1. Functional Requirements
   1. Master Reset (separate spec?)

**Requirement Text:**

**Definition:**

**Requirement Author:**

**Goal:**

**Rationale:**

**Verification Type:**

* 1. Image, Application, and Script Installation

**Requirement Text:** The system shall only permit authorized, authenticated, and untampered software to be installed onto the system. System software shall only be modified by a Ford approved process.

Only Ford approved code-signing methods shall be used. All code-signing methods shall perform the following security checks:

* Verify the software or script being added or installed is un-tampered, has a valid signature, and has not been modified after the signing process.
* Verify the certificate chain used for signing the software or script leads up to a root public key. The root public key shall be hosted in the certificate store for that corresponding feature. **Reference Platform Digital Certificate Usage**.
* Verification shall occur on the processor where the software will be installed (i.e. CCPU shall verify CCPU applications, VMCU shall verify VMCU applications).

**Definition:** Software includes but is not limited to:

* Software Images
* Applications
* Scripts

Ford approved methods of software modification reference xx.xx.xx.

**Requirement Author:** Justin Mendenhall

**Goal: To prevent the installation of unauthorized or tampered software.**

**Rationale:** Only authorized, authenticated, and untampered software shall be installed on the system. Unauthorized or tampered software may be malicious and will affect system security, may affect system stability, may affect system safety, and may cause the system to operate in an unintended way.

**Verification Type:** DV

* 1. Security Keys

**Requirement Text:** The system shall follow a Ford approved process to authenticate itself with remote services, use secure messaging, and establish secure communication channels.

Each processor requiring access to a remote service, secure messaging, or secure communication shall have a package of at least 8 randomly generated symmetric keys.

Each processor shall not have access to another processor’s keys.

The security keys shall be stored in protected memory locations.

**Definition:** Ford approved authentication processes include SyncP S13a.

Reference A51 Specification for key generation.

Protected memory locations: HSM, SHE, encrypted partition, encrypted KeyBLOB. **Reference Operating System Security ARL.**

**Requirement Author:** Justin Mendenhall

**Goal: To provide credentials for secure communications and secure messaging.**

**Rationale:** The System needs to authenticate with remote services and remote telematics services. These keys and credentials permit the system and the remote service to authenticate and validate each message. Each processor shall only have access to its own credentials.

**Verification Type** DV

* 1. Platform Digital Certificate Usage

**Requirement Text:** The system shall use the following certificates and asymmetric keys for their prescribed purposes and functions using the table below as a template. The completed table shall be captured in a design document and reviewed with the security team.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Key** | **Certificate Store Name & Location** | **Revocation** | **Other Special Behavior** |
| Secure Boot/HAB |  |  |  |  |
| Image Reflash |  |  |  |  |
| Image Update |  |  |  |  |
| Application Installation |  |  |  |  |
| Application Execution |  |  |  |  |
| TLS |  |  |  |  |
| DRM |  |  |  |  |
| Debug Token |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Ford Motor Company shall house the facilities for generating the previously stated certificates and signing code for the platform using its CA infrastructure. Certificates and asymmetric keys required for HAB/Secure Boot shall only be generated by a Ford approved entity.

Signing and verification certificates shall use approved cryptographic algorithms, functions, and key sizes.

The Certificates shall contain the following data:

* *Issuer* – All certificates shall be issued by a Ford trusted Certificate Authority (CA)
  + Code signing certificates shall be issued by Ford Motor Company
  + Other necessary certificates (e.g. DRM, low-level hardware boot, or drivers) shall be issued by the responsible parties and verified by Ford and the Supplier.
* *Certificate Chain* –
  + Root Certificates for the stores shall be issued by Ford’s CA.
  + Partial Chain validation will be used. The intermediate certificate, acting as the “root” certificate, shall be derived from the proper root certificate. The Root certificate shall be issued by Ford’s CA.
  + Individual code-signing certificates shall be derived from the intermediate certificates. These specific code-signing certificates shall not be included in the software image, but shall be used to sign the software.
  + Certificates above the Intermediate CA shall not be included in SYNC Software Images
    - These certificates shall not validate signed files
    - Partial Chain Validation shall be used to ensure the Intermediate certificates are used as the “root” certificate.
  + All private keys and provisioning of the signing keys shall be managed by the CA.
  + Signing of the final production images shall be managed by Ford’s code-signing facilities.
* *Key Strength* - Certificates shall use RSA 2048 bit key.
* *Validity Period* – The Root Certificate for the SYNC code-signing certificates shall be valid for a minimum of 15 years.
* *Attributes* – Signing attributes are permitted to be attached to the package signature.
  + A method to restrict software installation to a single module shall be supported (e.g. ESN signing).
  + Time based installation period may be supported.
* *Purpose* – Signed files shall be validated against the certificate’s purpose.
  + “-purpose any” shall not be used.

The code-signing process shall support command line executables to permit Ford backend systems to implement a dynamic signing process.

**Definition:** Approved Cryptographic functions and key sizes for digital signatures are: RSA-2048

Approved hash functions are: SHA-256

**Requirement Author:**

**Goal: Features requiring a certificate shall use the specified certificate.**

**Rationale:** For features that are required to verify authenticity and integrity, a certificate is required. Using a certificate for whose purpose is defined for a given feature ensure that data was properly issued.

**Verification Type:** DV

* 1. Bluetooth Security

**Requirement Text:** The system shall not permit connections to devices that are not currently paired. The system shall only permit pairing during the explicit pairing process initiated by the consumer. The pairing process is the only period when the connection may be vulnerable. The system shall not reveal sensitive data from a device to be revealed when that device is not connected.

**Definition:**

**Requirement Author:**

**Goal:** **To prevent unpaired devices from connecting to the system and to prevent data leakage.**

**Rationale:** Unpaired devices may be able to connect to a system when a connection is not explicitly initiated by the consumer. This may lead to extraction of sensitive data or permit an attacker to probe the system for weaknesses. By only permitting a previously paired to device to connect to the system or by permitting a device to be paired during an explicit action initiated by the consumer, this reduces potential attack vectors.

**Verification Type:** DV

* 1. Wi-Fi Security: AP Mode

**Requirement Text:**

The System shall only be in AP mode when the system enables Projection Mode as defined by Apple Carplay and Android Auto. The firewall shall dynamically whitelist the Wi-Fi MAC address of the client using Android Auto or Apple Carplay, preventing other devices from connecting to SYNC via Wi-Fi. The system firewall shall be configured to prevent forwarding of packets from the client via SYNC (i.e. the client shall only be permitted to communicate with Android Auto or Apple Carplay). The system firewall shall prevent forwarding packets to the client that did not originate from SYNC. Per Projection Mode, only WPA/WPA2 security modes shall be supported. AP Mode shall not use weaker security modes.

Other configuration settings: AP shall be configured to not support 802.11r (KRACK Attack mitigation).

**Definition:** Reference Internet Gateway APIM SPSS for firewall configuration.

**Requirement Author:** Justin Mendenhall

**Goal: To minimize potential attack vectors on SYNC**

**Rationale:** AP mode will only be available when Projection mode is enabled. Only the device using projection mode shall be able to connect to SYNC. That device can only communicate to its respective projection application. This reduces attack available vectors.

**Verification Type:** DV and configuration review

* 1. Wi-Fi Security: Patches and Mitigation

**Requirement Text:** wpa\_supplicant shall have the following mitigations applied:

* 2015-1
* 2015-2
* 2015-3
* 2015-4
* 2015-6
* 2015-8
* 2016-1
* 2017-1

**Definition:** See <https://w1.fi/security> for vulnerability, patch, and mitigation information.

**Requirement Author:** Justin Mendenhall

**Goal: To remove known vulnerabilities in wpa\_supplicant**

**Rationale:**

**Verification Type:** DV and configuration review

* 1. DRM (other spec?)
  2. Sirius Channel Locking (other spec?)
  3. Door PIN Management (other spec?)
  4. Valet Mode (other spec?)
  5. Application Sandboxing

**Requirement Text:** The system shall only grant the minimum permission required for an application to execute. The system shall restrict an application’s ability to access data, communication paths/channels, and resources not owned by the application. A manifest shall be generated enumerating all data and resources that each application can access that is not owned by that application. The manifest shall be reviewed by the security team.

**Definition:** Reference the OS Security ARL, sections MAC and DAC.

**Requirement Author:** Justin Mendenhall

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

**Rationale:** Application Sandboxing limits an application from accessing resources controlled or owned by another application.

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

* 1. Rootless Operation

**Requirement Text:** Applications and processes shall not execute as root or with permissions equivalent to root. If an application or process requires root or root equivalent access, sign-off shall be required by the security team.

**Definition:** Reference the OS Security ARL, sections MAC and DAC.

Applications include but are not limited to: OS applications and services, applications, and scripts.

**Requirement Author:** Justin Mendenhall

**Goal: To prevent applications or processes from having the ability to modify or interfere with the operation of the system or another application or process.**

**Rationale:** If a process or application has root, root-equivalent, or root-like permissions or capabilities, it can modify files on the system, add/remove accounts, and install applications. Running applications and processes with the minimum amount of permissions required reduces the potential harm it can create if can create, either by design or malicious intent.

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

* 1. Binary Integrity Check

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

Note: Data residing in an Application’s private storage area is excluded from this requirement.

**Definition:** Application is a binary or an executable.

**Requirement Author:** Justin Mendenhall

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

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

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

**Verification Type:** DV or Configuration Review

* 1. Secure Boot

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

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

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

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

**Requirement Author:** Justin Mendenhall

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

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

**Verification Type:** DV

* 1. Data Security

**Requirement Text:** Sensitive data and Confidential data shall not be revealed to unauthorized users on the HMI, over CAN, or other connectivity interface. Sensitive data and Confidential data shall be immediately removed from the system when the associated paired device is deleted. Symmetric and Private Cryptographic keys shall not be revealed on any interface.

**Definition:**

Sensitive P.I.I., e.g. is GPS, Credit Card, DOB, Driving behavior, video or audio, SSN, Biometric or Medical

Confidential P.I.I., e.g., Name, Address, Phone Numbers, infotainment personality settings

Authorized users are paired and connected devices, data owner devices.

**Requirement Author:** Justin Mendenhall

**Goal: To protect consumer data against data leakage and unauthorized access.**

**Rationale:** SYNC stores consumer data (e.g. phonebook, SMS messages) and cryptographic keys.

**Verification Type:** DV

* 1. Secure Messaging

**Requirement Text:** The system shall implement S26, S25, and S23 series requirements.

**Definition:**

**Requirement Author:**

**Goal: To provide a secure and authenticated communication channel between the CCPU and VMCU**

**Rationale:**

**Verification Type:** DV

* 1. Memory Protections

**Requirement Text:** The system shall implement and enable the following memory protection mechanisms for the stack and heap of applications:

* Stack Canaries
* Address Space Layout Randomization (ASLR)
* Position Independent Execution (PIE) / Position Independent Code (PIC)
* Read-Only Relocation (RELRO)
* No Execute (nX) / Data Execute Prevention (DEP)

**Note:** QNX 7.0 toolchain enables Stack Canaries, PIE, PIC, RELRO and nX by default.

**Definition:**

**Requirement Author:** Justin Mendenhall

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

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

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

* 1. Code Obfuscation

**Requirement Text:** Sensitive data not protected by a tamper resistant IC or approved cryptographic functions shall be protected by code obfuscation.

**Definition:** Sensitive data in this context refers to private keys, symmetric keys, private certificates, credentials and critical code.

Coordinate with the security team for approved code obfuscation methods.

Approved cryptographic functions are: AES, RSA, ECC.

**Note:** AES shall NOT use mode EBC for data larger than 1 block.

**Requirement Author:** Justin Mendenhall

**Goal: To prevent extraction and leakage of sensitive and critical data.**

**Rationale:** Sensitive data may be extracted from applications through reverse engineering. If private keys, private certificates, symmetric keys, and credentials are hardcoded or stored in plain text, an adversary may be able to extract them and bypass security mechanisms. Code obfuscation makes this extracted more difficult.

**Verification Type:** DV and code review

* 1. File Validation

**Requirement Text:** The system shall verify that files accessed or processed by the system and its applications shall meet the following:

* The file type must be one that is allowed for the requested operation
* The file format must match the file extension
* If the file requires a digital signature, the file shall be signed and the signature must be valid for the intended purpose
  + Reference Application Signing Requirement
* If the file is not required to be signed, the format shall match the appropriate file type.
  + Reference Fuzz Testing Requirement
  + If the library or application contains a function to perform this validation function, it shall be used prior to executing or processing the file.
  + Else, use the OS command that performs the validation (e.g. Linux: “*file”, QNX: “*file”*)*

All file metadata must be appropriately processed. Reference Fuzz Testing and Malicious Input Validation Requirements.

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: To protect the system against malicious files**

**Rationale:** Files may not match their given file extension or expected format. This may cause the file parser act in an unintended or unexpected manner, which can impair system stability or worse, compromise the system. Verifying the file minimizes this risk.

**Verification Type:** DV

* 1. Debug Ports and Services

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

For Hardware Debug Ports, refer to ARL 020667

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

(Add debug token info)

**Requirement Author:** Justin Mendenhall

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

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

**Verification Type:** DV

* 1. Vehicle Network Protection

**Requirement Text:** The vehicle network shall be protected through a VMCU. The VMCU shall have a CAN signal whitelist enumerating signals which may be read from and written to the vehicle networks by the CCPU.

All VMCU updates, including whitelist updates, shall be signed. The code-signing certificate shall not be the same code-signing certificate used for the accompanying CCPU.

The CCPU shall not have direct access to the vehicle network.

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: To protect the vehicle networks from a compromised CCPU or SoC.**

**Rationale:** If the CCPU becomes compromised, the vehicle network is protected by the VMCU. The VMCU provides protection by having a CAN signal whitelist that only permits specified CAN signals which may be propagated onto the vehicle network by the CCPU.

**Verification Type:** DV

1. Process / Testing / Certification
   1. Threat Modeling

**Requirement Text:** All actors, components, data, and risks in the system shall be enumerated. All risks shall be eliminated, mitigated with sufficient explanation, or accepted and signed off by Ford. Mitigation explanations require sign off by the security team.

The Threat Modeling process used shall be approved by the security team.

**Definition:** Approved Threat Modeling processes included: Microsoft’s Security Development Lifecycle, processes listed in SAE J3061.

Components include the Operating System, middleware, application frameworks, and applications.

**Requirement Author:** Justin Mendenhall

**Goal: To identify potential vulnerabilities or weakness in the system design and to drive corrective action.**

**Rationale:** Threat modeling identifies potential risks in the system and aids in the process of creating a more secure system.

**Verification Type:**  Threat Model Report

* 1. Ethical Hack and Penetration Test

**Requirement Text:** The system shall undergo a penetration test and ethical hack by a 3rd party. The scope of the test shall include but are not limited to components and risks identified in the threat model. Review and acceptance of the 3rd party and the penetration test’s scope shall be performed by the security team.

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: To identify and remediate or mitigate vulnerabilities in the system.**

**Rationale:** Penetration testing and an ethical hack may identify vulnerabilities in the system and can validate or identify shortcomings in the security or the system. This process also identifies remediation and mitigation options of identified vulnerabilities.

**Verification Type:** Report

* 1. Fuzz Testing

**Requirement Text:** Protocols and programs shall be fuzz tested using a tool or engine approved by Ford.

Protocols that shall be fuzz tested include but are not limited to: Wi-Fi, Bluetooth, USB, Ethernet, other network protocols.

The fuzz tested data shall include but is not limited to: User input, media metadata, phone data, IVSU data.

System instability or vulnerabilities identified by fuzz testing shall be remediated.

**Definition:** Approved fuzz testing tools include: Defensics, umap2, AFL.

**Requirement Author:** Justin Mendenhall

**Goal: Test and improve program and system stability by transmitting random data to program interfaces**

**Rationale:** Fuzz Testing provides random data to the inputs of a program. This technique is useful on all systems especially systems with inputs presented to the system by the user through the HMI and indirect means. Fuzz Testing is also useful testing the stability of programs.

**Verification Type:** DV and report

* 1. Malicious Input Validation

**Requirement Text:** Validation shall be performed on interfaces permitting user input. Specific tests for input limits and acceptable input constraints shall be specified in individual functional specifications. The test cases shall validate that the component appropriately handles or ignores inputs with:

* Longer or shorter lengths than specified
* Containing invalid or non-printable characters
* Values that are outside the documented range or list of categories
* Protection against SQL injection (via bind variable, input sanitation, etc.)

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: Remove vulnerabilities introduced by malicious input**

**Rationale:** Many security exploits can be traced back to poor input validation. For example failing to check input length can lead to buffer overflow attacks and allow remote code execution.

This testing/validation reduces system attack vectors.

**Verification Type:** DV

* 1. Export Compliance

**Requirement Text:** SYNC shall comply with export and import requirements regarding the use of Cryptography within the system. All use of Cryptography shall be documented for Ford Legal reference. The report shall contain:

* Feature Name
* Vendor
* Purpose
* Cryptographic details including but limited to:
  + Library name
  + Cryptographic algorithm
  + Key size/length

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: Regulatory Compliance**

**Rationale:**

**Verification Type:** Report

* 1. Software Image Types

**Requirement Text:** The following system software images shall be created:

* DEVTEST
* PRODUCT

DEVTEST shall contain debug utilities enabled, DEV certificates and keys, and PRODUCT certificates and keys.

PRODUCT shall not contain debug utilities. PRODUCT shall only have PRODUCT certificates and keys.

All code signed for production use shall be issued through Ford’s production signing process or explicitly documented in an auditable manner if other entities need signing authority.

None of the images shall include mechanisms to bypass security mechanisms, run non-signed code, or contain undocumented services through any interface.

If such a backdoor mechanism is later discovered, it shall constitute a recall class issue, which the SI shall be liable for.

Installation mechanisms (e.g. via ESN signing) shall be the accepted means to run diagnostics, system access, or reverter code in production via a controlled auditable manner.

{add debug token info)

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: Permit testing on development images and remove development features and tools on production images**

**Rationale:** DEVTEST images permit testing of software. PRODUCT images remove development features and restrict default access to the system.

**Verification Type:** DV

* 1. Open Source Software

**Requirement Text:** The system software shall be scanned using an approved tool. The tool shall produce a report containing the following content:

* Open Source libraries and software
* License type associated with each open source component
* Known CVEs for each open source component

Known CVEs are for internal purposes only.

**Definition:** Approved tools include Appcheck, Blackduck

CVE: Common Vulnerabilities and Exposures

**Requirement Author:** Justin Mendenhall

**Goal: Compile the required information for Open Source Disclosure**

**Rationale:** Licenses associated with open source components require disclosure that those components are being used in system software. Having a process and a tool that can analyze system software facilitates this process.

**Verification Type:** Report

* 1. Vulnerability Remediation

**Requirement Text:** All vulnerabilities shall be assessed and remediated or accepted in accordance to their risk. All remediations and acceptance shall be approved and verified by the security team.

**Definition:**

**Requirement Author:** Justin Mendenhall

**Goal: To reduce the attack footprint of the system through known vulnerabilities**

**Rationale:** Many security exploits can be traced back to libraries containing unpatched vulnerabilities. The vulnerabilities shall be assessed to determine their risk and applicability

**Verification Type:** DV, Code Review, or Component Analysis

* 1. Extraneous Component Removal

**Requirement Text:**  Components that are not used by or required for the system to operate in accordance to system requirements shall be removed. Duplicative components and libraries, including multiple versions of the same component or library, shall be removed when possible.

**Definition:** Components include but are not limited to: Encoders, Decoders, services, daemons, libraries.

**Requirement Author:** Justin Mendenhall

**Goal:** **To minimize attack vectors.**

**Rationale:** Extraneous components may introduce attack vectors into the system. Extraneous components also consume memory. Removal of these components reduce attack vectors and memory usage.

**Verification Type:** Attestation and Design Review