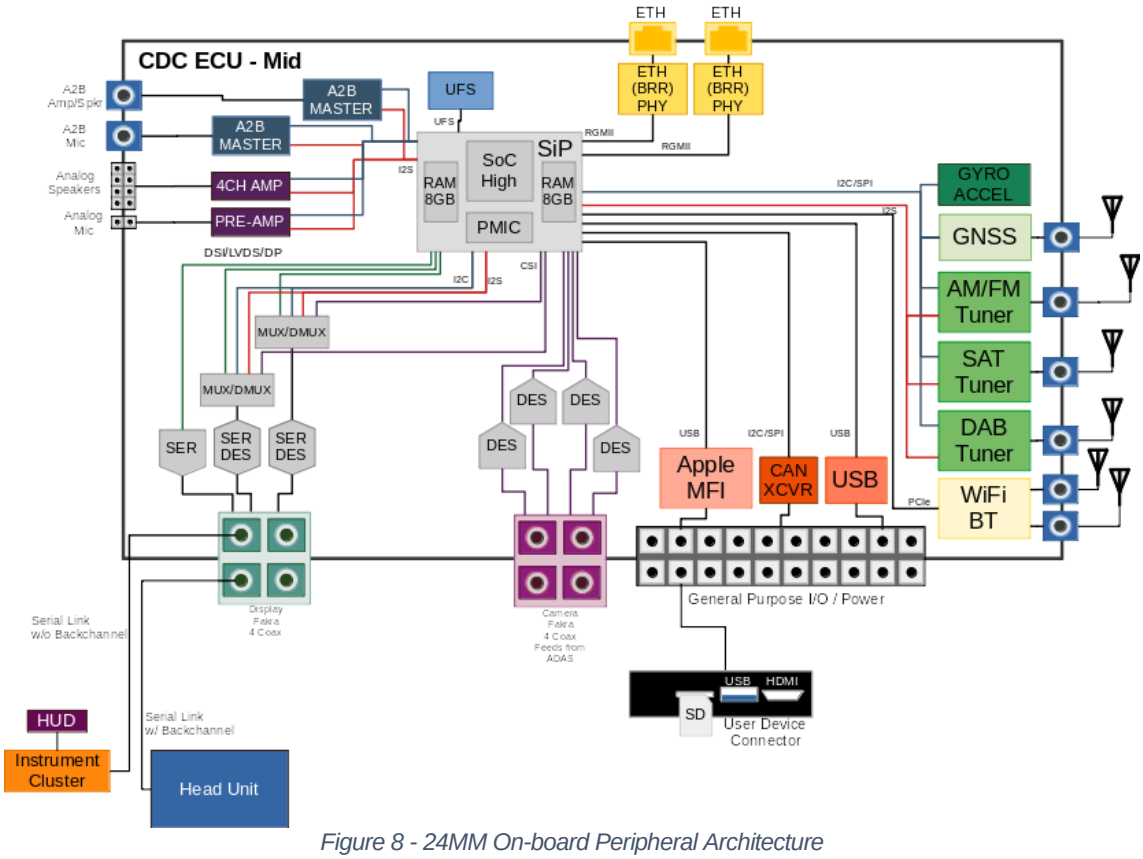

Figure 8 - 24MM On-board Peripheral Architecture

Attackers with physical access to the ECU will attempt to exfiltrate sensitive information from SoCs by monitoring and tampering with communication over peripheral buses.

The following table provides high level security requirements for communication between on-board peripherals.

ID	24MM.SEC.PLAT.PER.1
Requirement	The ECU shall disable all console access to the SoC, such as a Linux command-line interface via serial or SSH.
Reasoning	Prevents attackers from gaining access to SoC filesystem and operation.
Evidence	Board schematic and PCB layout.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.PER.2
Requirement	Peripheral bus communications between processors on the ECU shall be encrypted when transmitting sensitive information. This requirement applies only to communication between the SoC and TA100.
Reasoning	Encrypting bus level communication prevents attackers from reading security sensitive information.
Evidence	List of sensitive data elements transmitted between processors.
Threat Scenarios	TS20, TS21, TS24, TS29, TS34

ID	24MM.SEC.PLAT.PER.3
Requirement	Bus level communication containing sensitive information shall implement an integrity protection mechanism on message contents. This requirement applies only to communication between the SoC and TA100.
Reasoning	Integrity protection mechanisms prevent attackers from manipulating sensitive information and/or controlling the behavior of the ECU.
Evidence	List of sensitive data elements transmitted between processors.
Threat Scenarios	TS18, TS22, TS25, TS26, TS30, TS31

ID	24MM.SEC.PLAT.PER.4
Requirement	Peripheral chips that store or process sensitive information shall be cryptographically “bound” to the main application SoC. An attacker cannot remove the peripheral chip and use it outside of the SoC. Note: “Binding” refers to sharing some type of secret or token between the peripheral chip and the main SoC. Without the secret or token, communication between the two components is impossible. This prevents an attacker from removing the peripheral chip and using it with an attacker-controlled device to perform signing operations, extract sensitive information, or other security functions. Note: This requirement applies to security chips such as TPMs, eUICCs, or HSMs.
Reasoning	If an attacker can use the peripheral chip outside of the ECU, they may be able to use the chip to authenticate with backend services using laptops or other devices.
Evidence	Binding design review materials.
Threat Scenarios	TS20, TS21, TS24, TS29, TS34

ID	24MM.SEC.PLAT.PER.7
Requirement	The ability of peripherals to become a DMA bus master shall be disabled wherever possible
Reasoning	Prevents attackers from installing peripheral devices that could abuse DMA access to read entire contents of RAM.
Evidence	List of each peripheral that has DMA bus master capabilities.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.PER.8
Requirement	Peripherals with bus master capabilities shall have their memory access restricted to the minimum necessary. Note: This requirement could be satisfied by using an IOMMU.
Reasoning	Prevent PCIe or other peripherals from accessing private memory not intended for that peripheral’s use.
Evidence	List of the IOMMU mapping and allocations
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

4.15 Automated Testing

ID	24MM.SEC.PLAT.AT.1
Requirement	Interfaces to support automated testing shall be provided as a diagnostic facility available on debug software images. Note: In particular, 24MM.SEC.PLAT.DBG.PROD.3 applies to interfaces which support automated testing.
Reasoning	The ability to perform automated testing of the ECU integrated with the entire vehicle system is critical to ensuring that system satisfies security requirements and meets customer expectations.
Evidence	Automated testing design review materials.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.PLAT.AT.3
Requirement	Activation of interfaces to support automated testing shall not degrade the performance of the ECU beyond the tolerances otherwise considered acceptable for customer experience.
Reasoning	The behavior and performance of the ECU during automated testing should be equivalent to and representative of what a user would expect under normal operating conditions. Deviation would reduce the applicability of test results.
Evidence	N/A
Threat Scenarios	TS7, TS15, TS16

5 Application Requirements

5.1 Cryptographic Algorithms

ID	24MM.SEC.APP.CRYP.1
Requirement	All software shall only use the encryption and signing algorithms specified in section 4.1 Cryptographic Algorithms. Software shall not use custom cryptographic algorithms.
Reasoning	Use of obsolete or custom cryptography does not provide sufficient protection from cryptanalysis, so well understood and robust algorithms must be used to prevent an attacker from compromising the data protected by the cryptographic algorithms.
Evidence	List of all cryptographic algorithms that are used in the system (including open-source software and third-party software).
Threat Scenarios	TS20

5.2 Key Management

ID	24MM.SEC.APP.KEY.1
Requirement	All software shall only use cryptographic keys issued by Toyota and handled securely according to section 4.2 Key Management.
Reasoning	Key material is the basis of trust for the device, so Toyota must manage the full key lifecycle to ensure keys are not leaked
Evidence	List of all cryptographic keys that are used in the system.
Threat Scenarios	TS20, TS21

5.3 Secure Boot

ID	24MM.SEC.APP.SB.1
Requirement	All software shall be authenticated before execution according to section 4.3 Secure Boot. Applications shall not dynamically load and execute plugins or scripts that have not first been verified as part of secure boot.
Reasoning	An unauthenticated mechanism for loading and executing software could allow an attacker to execute malicious code.
Evidence	Secure boot design review materials.
Threat Scenarios	TS1-5, TS9-13

5.4 Secure Updates

There are two broad approaches for updating data on the ECU:

- System Software Update: Update mechanism that can apply to the whole ECU as described in section 4.4 Secure Updates.
- Application Specific Updates: Custom update mechanism for updating configuration data or databases that is unique to each application. These updates could be distributed by downloading from a webserver, a USB drive, a mobile phone, or any other means. Examples:
  - Navigation map data updates.
  - Updates to machine-learning models.

The following requirements place limitations on these update mechanisms.

ID	24MM.SEC.APP.UPD.1
Requirement	All software shall only be updateable and modifiable through the system update process described in section 4.4 Secure Updates. There shall be no application specific update mechanism or side-channel for updating or modifying software.
Reasoning	Updates to software must be directly controlled and issued by Toyota to ensure proper functional and security testing is performed.
Evidence	Software update design review process.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.UPD.2
Requirement	Application specific updates shall only update configuration data or databases that require rapid and frequent updates. No other data or software shall be updated via an application specific update.
Reasoning	Updates outside of the normal processes have a higher risk of being compromised or susceptible to a fault. Limit to only data that requires frequent updates to lower risk.
Evidence	Documentation of: 1) What applications require special update mechanism 2) Process of updating application specific data 3) What application data / configurations are included in this special update?
Threat Scenarios	TS18, TS30, TS31

ID	24MM.SEC.APP.UPD.3
Requirement	Any security-related configuration data shall only be modifiable through the secure update process as described in section 4.4 and shall be protected by the secure boot process as described in section 4.3. Example: Security-related configuration data includes:

	• Mandatory Access Control (MAC) policies • Sandboxing configuration values • Log settings • Certificates Note: All static configuration data shall also be protected by secure boot as described in section 4.3. This requirement makes it explicit that all security-related configuration data must be static.
Reasoning	An attacker may be able to modify any data not protected by secure boot. Must prevent an attacker from modifying anything that would impact the security of the ECU.
Evidence	Documentation of how security configurations are handled by each application.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.APP.UPD.4
Requirement	The ECU must verify the data obtained from application specific updates has been issued by Toyota. This could be through a: • Digital signature • HMAC • CMAC • Other mechanism approved by Toyota.
Reasoning	A third party should not be directly trusted with placing data on the ECU. Corrupt or malformed data could lead to ECU compromise, and an attacker may be able to break a third party's systems to distribute malicious data.
Evidence	Documentation for each application specific update method.
Threat Scenarios	TS18, TS30, TS31

ID	24MM.SEC.APP.UPD.5
Requirement	The design of all application specific updates must be approved by Toyota. The supplier shall document the method and review with Toyota.
Reasoning	Updates outside of the normal processes have a higher risk of being compromised or susceptible to a fault. Toyota must understand and approve the design to ensure a secure implementation.
Evidence	Documentation of: 1) What applications require special update mechanism 2) What application data / configurations are included in this special update? 3) How these update packages are verified and applied to the target.
Threat Scenarios	TS18, TS30, TS31

5.5 Secure Debug

ID	24MM.SEC.APP.DBG.1
Requirement	During development all debug and diagnostic hardware and software interfaces shall require authentication before use. See section 4.5 Secure Debug.
Reasoning	Prevent an attacker that obtains access to a debug unit from easily extracting useful information from the device.
Evidence	List of each debug interface and authentication method enabled during development.
Threat Scenarios	TS1, TS3, TS6, TS9, TS11, TS14, TS18-26, TS30, TS33-36, TS38, TS39, TS44, TS46, TS47

ID	24MM.SEC.APP.DBG.2
Requirement	All hardware and software debug and diagnostic functions and interfaces shall be removed for production devices apart from the facilities listed in 24MM.SEC.PLAT.DBG.PROD.1. If an interface cannot be removed, it must be permanently disabled. See section 4.5 Secure Debug.
Reasoning	Prevent an attacker from abusing diagnostic and debug facilities to compromise an ECU or gather information useful for reverse-engineering.
Evidence	List of each debug interface and authentication method enabled during development.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

5.6 HLOS Secure Storage

ID	24MM.SEC.APP.STG.1
Requirement	All persistent sensitive information shall be stored in the single, systemwide HLOS secure storage solution as described in section 4.8 HLOS Secure Storage. Note: This is in addition to the Full Disk Encryption requirements of 4.9. From an external perspective, data stored in HLOS secure storage is double encrypted.
Reasoning	Sensitive information must be protected from disclosure and tampering both externally at rest, and internally by unauthorized, compromised applications.
Evidence	N/A
Threat Scenarios	TS20, TS21, TS24, TS29

5.7 Logging

ID	24MM.SEC.APP.LOG.1

Requirement	Every software component shall perform logging using a subsystem intended specifically for logging. Software components shall not directly send logs to the standard output or standard error streams, or to non-standard / custom output facilities outside of the logging subsystem.
	
	Note: Standard output and standard error may be still redirected into the logging system.
	
	Note: The following software patterns shall be converted to use an appropriate logging subsystem or removed from the component's source.
	• SHELL:
	  ○ echo (file descriptors 1, 2)
	  ○ cat (file descriptors 1, 2)
	• C language:
	  ○ printf(…);
	  ○ fprintf(stdout,…);
	  ○ fprintf(stderr,…);
	• C++ language:
	  ○ std::cout << …;
	  ○ std::cerr << …;
	• Python:
	  ○ print …
	  ○ print >> sys.stderr, …
	• Java:
	  ○ System.out.print(…);
	  ○ System.out.println(…);
	  ○ System.err.print(…);
	  ○ System.err.println(…);

Reasoning	Standard output and standard error do not provide the necessary log formatting. They also could introduce buffering delays and could drop some log messages in the event of a shutdown or power loss. Finally, they do not allow easy configuration of different logging destinations.
Evidence	Code review results. Code review must verify this requirement as part of review.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.2

Requirement	A log record shall be generated for cybersecurity relevant events, including whenever a cybersecurity mechanism blocks or denies an action. This includes:
	• Authorization errors. Examples:
	  ○ A process attempts to access memory it is not authorized for.
	  ○ A process attempts to access a file it is not authorized for.
	  ○ Unauthorized policy changes to Mandatory Access Control.
	  ○ Violations of Mandatory Access Control policy.
	• Authentication errors. Examples:
	  ○ A Linux user login fails.
	  ○ Wi-Fi authentication failure.
	  ○ PKI validation failure.

	• Hardware Access Control error. Example:
	  ○ SoC throws an exception when normal world process accesses peripheral reserved for TrustZone.
	• Unexpected data:
	  ○ High CPU, memory, IO, or network usage.
	  ○ Low free storage.
	  ○ Malformed communication packets or frames.
	  ○ Unexpected return values or function parameters.
	• Failed updates.
	• Unexpected system reboot.
	• System crashes.
	• Time delta of more than 24 hours between RTC and gPTP.
	
	Note: The Linux audit framework should be used where applicable to collect security-relevant events.
	
	This requirement is recommended.

Reasoning	Detecting cybersecurity events is critical for real-time and forensic analysis.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.3
Requirement	A log record shall be generated for normal events that are especially security relevant. This includes: • Successful updates. • Kernel modules loaded / unloaded. • User logins. • Debug tool accesses. • Startup / shutdown Note: The Linux audit framework should be used where applicable to collect security-relevant events.
Reasoning	Tracking important security related actions is critical for real-time and forensic analysis.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.4
Requirement	The ECU shall log the versions of every software component at every ECU start. Note: If multiple software components are grouped into update “packages” or “units” that are always updated synchronously, then just the “package” or “unit” version number can be logged.
Reasoning	Having the version information of every software component is critical for forensic analysis.
Evidence	Documentation and tools necessary to extract and review log records.

Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.5
Requirement	The software version information logged by the ECU should map to a specific software release that can be in source code control.
Reasoning	It is essential to easily find the exact version of the software source code that generates every log message.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.6
Requirement	Logs shall not contain data of unknown meaning. Note: Logs may contain invalid parameter values if the logging of such data does not violate any other requirement in this document.
Reasoning	Logging is to be used to instrument applications appropriately. It must not be used to log, for example, raw protocol data. Use other means, such as protocol analyzers with debug certificates, to view and decode protocol data in lab, development, testing, certification, and other non-production environments.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS20, TS21, TS24, TS29, TS34

ID	24MM.SEC.APP.LOG.7
Requirement	Sensitive information shall not be logged in cleartext at any log level. If sensitive information is logged in an encrypted manner, there must be a Toyota controlled process to decrypt with appropriate approvals. This includes: • Personal Information as defined in the P21MM Privacy Specification. • Raw protocol streams such as CAN frames or HTTPS payloads. • Firmware dumps. • Financial information • Cryptographic keys or state information • Bluetooth link keys • Wi-Fi passphrase • PIN codes • Passwords
Reasoning	Log files cannot be allowed to inadvertently “leak” sensitive information to VSOC operators, technicians, or other consumers of log information.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS20, TS21, TS24, TS29, TS34

ID	24MM.SEC.APP.LOG.8
Requirement	Stack traces, memory dumps, and other raw debug and tracing information shall not be logged at any log level. Note: This requirement does not prevent implementation of any Intrusion Detection System (IDS) requirements or other similar requirements defined in other documents. All IDS requirements shall still be implemented.
Reasoning	Detailed information on the structure and behavior of software is especially useful to an attacker that is trying to exploit potential vulnerabilities.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.9
Requirement	Logs shall contain a component name or label that allows logs to be filtered according to the source software component.
Reasoning	It is often important to focus log collection on specific software components that exhibit anomalous behavior. Having specific labels for each component enables this.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.10
Requirement	Component names or labels used in logs should directly map to names or labels used in software source code.
Reasoning	This ensures that source code version control labels may be matched against running code, reducing the chances of source code to binary mismatches.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.11
Requirement	Logs shall contain the absolute time the log was generated or a relative time from a known absolute time. The time shall include the year, month, day, hour, minute, and second. The time precision shall be a minimum of one second. The time precision should be one millisecond if possible.
Reasoning	It is critical during forensic analysis to know the exact time every log event took place.
Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.LOG.12
Requirement	Logs shall contain the severity level of the information contained in the log message.
Reasoning	It is critical during real-time analysis and forensic analysis of cybersecurity events to be able to filter out high severity events.

Evidence	Documentation and tools necessary to extract and review log records.
Threat Scenarios	TS1-5, TS9-13

5.8 Certificate Management

ID	24MM.SEC.APP.CRT.1
Requirement	The CA store maintained by the certificate manager shall be the only CA store on the ECU (see section 4.12 Certificate Management). Software shall not use application specific CA stores.
Reasoning	Eases maintenance and prevents a gap in security between different CA stores.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.CRT.2
Requirement	All software shall perform certificate revocation checking according to section 4.13.5 Certificate Revocation Checks using the common certificate revocation checking service provided by the certificate manager (see section 4.12 Certificate Management).
Reasoning	A common implementation of certificate revocation checking will ensure a robust approach that does not suffer from unique implementation errors or gaps. It also allows new functionality to be added easily to upgrade the overall security of the system.
Evidence	N/A
Threat Scenarios	TS59-64

5.9 Communications

5.9.1 External Communications

External communications encompass communication performed with remote servers, such as Toyota servers. This communication can be performed over many interfaces including the 24DCM cellular interface or the 24MM Wi-Fi interface.

Attackers will attempt to monitor and tamper with network communication over these interfaces to exfiltrate sensitive information.

The authentication mechanism for Toyota and 3rd party servers is mutual-TLS version 1.3. The authorization mechanism for all servers is OAuth 2.0.

ID	24MM.SEC.APP.COM.EXT.1
Requirement	All communications with out-vehicle entities shall use TLS 1.3 or higher. The only exceptions are: 1. DNS for traffic to non-Toyota / non-3rd party servers.

Reasoning	TLS prevents eavesdropping and MitM attacks by encrypting the data and authenticating the endpoint.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.APP.COM.EXT.3
Requirement	The ECU shall authenticate with all Toyota or 3rd party backend services using mutual-TLS.
Reasoning	Mutual TLS provides strong authentication of both the server and the client.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.COM.EXT.4
Requirement	The ECU shall use OAuth 2.0 to perform authorization with all Toyota or 3rd party backend services.
Reasoning	OAuth 2.0 provides a robust, well-proven mechanism to perform authorization.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.COM.EXT.5
Requirement	Implicit OAuth2.0 flows shall not be supported.
Reasoning	Implicit flows carry greater risk of disclosure of the token.
Evidence	Documentation of all OAuth authorization flows.
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.COM.EXT.6
Requirement	The following fields of OAuth 2.0 tokens shall be validated by the ECU before use: • Signature • Validity Window • Audience • Claims (if present)
Reasoning	Ensure token is valid and has not been supplied by a malicious endpoint.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.COM.EXT.7
Requirement	OAuth 2.0 tokens shall be encrypted if they contain sensitive information.
Reasoning	Ensure system internal information is not leaked through the tokens.
Evidence	N/A
Threat Scenarios	TS20, TS21, TS24, TS29, TS34

ID	24MM.SEC.APP.COM.EXT.8
Requirement	All authorization tokens must be communicated through a secure channel such as TLS.
Reasoning	Keep request information and token values secret from an eavesdropping attacker to prevent an attacker gaining access to a backend system.
Evidence	N/A
Threat Scenarios	TS20

ID	24MM.SEC.APP.COM.EXT.9
Requirement	Whenever possible, the backend service shall use certificate bound access tokens (RFC 8705). The tokens shall be OAuth 2.0 tokens bound to the mutual-TLS client certificate.
Reasoning	Prevents tokens from being removed and used on another device.
Evidence	N/A
Threat Scenarios	TS59-64

ID	24MM.SEC.APP.COM.EXT.10
Requirement	All persistent authorization tokens shall be stored in HLOS Secure Storage (see 4.8 HLOS Secure Storage).
Reasoning	Ensure an attacker cannot extract authorization tokens to impersonate a user.
Evidence	N/A
Threat Scenarios	TS20

ID	24MM.SEC.APP.COM.EXT.11
Requirement	Sensitive information sent between the ECU and backend services shall be end-to-end encrypted on top of the TLS transport encryption.
Reasoning	TLS provides encryption for transport over the Internet, but sensitive information must be protected during all stages of communication (for instance after a proxy or gateway) until it reaches the backend service.
Evidence	List of services using end-to-end encryption on the ECU.
Threat Scenarios	TS20

ID	24MM.SEC.APP.COM.EXT.12
Requirement	The ECU shall not run any IP-based services that accept connections from an external interface including Wi-Fi and cellular.
Reasoning	Accessible network services are a major attack surface with an elevated risk of compromise by an attacker. Removing those surfaces reduces the risk of compromise.

Evidence	N/A
Threat Scenarios	TS59-64

5.9.2 Wi-Fi

ID	24MM.SEC.APP.COM.WIFI.1
Requirement	Any function that initiates a Wi-Fi connection (wireless Carplay, OTA from a mobile device, etc.) shall use a WPA2 or WPA3 connection. The connection shall not be unencrypted or unauthenticated.
Reasoning	Prevent an automated service from creating an insecure connection to the head unit.
Evidence	Documentation on how functions initiate Wi-Fi connections and derive the passphrase.
Threat Scenarios	TS59-64, TS68

5.9.3 Bluetooth

ID	24MM.SEC.APP.COM.BLT.1
Requirement	Bluetooth services shall be developed in accordance with NIST SP 800-121 revision 2. Supplier shall document how they followed recommendations from NIST SP 800-121 revision 2 and Toyota shall review and approve. Note: This document provides recommendations on Bluetooth Security Modes to help protect against attackers. NIST, "SP 800-121 Rev.2, Guide to Bluetooth Security," [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-121/rev-2/final.
Reasoning	Ensuring that the Bluetooth interface developed is compliant with NIST SP 800-121 revision 2 prevents the attacker from abusing Bluetooth services to extract sensitive information or perform unintended actions.
Evidence	Documentation on how Bluetooth is configured to be compliant with this requirement.
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.APP.COM.BLT.2
Requirement	All Bluetooth services shall require Security Mode 4 Level 4 for Bluetooth BR, EDR and High Speed (HS) connections. Note: Some use cases may need to support different security levels. Toyota must grant approval for any use cases that use a different security level.

Reasoning	Ensuring that the Bluetooth BR, EDR and HS connections use the security mode 4 and service level 4 mitigate eavesdropping, unauthorized access, and Man-in-the-Middle attacks.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.APP.COM.BLT.3
Requirement	All Bluetooth services shall require Security Mode 1 Level 4 for Bluetooth LE connections. Note: Some use cases may need to support different security levels. Toyota must grant approval for any use cases that use a different security level.
Reasoning	Ensuring that the Bluetooth LE uses security mode 1 and service level 4 mitigate the Man-in-the-Middle attacks.
Evidence	N/A
Threat Scenarios	TS59-64, TS68

ID	24MM.SEC.APP.COM.BLT.4
Requirement	Bluetooth services with sensitive communication shall implement application-level authentication and encryption on top of Bluetooth's existing mechanisms. See Security Recommendation #25 Table 4-2. Bluetooth Piconet Security Checklist from NIST SP 800-121 revision 2 https://csrc.nist.gov/publications/detail/sp/800-121/rev-2/final. Note: Bluetooth services with sensitive communication include communications between Toyota mobile applications and the MM device such as: • User Profile • Remote Parking
Reasoning	An access control method, such as challenge/response, on characteristics helps protect against attackers with access over Bluetooth from accessing all characteristics.
Evidence	List of Bluetooth services implementing application-level encryption.
Threat Scenarios	TS59-64, TS68

5.9.4 In-Vehicle Communications

ID	24MM.SEC.APP.COM.INT.1
Requirement	All services accessible by other ECUs or outside devices shall require authentication before they can be used. Exceptions for diagnostics or other functions shall be documented and approved by Toyota. Example: Services could include UDS diagnostics, update agents, and control functions. Note: This includes services on any bus type, including CAN, Ethernet, and USB.

Reasoning	Prevent an attacker with access to an in-vehicle bus from abusing services on the ECU.
Evidence	List of available services and the corresponding authentication mechanism.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.COM.INT.2
Requirement	Any software loaded and executed from USB mass storage shall have a certificate whose signature is verified before execution of the software. The certificate must be verified as being issued by Toyota. Note: This clarifies that 24MM.SEC.PLAT.SB.1 applies to software loaded via USB. Note: Software includes shell scripts.
Reasoning	Verifying signatures for all software executed on the head unit prevents an attacker from loading malicious contents onto the ECU using USB mass storage devices.
Evidence	Documentation describing implementation of signature verification.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.COM.INT.3
Requirement	Any data loaded from USB mass storage and stored on the ECU shall have a certificate whose signature is verified. The certificate must be verified as being issued by Toyota. If signature verification fails, the data is deleted from the ECU. Note: This includes updates to data such as navigation map data and media databases. Note: This is clarifying that 24MM.SEC.UPD.1 applies to software and data loaded via USB.
Reasoning	If some data can be updated without authentication, an attacker could install maliciously formatted data that takes advantage of vulnerabilities in the software that parses the data.
Evidence	Documentation describing implementation of signature verification.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

5.10 Software Hardening

ID	24MM.SEC.APP.HRD.1
Requirement	ARM Execute Never (XN) bit shall be enabled and configured by software so that only memory regions containing code can be executed. Note: Execute Never (XN) bit tags regions as ‘do not execute’ or ‘non- executable code’. Any attempt to execute code for this region shall result in a segmentation fault that halts the execution.
Reasoning	Enabling XN bit can prevent an attacker from inserting and executing malicious code in data regions of memory, making it harder to exploit buffer overflow vulnerabilities.

Evidence	Unencrypted filesystem image.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.2
Requirement	Wherever possible, software shall be compiled as Position Independent Executables (PIE) to allow ASLR. This includes main applications and libraries. Note: PIE allows ASLR for the actual binary's code. This can be enabled using the '-pie' linker flag.
Reasoning	PIE is necessary to support ASLR. Enabling PIE allows the software to be compiled as position independent and makes it difficult to predict addresses of components needed for exploitation. This makes it more difficult to exploit the software.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.3
Requirement	Wherever possible, software shall be compiled with the full Relocation Read-Only (RELRO) option. This includes main applications and libraries. Note: This can be enabled using '-Wl,-z,relro' linker flags
Reasoning	Enabling the full RELRO makes the entire Global Offset Table (GOT) read-only. Thus, it mitigates the risk for some GOT overwrite attacks.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.4
Requirement	Wherever possible, software shall be compiled with the Immediate Binding option. This includes main applications and libraries. Note: Enabling immediate binding can increase startup time. If this cannot be enabled to meet KPI then an exception must be approved by Toyota. Note: This can be enabled using '-Wl,-z,now' linker flags.
Reasoning	Enabling the immediate binding flag makes the linker resolve all relocations before start of the program. Thus, it mitigates the risk of memory corruption attacks. The flag combined with RELRO flag can mitigate more GOT overwrite attacks.
Evidence	Unencrypted filesystem image.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.5
Requirement	Wherever possible, software shall be compiled with compiler options that detect and prevent software exploitation, otherwise known as Control-Flow Integrity (CFI) options.

	Note: The following are some supported flags: • Clang -fsanitize=cfi • GCC -fvtable-verify: C++ virtual table verification. • Both -fstack-protector-all: Stack protection. -D_FORTIFY_SOURCE=2: Buffer overflow checks. Note: If any options cannot be enabled due to KPI requirements then an exception must be approved by Toyota.
Reasoning	Enabling CFI flags can detect the buffer overflows and terminates the program during runtime. Thus, it will mitigate the risk of malicious code insertion.
Evidence	List of compiler flags used for building supplier created software.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.6
Requirement	Whevere possible, software compiled with GCC, including main executables and libraries, shall be compiled by enabling the protection for stack clash. Note: This can be enabled using '-fstack-clash-protection' flags Note: If this option cannot be enabled due to KPI requirements then an exception must be approved by Toyota.
Reasoning	Enabling stack clash protection flag can detect the unconstrained growth of stack or heap memory usage. Thus, it will mitigate vulnerabilities like denial-of-service.
Evidence	List of compiler flags used for building supplier created software.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.HRD.7
Requirement	Wherever possible, software complied with Clang, including main executables and libraries, shall be compiled by enabling the ShadowCallStack. Note: This can be enabled passing the "-fsanitize=shadow-call-stack" flag to both compiler and linker command lines.
Reasoning	Enabling ShadowCallStack flag can protect programs against overwriting return address. Thus, it will mitigate the risk for arbitrary code execution.
Evidence	List of compiler flags used for building supplier created software.
Threat Scenarios	TS1-5, TS9-13

5.11 Software Sandboxing

ID	24MM.SEC.APP.SBX.1
Requirement	The supplier shall develop a custom Mandatory access control policy that restricts each process's access to controlled resources to the minimum necessary to perform that process's function. Controlled resources include: • Files • Directories • Inter-process communication channels • Other processes • Peripherals Note: The supplier may utilize MAC policies provided by the distribution or platform. Note: The supplier should consider a Role Based Access Control (RBAC) approach to designing the MAC policy. This requirement is recommended.
Reasoning	An intelligently designed MAC policy can be a powerful defense that prevents an attacker from broadening access to the system after an initial attack.
Verification	Mandatory Access Control Policy.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.SBX.2
Requirement	Processes shall only be granted the necessary privileges using Linux capabilities.
Reasoning	Limiting access to only the required facilities helps prevent an attacker from taking advantage of a compromise to perform unintended actions such as mount filesystems, load kernel modules, spoof packets by denying access to raw sockets, altering attributes in the filesystem. This requirement is recommended.
Evidence	List of capabilities assigned to each process and application.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.SBX.3
Requirement	Wherever possible, each process shall run with a unique user ID. Note: Child "worker" processes of a main application process may use the same user IDs. Note: Access control policies shall be written to accommodate this requirement, instead of access control policies dictating where this requirement can be implemented.
Reasoning	Provides finer-grained access control by allowing specific rules to be given to each process.
Evidence	List of processes and whether the user ID is unique or common. Rationale for all common user IDs.

Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.SBX.4
Requirement	Linux seccomp filtering shall be implemented for every possible process to restrict access to only the necessary system calls. Note: If seccomp filtering causes performance targets to not be met for a certain application, then the supplier may request an exception that documents the performance target that is failing and the functionality of the application. All exceptions must be approved by Toyota.
Reasoning	Restrict attack surface of Linux kernel by removing access to unneeded system calls.
Evidence	Seccomp configuration for each process.
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.SBX.5
Requirement	Linux namespaces shall be used to restrict access to only the necessary resources. The following namespaces shall be used: • Network namespace for limiting access to network interfaces, etc. The following namespaces should be used: • IPC namespace to prevent unintended IPC communication. • Mount namespace to prevent access to unnecessary files. • PID namespace to prevent manipulation of other processes. • cgroup namespace to prevent modification of resource limits. This requirement is recommended.
Reasoning	Prevent a compromised process from manipulating other processes or accessing unauthorized information.
Evidence	Namespace configuration for each process..
Threat Scenarios	TS1-5, TS9-13

ID	24MM.SEC.APP.SBX.6
Requirement	Linux cgroups shall be used to prevent an attacker from consuming excess resources such as memory, CPU time, and PIDs. This requirement is recommended.
Reasoning	Prevent a compromised process from maliciously lowering the performance or crashing the rest of the system by consuming resources.
Evidence	Limits configured for each process in cgroups.
Threat Scenarios	TS1-5, TS9-13

6 SA6155/SA8155 Requirements

This section documents specific requirements and decisions which are to be made to securely boot the Qualcomm SA8155 and SA6155 SOCs.

6.1 Secure Boot

ID	24MM.SEC.QC.SB.1
Requirement	Multiple Root Certificate (MRC 2.0) shall be supported by writing the SHA-384 (if supported, SHA-256 if not) hash of the concatenation of 4 root certificate hashes to the OEM_PK_HASH fuses.
Reasoning	This enables the Multiple Root Certificate feature of the SAx155 which allows Toyota to revoke a leaked root certificate and replace it.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.2
Requirement	After verification of a successful update, the combination of ROOT_CERT_ACTIVATION_LIST and ROOT_CERT_REVOCATION LIST shall always specify that exactly one root hash is activated and not revoked at a time. In other words, after a successful update to a new root cert, all previously activated certificates must be marked as revoked.
Reasoning	This reduces the attack surface by limiting the number of effective certificate hashes.
Evidence	Documentation of procedure for revoking root certificate.
Threat Scenarios	TS21

ID	24MM.SEC.QC.SB.3
Requirement	ROOT_CERT_REVOCATION_LIST shall be blown for any root certificate that has reached end of life (leaking, rotation, etc.).
Reasoning	This reduces the attack surface by limiting the number of effective certificate hashes.
Evidence	Documentation of procedure for revoking root certificate.
Threat Scenarios	TS21

ID	24MM.SEC.QC.SB.4
Requirement	AUTH_EN and PK_HASH_IN_FUSE shall be enabled for SEC_BOOT1-3.
Reasoning	This enables image authentication for secure boot.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.5
Requirement	OEM_HW_ID and OEM_PRODUCT_ID fuses shall be set and used during secure boot and updates.

Reasoning	This keeps images from another hardware or product from being loaded.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.6
Requirement	OEM_PRODUCT_ID shall be different for debug and production ECUs.
Reasoning	Listing of configured eFuse values.
Evidence	Verify OEM_PRODUCT_ID values are unique between debug and production unit.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.7
Requirement	OEM_COUNTER_MEASURE_ENABLE fuse shall be blown.
Reasoning	This enables a feature which is a mitigation for CLKSCREW attacks.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.8
Requirement	Image signing shall be done using RSA-4096 (exponent 65537) with a SHA-384 hash.
Reasoning	Use most secure algorithms supported to provide longest resistant to attack.
Evidence	Collection of boot images.
Threat Scenarios	TS20

ID	24MM.SEC.QC.SB.9
Requirement	Anti-Rollback (ARB) shall be enabled by setting BOOT_ANTI_ROLLBACK_EN, TZAPPS_ANTI_ROLLBACK_EN, and PIL_SUBSYS_ANTI_ROLLBACK_EN.
Reasoning	Anti-Rollback feature can be used to permanently close security holes.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.10
Requirement	ARB (Anti Rollback) version shall be incremented when an update is meant to mitigate a security vulnerability.
Reasoning	By increasing the ARB version for a section, the bootloader should blow a fuse which disallows an older version from being loaded.
Evidence	Documentation of procedure for issuing a security update with rollback protections.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

ID	24MM.SEC.QC.SB.11
Requirement	FORCE_USB_BOOT_DISABLE may be left at 0 to enable FORCE_USB_BOOT pin to redirect boot flow to USB booting. Note: USB boot shall still follow all secure procedures of section 4.3 Secure Boot.
Reasoning	USB may be used to flash returned devices with different firmware.
Evidence	Listing of configured eFuse values.
Threat Scenarios	TS1-5, TS9-13, TS18, TS30, TS31

6.2 Trusted Execution Environment

ID	24MM.SEC.QC.TEE.1
Requirement	The QTEE configuration variable “cmdlib_gppo_rpmb_enablement” shall be set to 1 (enabled).
Reasoning	Ensures Trusted Applications (TAs) cannot be rolled back to older versions.
Evidence	QTEE configuration.
Threat Scenarios	TS1-5

ID	24MM.SEC.QC.TEE.2
Requirement	The QTEE configuration variable “OEM_keystore_enable_rpmb” shall be set to 1 (enabled).
Reasoning	Ensures keystore is protected from rollback attacks.
Evidence	QTEE configuration.
Threat Scenarios	TS18

ID	24MM.SEC.QC.TEE.3
Requirement	The QTEE configuration variable “OEM_allow_rpmb_key_provision” shall be set to 0 (disabled).
Reasoning	Ensures RPMB key cannot be maliciously reprovisioned.
Evidence	QTEE configuration.
Threat Scenarios	TS1-5, TS18

ID	24MM.SEC.QC.TEE.4
Requirement	The QTEE configuration variable “OEM_disable_rpmb_autoprovisioning” shall be set to 1 (disabled).
Reasoning	Ensures RPMB key cannot be maliciously reprovisioned.
Evidence	QTEE configuration.

Threat Scenarios	TS1-5, TS18

ID	24MM.SEC.QC.TEE.5
Requirement	The QTEE configuration variable “OEM_app_sandbox_default” shall be set to 1 (enabled).
Reasoning	Ensures 3rd party trusted applications operate with lesser privileges in TrustZone.
Evidence	QTEE configuration.
Threat Scenarios	TS1-5

6.3 Full Disk Encryption

ID	24MM.SEC.QC.FDE.1
Requirement	External flash storage of Qualcomm SoC shall be compatible with Qualcomm Inline Crypto Engine (ICE). Example: SA6150/SA8150 must use UFS flash.
Reasoning	ICE provides efficient method of implementing full disk encryption, but this may depend on specific types of flash storage.
Evidence	HBOM and datasheet of flash chip.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.QC.FDE.2
Requirement	Qualcomm ICE shall be configured to use AES XTS mode for full disk encryption in accordance with NIST SP 800-38E.
Reasoning	XTS is designed specifically for full disk encryption use cases and provides more protection against manipulation of ciphertext than other supported modes.
Evidence	Documentation of ICE configuration.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.QC.FDE.3
Requirement	Qualcomm Unified Image Encryption (UIE) shall be implemented for the following boot images: • All Trusted Applications Note: If UIE performance is enhanced to minimize decryption latency, then all boot images shall support UIE.
Reasoning	Encrypting boot images makes it more difficult to perform reverse engineering to identify and exploit flaws in the early boot code that could be exploited by an attacker.
Evidence	Collection of boot images.

Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.QC.FDE.4
Requirement	Qualcomm Full Disk Encryption (FDE) shall be implemented to encrypt the entire Linux filesystem saved on external flash. Note: This includes all partitions, not just the "/data" partition. Extra support will need to be added on top of the Qualcomm provided implementation that only encrypts "/data".
Reasoning	Full disk encryption increases the difficulty of reverse-engineering and provides a layer of protection to sensitive information mistakenly stored outside of secure storage.
Evidence	Raw flash dump.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

ID	24MM.SEC.QC.FDE.5
Requirement	The Linux kernel, initramfs, and DTB images shall be encrypted in flash. The Qualcomm APPSBL shall be extended to support decrypting those images. This requirement is recommended.
Reasoning	Qualcomm UIE and FDE functions do not support encrypting Linux kernel, initramfs, and DTB. Encrypting those images increases the difficulty of reverse-engineering and makes exploiting vulnerabilities more difficult.
Verification	Raw flash dump.
Threat Scenarios	TS8, TS17, TS20, TS21, TS24, TS29, TS34, TS39

6.4 SoC Memory Access Configuration

Requirements 24MM.SEC.QC.SOC.1-2 *removed since XPU and IOMMU settings cannot be directly controlled by OEM.*

6.5 Peripherals

Requirements removed.

6.6 Software Sandboxing

Removed 24MM.SEC.QC.SBX.1 as it does not apply to this processor.

7 Appendix

7.1 Use Cases

These use cases give more background and rational for some of the cybersecurity features discussed above. Each use case is described using the following structure:

Threat Scenario: <Description of the cybersecurity problem this feature solves>

Countermeasure: <High level description of the security feature>

Example Use Case: <Walk through of how the security feature prevents an example attack>

7.1.1 Secure Boot

Threat Scenario: Attackers will modify software stored in external flash to perform malicious actions to the ECU. These malicious actions could include dumping sensitive information, installing persistent malware, and adding debug or development facilities to perform reverse engineering.

Countermeasure: Design the ECU to execute only authenticated software issued by Toyota. This includes software at all levels of the software stack, starting after the mask ROM of the SoC and including all user space Linux applications loaded into RAM. The process by which every software component is validated before execution is called secure boot.

Example Use Case: An attacker finds an ECU in a junk yard. The attacker tries to extract sensitive information by replacing early boot images with malicious software. The malicious software would decrypt and dump the secure storage contents of the ECU. However, because the ECU implements secure boot, the ECU refuses to execute the malicious software, and the sensitive information is protected.

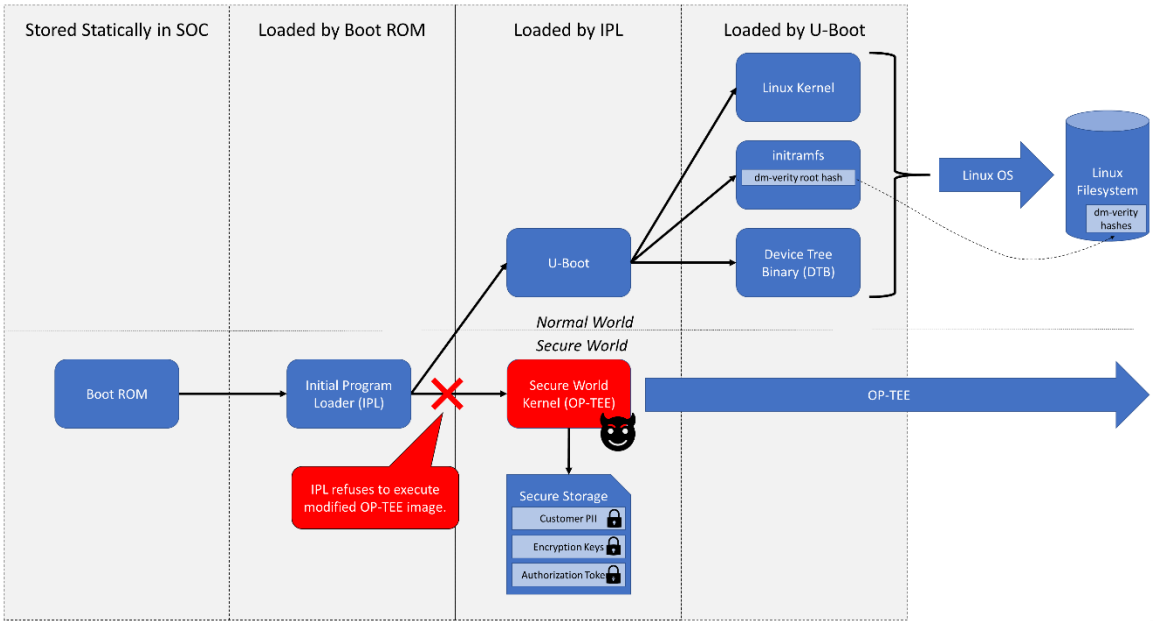

Figure 9 - Typical secure boot flow. Secure boot prevents execution of modified OP-TEE image.

7.1.2 Secure Updates

Threat Scenario:

1. Software on the ECU is susceptible to an unknown vulnerability that is discovered after the vehicle has entered operation. An attacker may discover this vulnerability and use it to attack the vehicle.
2. The update process could be abused by an attacker to perform a malicious update that installs malicious software on the ECU.

Countermeasure:

1. Implement a remote update process for every piece of software on the ECU.
2. Require update packages to be signed by Toyota and encrypted to preserve confidentiality.

Example Use Case: An attacker discovers a critical flaw in the Linux Bluetooth stack that affects the ECU. This flaw allows an unauthenticated attacker to achieve arbitrary code execution on the ECU and then send messages on in-vehicle buses.

As soon as Toyota becomes aware of the vulnerability, Toyota creates a patch and conducts validation and testing. Then Toyota signs and encrypts the update package and begins an OTA campaign to distribute the patch to all affected vehicles. This process can take place in a matter of days and be applied to all affected vehicles. In contrast, a SW update at a dealer could take months to reach even a small number of vehicles.

7.1.3 Secure Debug

Threat Scenario: Sometimes it is necessary to debug production devices using interfaces such as JTAG or by dumping a forensic log. While these functions can be useful to diagnose issues in production devices, they also provide a powerful capability for attackers to take control and reverse-engineer the ECU.

Countermeasure: Debug interfaces are only enabled after a debug user is successfully authenticated. The authentication mechanism must rely on a hardware-based authentication mechanism that will then activate the necessary debug ports.

Example Use Case: A customer returns their ECU to a dealer. The ECU is demonstrating unusual behavior that could indicate the result of a cyberattack. The dealer technician requests and is issued a certificate to extract the encrypted forensic logs using a debug port. However, the technician is not given the key to decrypt the logs. The encrypted forensic logs are then sent to Toyota for decryption and analysis.

An attacker is unable to access the forensic logs or enable the debug functionality of the ECU because they cannot create a valid debug certificate.

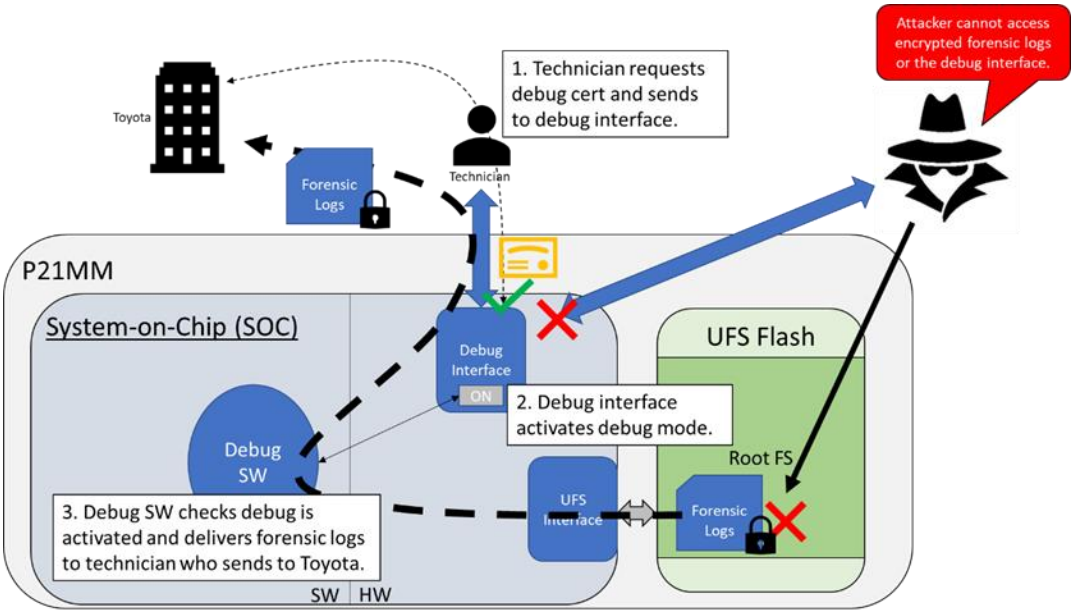

Figure 10 - Securely retrieving forensic logs via debug interface.

7.1.4 Trusted Execution Environment

Threat Scenario: The ECU will contain a variety of sensitive information including cryptographic keys, user PII, payment information and more. This data must be kept secure to protect the integrity of the device's operation and maintain user privacy. However, attackers have developed sophisticated techniques to reverse-engineer and extract information from devices.

Countermeasure: Trusted Execution Environments (TEE) are special hardware protected exclaves for executing trusted services that process and/or store sensitive information. These environments are anchored by a root of trust for verification during secure boot to ensure complete separation of execution and volatile and nonvolatile memory resources from the "normal world" applications. That is, Trusted Execution Environments are special security environments that execute streamlined, rigorously validated software to provide security services. These environments operate on sensitive information that is not accessible to normal applications.

Implementations of these environments include Arm TrustZone and special security coprocessors such as Secure Processing Elements (SPE). The following figure illustrates the separation of the "trusted execution environment" and the "normal world execution environment".

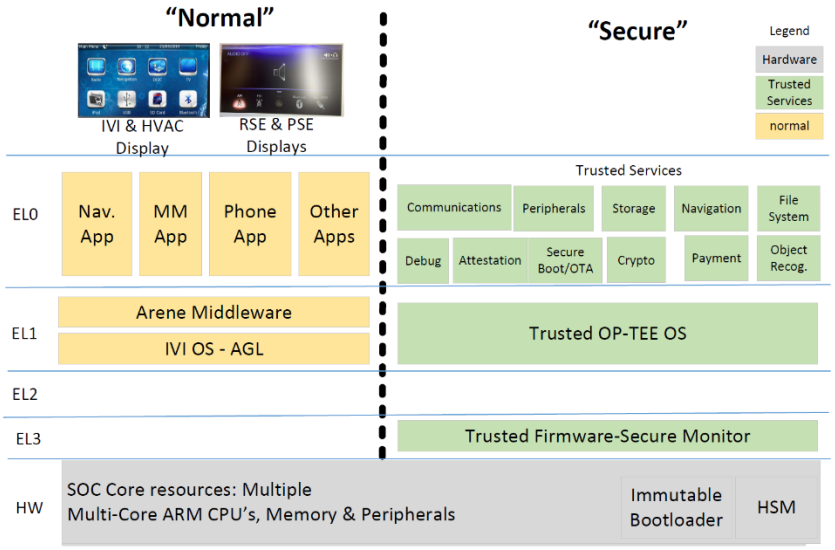

Figure 11 - TrustZone isolation between "normal world" and "secure world".

A TEE can be used to implement many secure services. Some of these services may be generic, such as trusted storage or cryptographic services. Others may be application specific and require custom software implemented in a Trusted Application.

Example Use Case: The security of the update mechanism is critical to prevent an attacker from abusing the update mechanism to install malicious software. The keys used for encryption and signature verification must be kept secret and integrity protected, so they should be managed by the TEE. In addition, the logic itself for verifying that an update package is authentic and decrypting it is security critical, so that logic should be implemented by the TEE.

The update flow is as follows:

1. The normal world update application receives a signed and encrypted update package.
2. The update application does not have access to the key for decryption, so it sends the update package to the TEE for decryption and authentication.
3. The update trusted application decrypts the update package using a key managed by the TEE. It does not share the decrypted package with the normal world update application.
4. The trusted update application verifies the signature of the update package using a key managed by the TEE.
5. If the signature is verified as being issued by Toyota, then the decrypted package is sent back to the normal world update application.
6. The normal world update application can then apply the update as specified.

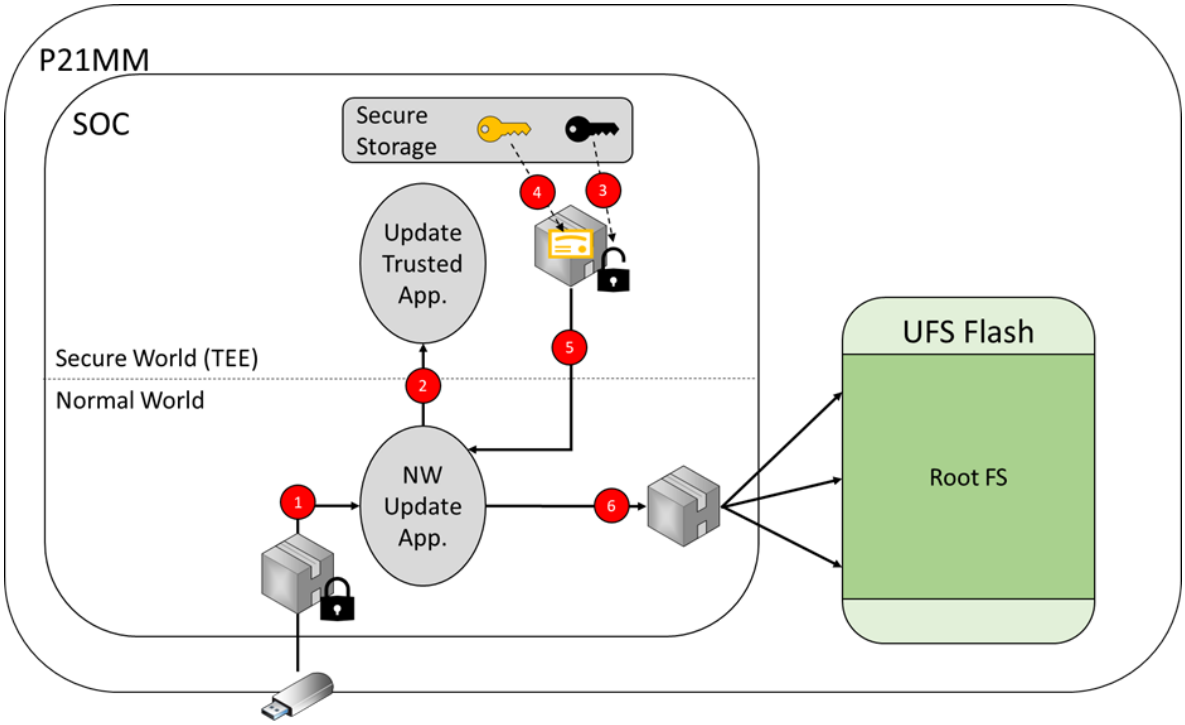

Figure 12 - Update package verification using TEE.

This approach minimizes the risk a vulnerability in the parsing logic of the normal world update application bypasses the authentication of the update. The normal world update application cannot even parse the update image until after the image has been authenticated and decrypted by the high-integrity trusted update application.

7.1.5 HLOS Secure Storage

Threat Scenario: Attackers will try a variety of means to access sensitive information. This could include removing flash memory and extracting its contents or exploiting software vulnerabilities to access the ECU and leak sensitive information.

Countermeasure: Encrypt all sensitive information before storing and limit access to only authorized software. Encryption key is managed by Trusted Execution Environment (TEE). This solution is referred to as High-Level Operating System (HLOS) Secure Storage.

Use Case Example: The remote services application needs to store a long-term authentication token for a backend service. Without encryption, an attacker could extract flash to read the authentication token and potentially use it to extract PII data. To protect the token, the remote services application sends the token to secure storage where it is encrypted and written to flash. At the next boot, the remote services application authenticates with secure storage and requests the token.

An attacker that has exploited a vulnerability in the Bluetooth stack also requests the authentication token from secure storage. However, the attacker is authenticated as the Bluetooth stack, not the remote services application, and so the secure storage mechanism refuses to provide the attacker with the user profile.

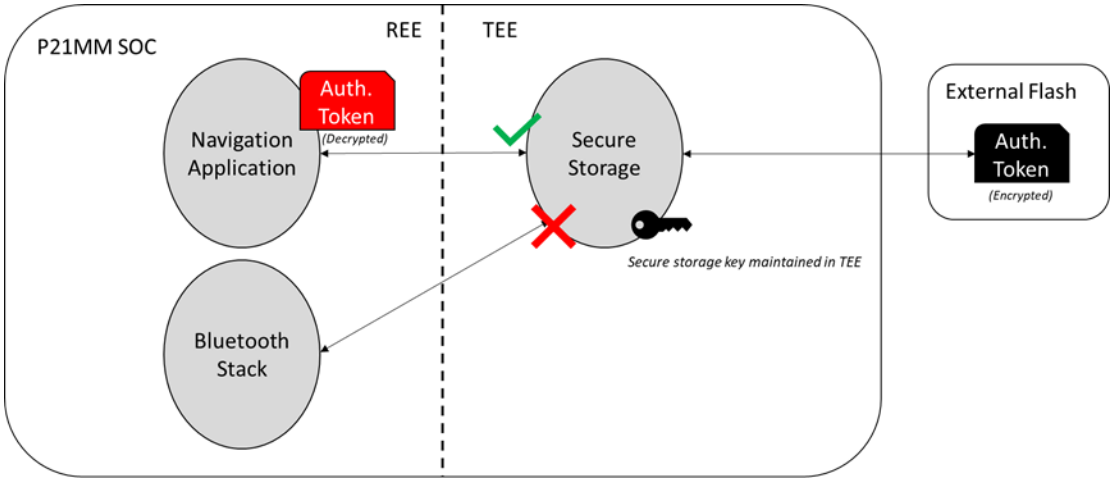

Figure 13 - Secure storage ensures only the correct software can access the sensitive information and that the sensitive information is encrypted.

7.1.6 Full Disk Encryption

Threat Scenario: Linux application software and drivers implement most of the functionality of the ECU. Attackers target this software for reverse-engineering to find vulnerabilities that can be exploited on a running system. The software is typically stored in an external flash that can be removed and read by an attacker.

Countermeasure: Encrypt all external flash contents, including the entire Linux root filesystem. This is commonly referred to as “Full Disk Encryption.” This can be accomplished with minimal performance impact by using inline encryption/decryption blocks on the SoC. This allows flash storage to be treated normally by application software. The data is transparently encrypted before leaving the SoC and transparently decrypted before reception by the flash driver.

Use Case Example:
Initial boot images are encrypted with an SoC secure boot encryption key. All subsequent boot images and the Linux root filesystem are encrypted using the inline crypto block. If an attacker extracts the flash contents, they will only see encrypted data, and the key will be tied to internal SoC secrets that are inaccessible. The attacker is then prevented from reverse-engineering the ECU software.

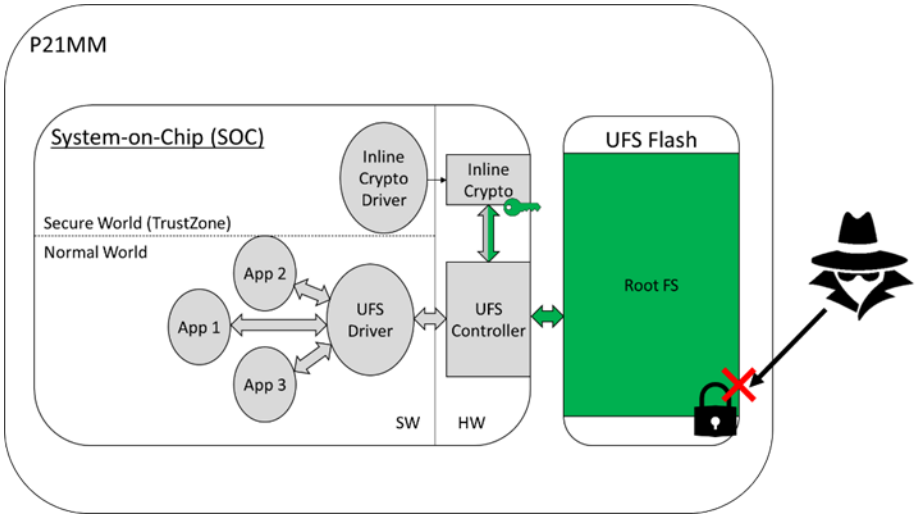

Figure 14 - All contents of external flash are transparently encrypted

7.1.7 SoC Memory Access Configuration

Threat Scenario: Attackers that gain control of a processor core or a peripheral on a privileged bus may be able to access memory that should be reserved for another processor core on the SoC.

Countermeasure: SoCs often contain advanced access control capabilities to restrict access to certain memory ranges and peripherals to only specific processor cores. For example, bus mastering peripherals can be restricted to only certain memory ranges.

A goal of P21 is to leverage all available memory access controls to restrict processors and peripherals access to only the memory ranges necessary. This implements the principle of least privilege to the SoC hardware.

Note that these requirements are not referring to normal Linux memory management, which is concerned with separating memory between kernel and user space and between user space applications. These requirements are in addition to normal Linux memory management and are focused on removing the ability of an entire processor or peripheral from accessing a memory range.

Use Case Example: An attacker compromises the Linux kernel on the application processor. The attacker attempts to send a message to the local CAN interface that is intended to be accessed only by the real-time processor. However, the SoC access controls are configured by a trusted application to only give access to that peripheral to the real-time processor. The attacker's attempts to access the peripheral are detected and blocked, preventing the attacker from broadening their access.

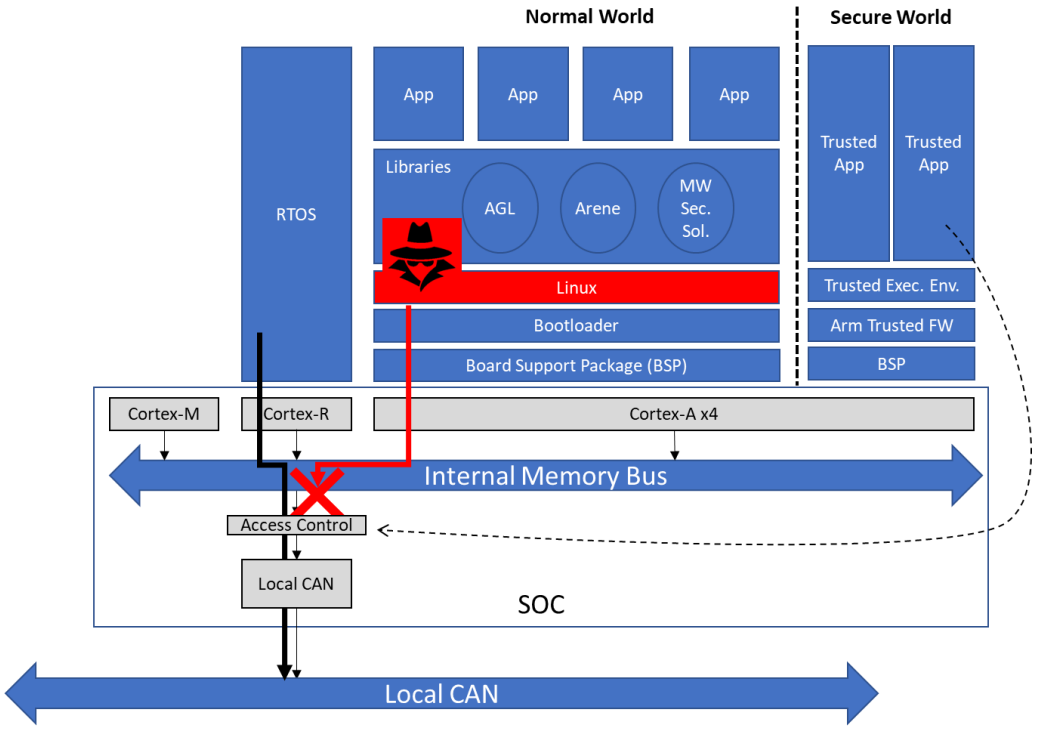

Figure 15 - Attacker attempts to access local CAN but is blocked by SoC access controls.

7.1.8 Logging

Threat Scenario: Cyberattacks often trigger software exceptions (e.g., software crash) or unusual behavior that can be detected and reported by the ECU. However, current vehicles do not forensically log this information for analysis or report it in real-time.

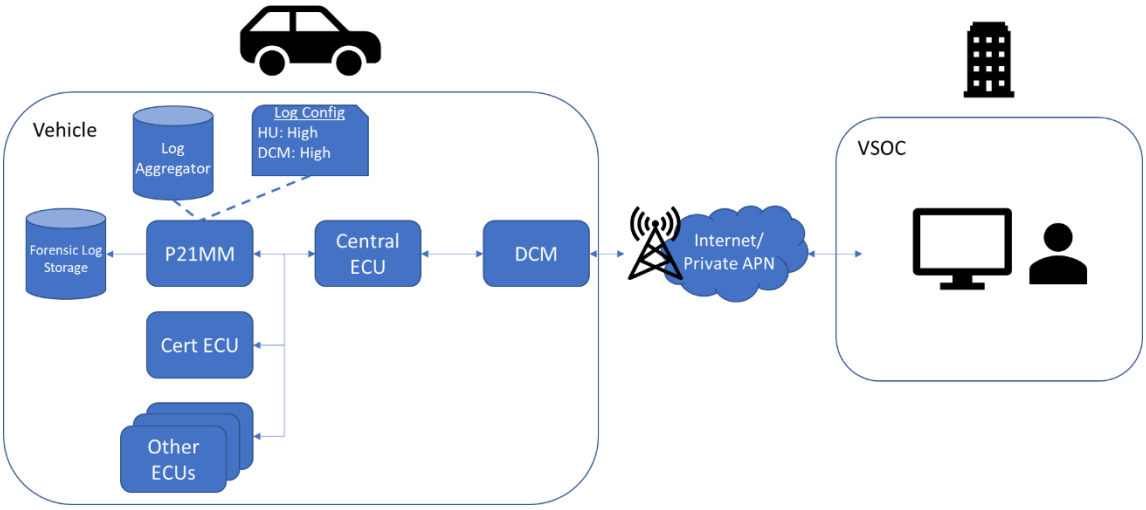

Figure 16 - Real-time logging environment.

Countermeasure: Collect additional cybersecurity logs from across the vehicle and aggregate them in the central ECU. The central ECU and the ECU forensically log the messages in confidentiality and integrity protected storage. The central ECU and/or the ECU immediately forwards critical logs to the Vehicle Security

Operations Center (VSOC). The VSOC then analyzes the logs to detect cybersecurity events in real-time. As needed the VSOC can adjust which events are forwarded to further investigate potential cybersecurity events.

The central ECU and/or ECU should perform pre-processing of the log data to reduce the size of transmission. Logs on the central ECU and ECU can also be downloaded locally, but they must remain confidentiality protected until received by an authorized user.

Use Case Example: Refer to example use case depicted in Figure 17.

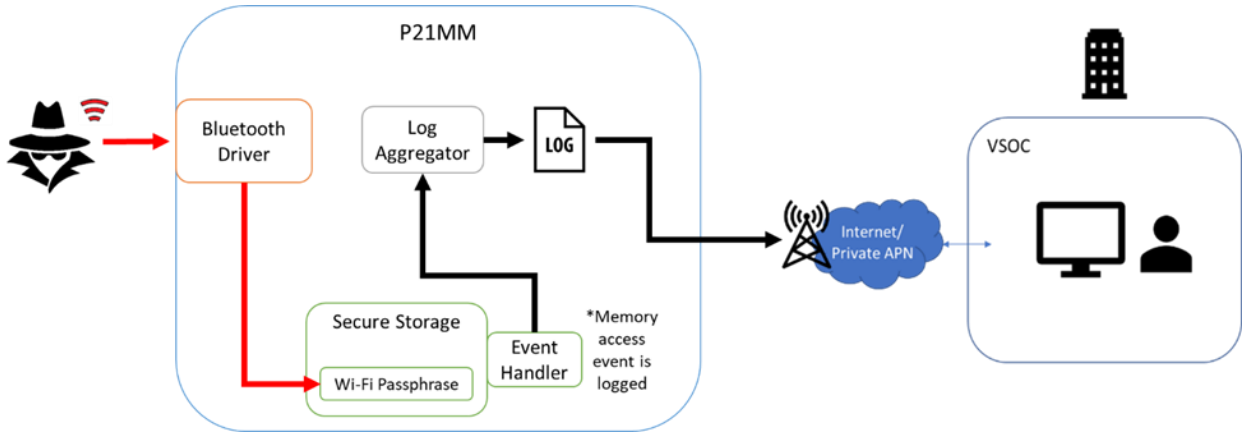

Figure 17 - Example of Malicious Activity of Bluetooth Driver Logged and Sent to VSOC

The log aggregator software in the ECU has been programmed to log the behavior of the Wi-Fi software driver. The Wi-Fi software driver contains a boundary check bug in the processing of a link layer message. An attacker exploits the boundary check bug to gain arbitrary code execution. The attacker attempts to read the MQTT credentials from secure storage. The secure storage service detects the Wi-Fi driver is not authorized to read this data from secure storage and logs the unauthorized access attempt.

The log aggregator on the HU collects the log message from the secure storage service. The log is sent to VSOC for analysis. VSOC sends a request to the HU to increase the log level pertaining to the Wi-Fi software driver. VSOC can then better determine the root cause of the malicious behavior and prepare an update to fix the software bug.

7.1.9 Certificate Management

Threat Scenario: Some applications have their own certificate manage strategy that is independent of other applications or the operating system. This includes having their own list of trusted certificate authorities (CAs) and/or their own methods for certificate revocation checks. These differences introduce potential gaps in security between different software components. Also, it is difficult to determine when there is an expiring or revoked certificate that needs to be updated, since multiple lists must be tracked.

Countermeasure: Create a centralized certificate manager used by all software components. The certificate manager manages a single, authoritative list of root certificates. It also implements a common service or library for performing revocation checking and a common revocation status cache that can be used by all applications

Use Case Example: The MQTT client connects to a 3rd party server. The server presents a certificate that must be validated. The MQTT client uses the certificate manager library or service to validate the certificate chain is linked to a root certificate in the common, trusted root CA store. It also uses the certificate manager library or service to perform revocation checks of every certificate in the chain.

Next, the Vehicle Location Service connects to the Toyota backend through the DCM gateway. The Vehicle Location Service uses the same certificate manager library or service to validate the root certificate and the revocation status.

Finally, the Safety Connect application connects to the Toyota backend through the head unit gateway. The Safety Connect application uses the same certificate manager library, and the revocation check is performed instantly using the cached response from the Vehicle Location Service connection.

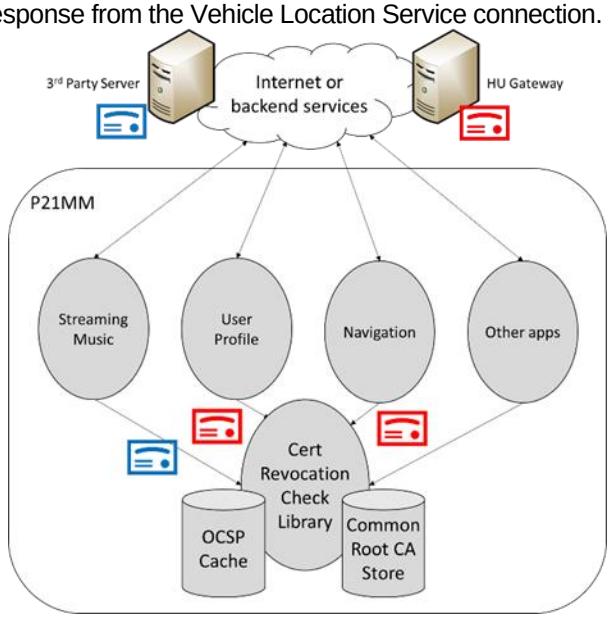

Figure 18 - All applications and services use common root CA store and revocation check implementation.

7.1.10 Communications

Threat Scenario: Attackers will attempt to eavesdrop and manipulate communication to and from the MM device. Critical control messages and sensitive information transferred over these communication channels must have security mechanisms.

Countermeasure: The MM device shall enforce authenticity and confidentiality mechanisms to secure communication. Authenticity can be achieved by verifying the trust of a device so sensitive information is not transferred to attackers. Confidentiality can be achieved by encrypting data so sensitive information is not read by attackers. Satisfying authenticity and confidentiality on the MM device's communication starts with security requirements around each supported communication channel. Figure 19 displays a high-level communication architecture for 24MM.

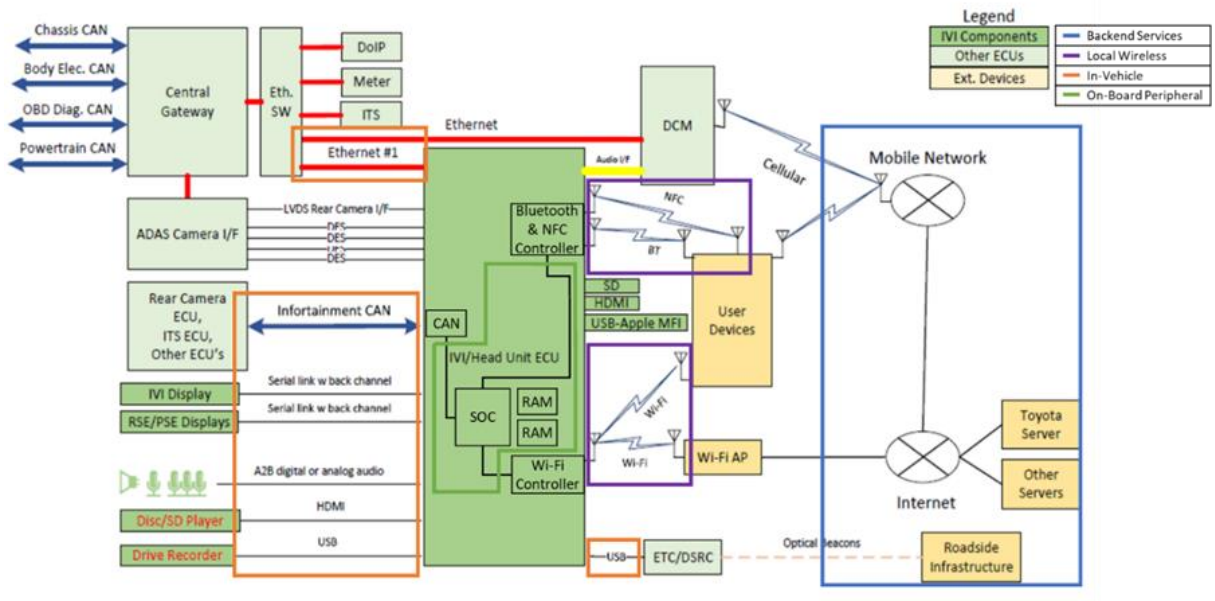

Figure 19 – 24MM Communication Architecture

The anticipated communication channels to be used on 24MM are separated into four (4) categories: backend services, local wireless, in-vehicle, and on-board peripheral. Each category shown in Figure 19 is discussed in the following sections.

Use Case Example: The ECU connects to an access point hosted at a public location such as a coffee shop as shown in Figure 20.

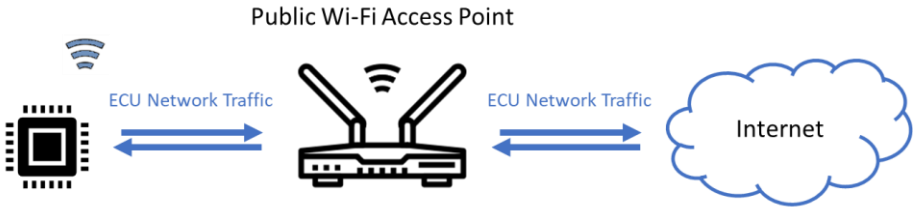

Figure 20 - ECU connected to Public Wi-Fi Access Point

An attacker accesses the same public Wi-Fi access point the ECU is connected to. The attacker then performs a Man-In-The-Middle (MiTM) attack against the ECU and the internet by using a common Address Resolution Protocol (ARP) poisoning technique as shown in Figure 21.

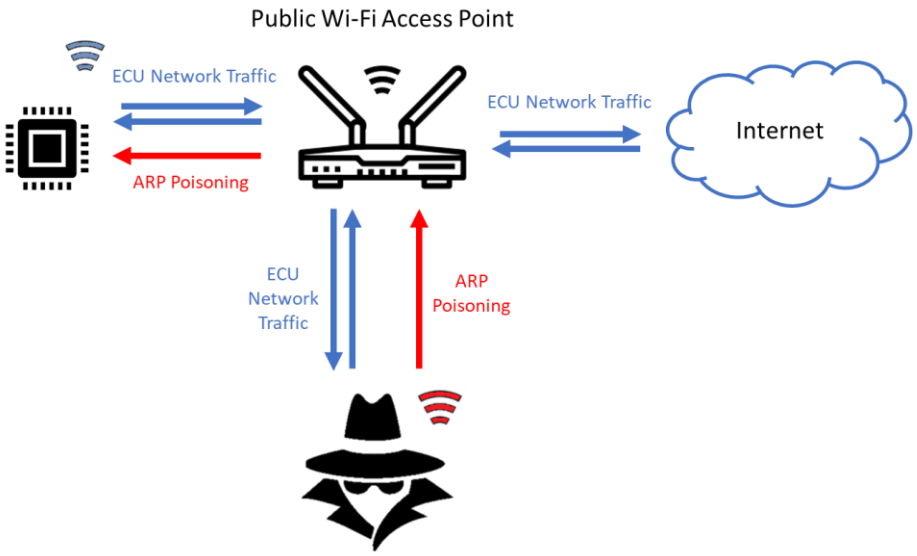

Figure 21 - Attacker ARP Poisoning ECU to MiTM ECU Network Traffic

The attacker can now eavesdrop on all network traffic and can tamper with the contents of message to and from the ECU to the internet. If the ECU uses Transport Layer Security (TLS) to ensure authenticity and confidentiality of network traffic, then the user's data is protected against the attackers MiTM attack.

7.1.11 Software Hardening

Threat Scenario: Software may contain vulnerabilities that could be exploitable by an attacker. Many vulnerabilities can be caught by code reviews, vulnerability scanning, and other means. However, it is always possible a security issue is not detected, and the final software contains unknown vulnerabilities.

Countermeasure: Use techniques to make exploiting software vulnerabilities more difficult and unreliable. This includes:

- Address randomization. Prevents attackers from having consistent addresses of data elements necessary to conduct an exploit, increasing the complexity of an attack.
- No execute mechanisms. Reduces memory locations an attacker can load and execute malicious code.
- Control-Flow Integrity. Detect buffer overflows and invalid code execution paths. This forces the attacker to bypass extra security checks to conduct an exploit, increasing the complexity of the attack.

Use Case Example: Assume a normal software example shown in Figure 22. A user sends a request message to the software to retrieve sensitive information stored in secure storage. The normal behavior for the network driver is to receive the incoming message and then verify the digital signature to authenticate the user and then return the sensitive information to the user.

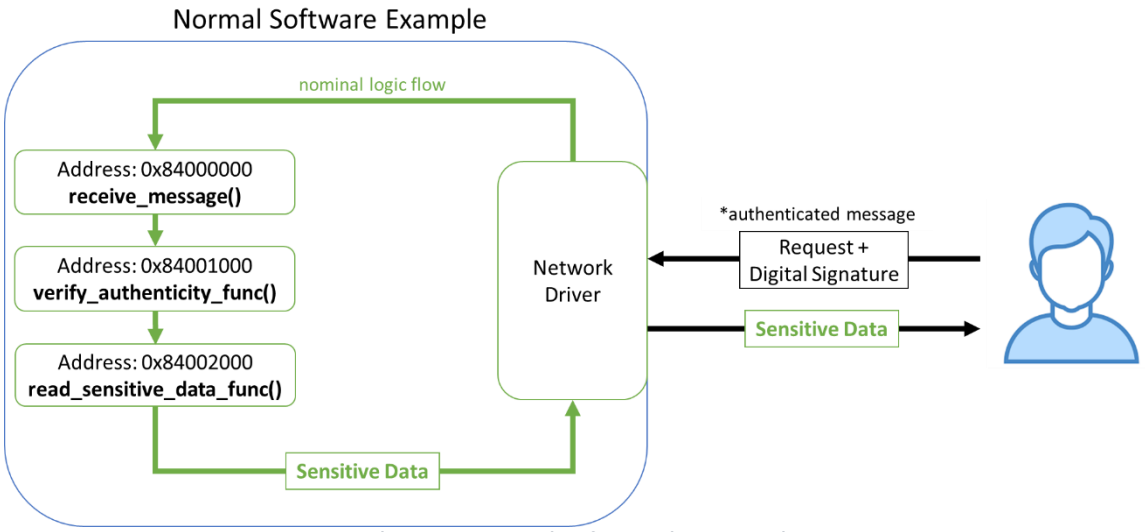

Figure 22 - Normal Software Flow Example

Let us take that same software example but assume a boundary check error exists in the network driver. Let us also assume the attacker was able to acquire the software binary for this example from previous testing and spent the required time to reverse engineer the software flow. Figure 23 shows the attacker and the flow of the software during the attack. The attacker exploits the boundary check error because the software did not have proper runtime security mechanisms like Control Flow Integrity (CFI) and Address Space Layout Randomization (ASLR).

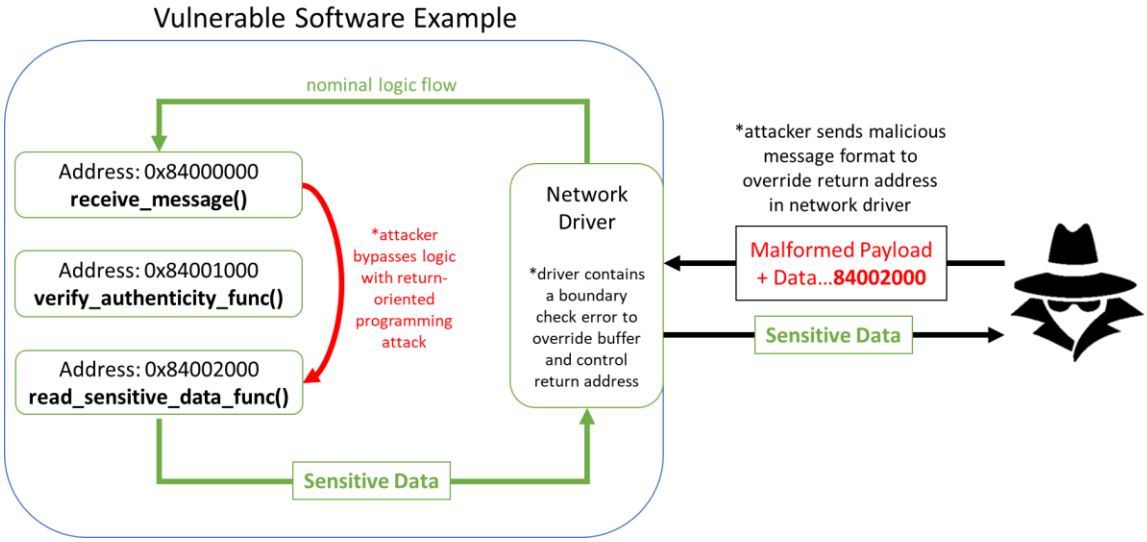

Figure 23 - Vulnerable Software Flow Example

Figure 24 shows how this attack could be prevented by hardening the software. ASLR prevents the attacker from knowing the addresses in memory to bypass the authenticity check because the addresses are properly randomized when the software is loaded. CFI prevents the execution flow from being manipulated because the nominal flow is enforced, which requires the authenticity check to be performed. Finally, buffer overflow detection would also prevent the manipulation of the execution flow because the overflow would be detected before it could be exploited.

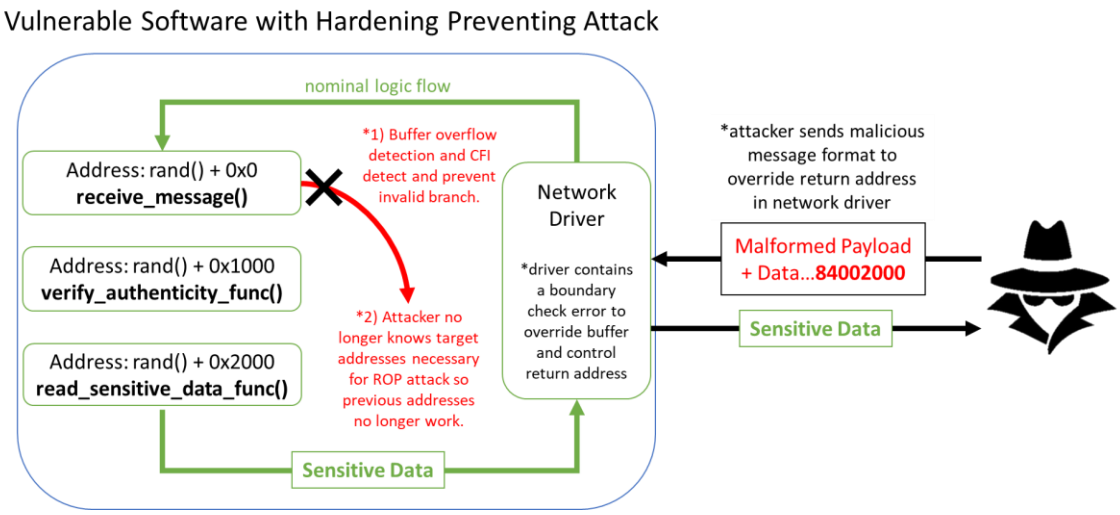

Figure 24 - ASLR, CFI, and buffer overflow detection prevent successful exploitation.

7.1.12 Software Sandboxing

Threat Scenario: If an attacker compromises one software component, they can expand their access to affect other, unrelated software components.

Countermeasure: Create a "sandboxing" system for application software. The sandbox isolates unrelated applications from each other. Sandboxes restrict what resources each application can access to the minimum required.

Linux provides several features to help isolate applications. These features should be combined to implement strong sandboxes with overlapping and reinforcing security. Some of these features include:

- Mandatory Access Control (e.g., SELinux)
- Namespaces
- Cgroups
- Seccomp
- Capabilities

For prior ECUs, SELinux has been the primary, and in some cases only, mechanism to prevent a compromised application from performing unauthorized actions. However, SELinux policies are notoriously difficult to configure to provide robust security and strong isolation. Also, SELinux does not have all the capabilities provided by other isolation features. Complementing SELinux with these other Linux isolation features will enhance the robustness and breadth of application isolation.

Use Case Example: The MQTT client is exploited by an attacker. The attacker has arbitrary code execution within the context of the MQTT client. The attacker then tries to send a message to the CAN process via the CAN inter-process communication mechanism.

However, in this example the MQTT client should never need to send messages onto CAN. Therefore, the sandbox has been configured so that:

1. The MQTT client is in a separate PID namespace and cannot find or manipulate the CAN process.
2. The MQTT client is in a separate network namespace and cannot interact with the CAN IPC mechanism.
3. The MQTT client is only given the capabilities necessary to perform MQTT functions. Therefore, the attacker does not have the privileges necessary to remove the isolation protections.

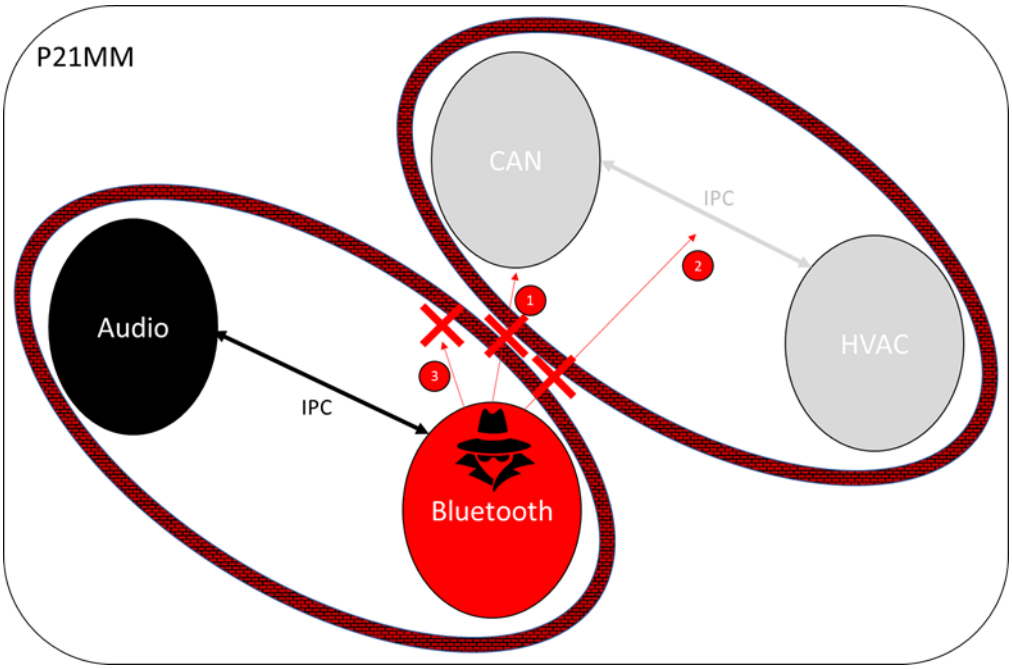

Figure 25 - Attacker prevented from interacting with unrelated processed by namespaces and capabilities restrictions.

References

[1] Auto-ISAC, "Security Development Lifecycle: Best Practice Guide," [Online]. Available: http://autoisac.wpengine.com/wp-content/uploads/2020/02/7_Auto-ISAC-BPs_SDL_02February2020_TLP_White.pdf.

[2] A. Cavoukian, "Privacy by Design: The 7 Foundational Principles," [Online]. Available: https://www.ipc.on.ca/wp-content/uploads/resources/7foundationalprinciples.pdf.

[3] MITRE, "CWE-329: Generation of Predictable IV with CBC Mode," [Online]. Available: https://cwe.mitre.org/data/definitions/329.html.

[4] Linux Kernel, "dm-crypt," [Online]. Available: https://www.kernel.org/doc/html/latest/admin-guide/device-mapper/dm-crypt.html.

[5] Wi-Fi Alliance, "Technical Note: Removal of TKIP from Wi-Fi®Devices," 16 March 2015. [Online]. Available: https://www.wi-fi.org/download.php?file=/sites/default/files/private/Wi-Fi_Alliance_Technical_Note_TKIP_v1.0.pdf.

[6] Qualcomm Technologies, "Trusted User Interface Technical Overview, 80-P7202-14".

[7] Qualcomm Technologies, "Security For Qualcomm Automotive Platforms, 80-PM231-4".