DOC ID:

Test Report ID:

BIS Essential Requirements Test Report for CCTV Technical Requirements Test Report

1. Testing Lab Details

Test Laboratory Name & Address	
Designation Information	BIS
Whether the lab is designated to carry out the tests	YES
Date of Commencement of testing	
Date of Completion of testing	
Test Report No.	
Total No of pages	

This is the test report for validation of technical requirements as prescribed in the BIS ER for CCTV.

2. Applicant Details

Applicant Name and Address	
OEM Name and Address	
Product Details	
Model No	
Interface (s) offered for Test	

Test report prepared by		
Test report reviewed and approved by		
Date of issue of test report		

3. Description of the DUT

Product Name / Model Number	
Product / Device Description	
Software/Firmware Major and Minor Versions	
Kernel version	
Serial Number of the DUT	
Associate Models	

4. Result Summary

Requirement	Parameter Name	Final Result (Pass/Fail/Not Applicable)	Remarks
1.1	Application Debug Interface Protection		
1.2	Device Unique Crypto Keys		
1.3	On-Chip Debug Interface Protection		
1.4	Trusted Execution		
1.5	Secure Storage		
1.6	Tamper Protection		
1.7	IP Protection		
1.7	Boot Image Validation		
1.9	Secure PRNG Usage		
2.1	Memory Protection Controls		
2.2	Data Transit Security		
2.3	Server Connection Validation		
2.4	Banned C Functions		
2.5	Software Bill of Materials		
2.6	Secure Code Review		
2.7a	Digital Signature Pinning		
2.7b	Reverse Engineering Protection		
2.8	Update Process Security		
2.9	Code Signing Verification		
2.10	Firmware Downgrade Protection		
2.11	Scheduled Firmware Updates		
3.1	Wireless Mutual Authentication		
3.2	Wireless Encrypted Channel		

3.3	Trusted Supply Chain		
3.4	Supply Chain Risk Management		
3.5	Proprietary Protocols Management		
4.1	Anti-Counterfeit Measures		
4.2	Threat Mitigation		
4.3	Malware Detection Deployment		
4.4	Supply Chain Risk Assessment		

5. Detailed summary:

Requirement	Parameter Name	Test Case	Result	Remarks	Page
1.1	Application Debug Interface Protection	1.1.1			
		1.1.2			
		1.1.3			
		1.1.4			
1.2	Device Unique Crypto Keys	1.2.1			
		1.2.2			
		1.2.3			
1.3	On-Chip Debug Interface Protection	1.3.1			
		1.3.2			
		1.3.3			
		1.3.4			
1.4	Trusted Execution	1.4.1			
1.5	Secure Storage	1.5.1			
		1.5.2			
		1.5.3			
1.6	Tamper Protection	1.6.1			
		1.6.2			
1.7	IP Protection	1.7.1			
1.8	Boot Image Validation	1.8.1			
		1.8.2			
1.9	Secure PRNG Usage	1.9.1			
		1.9.2			
2.1	Memory Protection Controls	2.1.1			
2.2	Data Transit Security	2.2.1			
		2.2.2			
		2.2.3			

		2.2.4			
		2.2.5			
2.3	Server Connection Validation	2.3.1			
		2.3.2			
		2.3.3			
		2.3.4			
		2.3.5			
2.4	Banned C Functions	2.4.1			
2.5	Software Bill of Materials	2.5.1			
		2.5.2			
		2.5.3			
2.6	Secure Code Review	2.6.1			
2.7a	Digital Signature Pinning	2.7a.1			
		2.7a.2			
2.7b	Reverse Engineering Protection	2.7b.1			
2.8	Update Process Security	2.8.1			
2.9	Code Signing Verification	2.9.1			
		2.9.2			
2.10	Firmware Downgrade Protection	2.10.1			
2.11	Scheduled Firmware Updates	2.11.1			
3.1	Wireless Mutual Authentication	3.1.1			
3.2	Wireless Encrypted Channel	3.2.1			
		3.2.2			
		3.2.3			
3.3	Trusted Supply Chain	3.3.1			
3.4	Supply Chain Risk Management	3.4.1			
3.5	Proprietary Protocols Management	3.5.1			
4.1	Anti-Counterfeit Measures	4.1.1			
4.2	Threat Mitigation	4.2.1			
4.3	Malware Detection Deployment	4.3.1			

4.4	Supply Chain Risk Assessment	4.4.1			

DOC ID:

Test Report ID:

Test Report for ER 01:2024 Essential Requirements for security of CCTV

Test Result Summary

#	Sections	Total No. of Test Case	Pass	Fail	Not Applicable (N/A)	Not Tested	Not Supported	Blocked	In Progress
1	Hardware Level Security Parameter								
2	Software/Firmware								
3	Secure Process Conformance								
4	Security Conformance at Product Development Stage								
	Total cases								

Pass	*The test case passed with no exceptions and meets the requirement specified in ER 01:2024 Essential Requirements for security of CCTV as per current test methodology*
Fail	*The test case failed to meet the requirement specified in ER 01:2024 Essential Requirements for security of CCTV as per current test methodology. Details of the failure are described in respective sections*
In Progress	*Testing is in progress and the lab has required details to complete testing*
Blocked	*Other test case failures prevented the execution of this test (OR)* *More details are required from the OEM/ODM to execute the test case.*
Not Applicable (N/A)	*The test case is not applicable to the product under test.*
Not Supported	*Feature is Not supported by the device so the test could not be performed.*
Not Tested	*Not tested. The feature is supported by the product under test, but external factors (lab configuration, e.g.) prevented execution of the test.*

.

Detailed Test Results

1. Hardware Level Security Parameter

1.1 Test Secure Debug Interfaces

Requirement Description
Verify that application layer debugging interfaces such as USB, UART, and other serial variants are disabled or protected by a complex password.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Datasheet of the SoC being used in the device.
- Documentation related to ports/interfaces enabled in the production devices and the related access control mechanism for protection of the same.
- Process flow of the Manufacturing/Provisioning of the device.

Test Plan
Total number of test cases: 4

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.1.1
Test Name: TC_ID_SECURE_DEBUG_INTERFACES

Objective:
Identification of the availability of debugging interfaces such as USB, UART, and other serial variants through the Datasheet of the SoC being used in the device under test.

Tools used:

Test Execution Steps:
1. Obtain the document from OEM regarding the debugging interfaces available in the DUT.
2. Verify the document for the presence of any debugging interfaces.

Expected Results for Pass:
The debugging interface present in the device is identified as per OEM documentation and their default state is defined.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.1.2
Test Name: TC_VERIFY_SECURE_DEBUG_INTERFACES

Objective:
Verification and validation of the ports/interfaces enabled in the production devices and the related access control mechanism for protection of the same as declared in the vendor documentation.

Tools used:

Test Execution Steps:

1. Configure test devices to match production settings.
2. OEM has to provide the Information on the debugging interfaces and their default state, and the protection mechanisms like password complexity, access control implemented in the device.
3. Attempt to interact with each debug interface using OEM provided by the debugging and communication tools to validate their operational status.
4. Verify whether the interfaces are as documented, disabled or protected, attempt-controlled access to verify security measures are effective.

Expected Results for Pass:
The device complies with Information provided by OEM on the debugging interfaces and their default state, and the protection mechanisms like password complexity, access control implemented in the device.

Test Observations:

.

Evidence Provided

Test Case Result:

TEST 3

Test Case: BIS-1.1.3

Test Name: TC_OEM_SECURE_DEBUG_INTERFACES

Objective:
Testing, in presence of OEM team, to verify the enabling/disabling of all the ports and debugging interfaces such as USB, UART, and other serial variants using their relevant hardware-based debuggers and access control mechanisms in case the interface is enabled.

Tools used:

Execution Steps:
1. Schedule a session with the OEM team to oversee the testing process.
2. Systematically test each interface with the OEM team to confirm the enablement status corresponds with documented expectations.
3. Use hardware-based debuggers and access control mechanisms to validate that interfaces are secure or disabled as claimed.

Expected Results for Pass:
Verification of enable/disable status for each interface, validated in presence of OEM team.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 4

Test Case: BIS-1.1.4

Test Name: TC_PROCESS_SECURE_DEBUG_INTERFACES

Description:
Process audit of the manufacturing facility to validate the vendor's claim regarding the debugging interfaces which are closed/disabled during provisioning.

Tools used:

Execution Steps:

1. Prepare an audit checklist based on the vendor's security claims and requirements for debugging interfaces.
2. Review the provisioning process, ensuring it matches the documented claims.
3. Assess the block connection diagrams and any other provided documentation to ensure there is consistency in how interfaces are managed during device provisioning.

Expected Results for Pass:
The manufacturing process adheres to the vendor's claims about disabling or securing debugging interfaces during device provisioning as verified from vendor’s Block connection diagrams or any other relevant documents.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.2 Verify Unique Cryptographic Keys and Certificates

Requirement Description
Verify that cryptographic keys and certificates are unique to each individual device.

DUT Confirmation Details
DUT Software Details

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- List of all keys and certificates being used in the device ecosystem.
- Key management life cycle (purpose, generation, storage, destruction/zeroization, validity, key changeover/rotation).

Test Plan
Total number of test cases: 3

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.2.1
Test Name: TC_OEM_VERIFY_UNIQUE_CRYPTOGRAPHIC_KEYS

Objective:
Identifying all the keys and certificates being used in the device ecosystem and verification through testing in presence of OEM team.

Tools used:

Test Execution Steps:

1. Coordinate a session with the OEM team to conduct joint testing.
2. Use tools like key and certificate management software to identify and inventory all the keys and certificates in the device ecosystem.
3. Test each key and certificate to ensure it functions as intended within the device's operational environment.
4. Validate the keys and certificates against the provided list to ensure there are no discrepancies.

Expected Results for Pass:
Verification records confirming each key and certificate is accounted for, functional, and matches the provided documentation.

Test Observations:

Evidence Provided:

Test Case Result:
.

TEST 2

Test Case: BIS-1.2.2
Test Name: TC_CODE_VERIFY_UNIQUE_CRYPTOGRAPHIC_KEYS

Objective:
Identifying all the keys and certificates being used in the device ecosystem and verification through code review.

Tools used:

Test Execution Steps:

1. Conduct a structured review of the codebase looking for implementation of cryptographic functions and the usage of keys and certificates.
2. Pay special attention to how the keys are generated, stored, and used within the code.
3. Look for any hard-coded or improperly handled keys or certificates.
4. Ensure that best practices for secure coding are adhered to, particularly around cryptography.

Expected Results for Pass:
A code review result that shows the uniqueness about the keys and certificates being used in the device ecosystem.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-1.2.3
Test Name: TC_PROCESS_VERIFY_UNIQUE_CRYPTOGRAPHIC_KEYS

Objective:
Identifying all the keys and certificates being used in the device ecosystem and verification through process audit of the key-life cycle process.

Tools used:

Execution Steps:

1. Review the key management lifecycle documentation provided by the OEM.
2. Trace the lifecycle of a sample set of keys from generation to destruction/zeroization.
3. Ensure that processes for key generation, storage, usage, and destruction are securely implemented and follow documented procedures.
4. Look for evidence of key rotation and renewal practices and validate their effectiveness.

Expected Results for Pass:
An audit that shows the uniqueness about the keys and certificates being used in the device ecosystem.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.3 Test On-Chip Debug Interface Security

Requirement Description
Verify that on-chip debugging interfaces such as JTAG or SWD are disabled or that available protection mechanism is enabled and configured appropriately.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Datasheet of the SoC being used in the device.
- Documentation related to ports/interfaces enabled in the production devices and the related access control mechanism for protection of the same.
- Process flow of the Manufacturing/Provisioning of the device.

Test Plan
Total number of test cases: 4

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.3.1
Test Name: TC_ID_TEST_ON-CHIP_DEBUG_INTERFACE_SECURITY

Objective:
Identification of the availability of debugging interfaces such as JTAG or SWD through the Datasheet of the SoC being used in the device under test.

Tools used:

Test Execution Steps:

1. Use the datasheet of the SoC to understand the pinout and communication protocols.
2. Document the type of interfaces found and their characteristics (e.g., data rates, protocols, security features).

Expected Results for Pass:

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.3.2
Test Name: TC_VERIFY_TEST_ON-CHIP_DEBUG_INTERFACE_SECURITY

Objective:
Verification and validation of the ports/interfaces enabled in the production devices and the related access control mechanism for protection of the same as declared in the vendor documentation.

Tools used:

Test Execution Steps:

1. Cross-reference the device's production configuration with the documentation to understand which interfaces should be enabled or disabled.
2. Systematically attempt to interface with each documented port using the appropriate debugging tools and protocols.
3. Attempt to access or bypass any disabled interfaces to test the effectiveness of the access control mechanisms.
4. Document the methods and tools used for testing each interface, along with the results of these tests.

Expected Results for Pass:
The ports/interfaces enabled in the production devices and the related access control mechanism for protection are same as declared in the vendor documentation.
The ports that are enabled are protected by the complex password and access control mechanisms.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-1.3.3
Test Name: TC_OEM_TEST_ON-CHIP_DEBUG_INTERFACE_SECURITY

Objective:
Testing, in presence of OEM team, to verify the enabling/disabling of all the ports and debugging interfaces such as JTAG or SWD using their relevant hardware-based debuggers and access control mechanisms in case the interface is enabled.

Tools used:

Execution Steps:

1. Arrange for the OEM team to be present during the testing sessions.
2. Present the test plan and expectations to the OEM team prior to starting, ensuring transparency and agreement on the process.
3. Utilize the OEM's expertise to access proprietary tools or knowledge about the device that may be required for thorough testing.
4. Execute the test plan for each interface, with the OEM team observing and participating as needed.
5. Record the process and outcomes meticulously, ensuring any enabling or disabling of ports is witnessed and confirmed by the OEM representatives.

Expected Results for Pass:
A test report with the OEM team’s acknowledgment and verification of the status of each port and interface. This report confirms that all debugging interfaces are enabled or disabled as per the vendor's stated policy.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 4

Test Case: BIS-1.3.4
Test Name: TC_PROCESS_TEST_ON-CHIP_DEBUG_INTERFACE_SECURITY

Description:
Process audit of the manufacturing facility to validate the vendor's claim regarding the debugging interfaces which are closed/disabled during provisioning.

Tools used:

Execution Steps:

1. Review the vendor's process flow documentation for the manufacturing/provisioning of the device.
2. Match each step of the actual process with the documented process, noting any deviations.
3. Verify that the disabling of debugging interfaces is part of the standard process and is performed consistently.

Expected Results for Pass:
The process of provisioning the debugging interfaces are verified from the vendors documents.

Test Observations:

Evidence Provided:

Test Result:

Overall Test Result

1.4 Test Trusted Execution Implementation

Requirement Description
Verify that trusted execution is implemented and enabled, if available on the device SoC or CPU.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Datasheet of the SoC being used in the device.
- User manual/ Technical specifications of the device.
- Code snippets of the TEE API call, wherever applicable.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.4.1
Test Name: TC_TEST_TRUSTED_EXECUTION_IMPLEMENTATION

Objective:
Identifying whether TEE/SE/TPM is available or not in the device through the SoC datasheet and technical documentation submitted by the vendor.

Further assessment is done based on scenarios as applicable to device as defined below:

- **CASE 1: TEE/SE/TPM is not available:** No further assessment.
- **CASE 2: TEE/SE/TPM is available and enabled:** Verification through code-review that crypto functions are called through TEE/SE/TPM APIs.
- **CASE 3: TEE/SE/TPM is available but not enabled by the vendor:** Termed as non-conformance to the requirement. OEM is required to enable and implement the TEE/SE/TPM.

Tools used:

Test Execution Steps:

1. Document Review:
    - Review the SoC datasheet and technical documentation provided by the vendor to determine if TEE/SE/TPM capabilities are present in the device.
    - Analyze the user manual or technical specifications to understand how the TEE/SE/TPM should be implemented and used within the device ecosystem.
2. Lab Setup and Preparation:
    - Establish a secure lab environment with the required tools for interfacing with and analyzing TEE/SE/TPM functionalities.
3. Execution:
   CASE 1: TEE/SE/TPM is not available:
    - Validate through the SoC datasheet that TEE/SE/TPM capabilities are indeed absent.
    - Confirm that no further testing related to TEE/SE/TPM is necessary.

   CASE 2: TEE/SE/TPM is available and enabled:
    - Verify the presence of TPM/TEE/SE either manually or using suitable tools. Verify the logs or any information related to the use of TPM/TEE/SE.
    - Perform static code analysis and verify any information related to the use of TPM.

CASE 3: TEE/SE/TPM is available but not enabled by the vendor:

- Identify through code analysis why TEE/SE/TPM is not enabled and whether this is due to misconfiguration or vendor decision.
- Document the findings and discuss the required actions with the OEM to enable TEE/SE/TPM features as per security requirements.

Expected Results for Pass:

- **CASE 1:** A report confirming the absence of TEE/SE/TPM on the SoC, implying compliance with this scenario, and indicating that no further TEE/SE/TPM-related testing is required.
- **CASE 2:** A report, detailing the results with a focus on the usage of TEE/SE/TPM APIs. The report should confirm the correct implementation and highlight any potential security issues, providing a clear path to remediation.
- **CASE 3:** A detailed account of non-conformance if the TEE/SE/TPM is present but not enabled.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.5 Verify Secure Storage of Sensitive Data

Requirement Description
Verify that sensitive data, private keys, and certificates are stored securely in a Secure Element, TPM, TEE (Trusted Execution Environment), or protected using strong cryptography.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- List of all keys and certificates being used in the device ecosystem.
- List of all the sensitive data with their intended usage and secure storage mechanism(s) as implemented along with secure configurations to be enabled in the device.
- Key management life cycle (purpose, generation, storage, destruction/zeroization, validity, key changeover/rotation) private keys and certificates.

Test Plan
Total number of test cases: 3
Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.5.1
Test Name: TC_OEM_VERIFY_SECURE_STORAGE_OF_SENSITIVE_DATA

Objective:
Identifying all the keys and certificates being used in the device ecosystem, sensitive data, and their storage mechanism(s); and verification through testing in presence of OEM team.

Tools used:

Test Execution Steps:

1. Prepare a test plan outlining the methods for identifying all keys, certificates, and sensitive data storage mechanisms.
2. Coordinate with the OEM team to schedule a testing session where they can witness and assist.
3. Utilize testing tools appropriate for secure storage verification, such as hardware security modules (HSMs) or equivalent, to validate the encryption and access controls.
4. Document the testing process and capture evidence of the storage mechanisms in action

Expected Results for Pass:
A test report confirms all keys, certificates, and sensitive data are securely stored according to the mechanisms and configurations provided in the OEM's documentation.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.5.2
Test Name: TC_CODE_ VERIFY_SECURE_STORAGE_OF_SENSITIVE_DATA

Objective:
Identifying all the keys and certificates being used in the device ecosystem, sensitive data, and their storage mechanism(s); and verification through code review.

Tools used:

Test Execution Steps:

1. Employ a SAST tool in the lab to perform static code analysis focusing on the implementation of secure storage mechanisms.
2. Manually review code segments that handle the storage and access of sensitive data, keys, and certificates, ensuring they adhere to security best practices.
3. Look for code that interacts with TPM, TEE, or Secure Element and review the associated API calls to confirm they are being used securely and correctly.
4. Verify that all sensitive data operations in the code are properly authenticated, authorized, and logged.

Expected Results for Pass:
A detailed analysis from Software Composition Analysis tool, supplemented with manual review findings. The code properly implements and uses secure storage solutions like TPM/TEE/SE for handling sensitive data.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-1.5.3
TestName: TC_PROCESS_ VERIFY_SECURE_STORAGE_OF_SENSITIVE_DATA

Objective:
Identifying all the keys and certificates being used in the device ecosystem, sensitive data, and their storage mechanism(s); and verification through process audit of the key-life cycle process.

Tools used:

Execution Steps:

1. Analyze the key management life cycle documentation provided, including the processes for key generation, storage, destruction/zeroization, validity, and rotation.
2. Investigate the actual key management practices within the environment to ensure they align with the documented processes.
3. Assess the secure storage solutions in the context of the key management life cycle, especially during key generation and destruction phases.
4. Confirm that the key management processes are designed to maintain the integrity and confidentiality of private keys and certificates throughout their lifecycle.

Expected Results for Pass:
An audit that confirms the key management life cycle is being followed as documented. The keys and certificates are securely managed from their creation to destruction, using strong cryptography.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.6 Check Tamper Resistance Features

Requirement Description
Verify the presence of tamper resistance and/or tamper detection features.

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Measures available in the device to prevent software tampering.
- Measures available in the device to prevent hardware tampering.

Test Plan
Total number of test cases: 2

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.6.1
Test Name: TC_CHECK_TAMPER_RESISTANCE_FEATURES_SOFTWARE

Objective:
Testing, in presence of OEM team, to verify the measures implemented in the device to prevent software tampering.

Tools used:

Test Execution Steps:

1. Verify the software tampering resistance and detection mechanisms implemented in the DUT in the vendor's document.
2. Verify whether the claims in the document are correctly implemented in the DUT or not.
3. Attempt to tamper any software in the DUT and observe the DUTs response.

Expected Results for Pass:
The software tamper resistance and detection features in the device are functioning as intended.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.6.2
Test Name: TC_CHECK_TAMPER_RESISTANCE_FEATURES_HARDWARE

Objective:
Testing, in presence of OEM team, to verify the measures implemented in the device to prevent hardware tampering.

Tools used:

Test Execution Steps:

1. With the OEM team present, inspect the physical device for tamper-evident seals, chassis intrusion detection mechanisms, or other physical security measures.
2. Evaluate the response of the device to simulated tampering attempts, like opening the device casing or attempting to access secured hardware components. Ensure that these tests are non-destructive and approved by the OEM.
3. Check for hardware security features like TPMs that provide physical security controls.

Expected Results for Pass:
The hardware tamper resistance and detection features in the device are functioning as intended.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.7 Test Intellectual Property Protection Enablement

Requirement Description
Verify that any available Intellectual Property protection technologies provided by the chip manufacturer are enabled.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image:

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Datasheet of the SoC.
- Documentation regarding the Intellectual Property protection technologies provided by the chip manufacturer which have been enabled.
- In case no Intellectual Property protection technologies are being provided by the chip manufacturer, then a declaration stating the same.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.7.1
TestName:
TC_TEST_INTELLECTUAL_PROPERTY_PROTECTION_ENABLEMENT

Objective:
Testing, in presence of OEM team, to verify the enabling of the Intellectual Property protection technologies provided by the chip manufacturer, if available.

Tools used:

Test Execution Steps:

- With IP Protection Technologies:
  1. Verify the operational status of IP protection technologies using the prescribed methods, which may include checking the SoC's configuration registers or using diagnostic commands.
  2. Attempt to read or manipulate protected areas of the chip to test the response of IP protection measures.
  3. Monitor for alerts, access denials, or any other behaviors that indicate the IP protection features are active.
- Without IP Protection Technologies:
  1. Confirm that the lack of IP protection is by design and document the declaration from the chip manufacturer.
  2. Ensure that alternative security measures are in place to protect the intellectual property associated with the SoC if the manufacturer does not provide dedicated IP protection technologies.

Expected Results for Pass:

- In the case of enabled IP protection, it is verified that the ip protection technologies have been enabled.
- If no IP protection technologies are present, the report should include the manufacturer's declaration for the same.

Test Observations:
Evidence Provided:
Test Case Result:
Overall Test Result

1.8 Verify Boot Image Signature Validation

Requirement Description
Verify the device validates the boot image signature before loading.

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Datasheet of the SoC.
- Technical specifications of the device regarding secure boot (should consist of keys involved and their management life cycle*, signature validation process and any other secure mechanisms if implemented).

Test Plan
Total number of test cases: 2

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.8.1
Test Name: TC_VERIFY_BOOT_IMAGE_SIGNATURE_VALID

Objective:
Testing, in presence of OEM team, to verify that the device boots up successfully with the documented secure boot process when a valid boot image is provided.

Tools used:

Test Execution Steps:

1. Review the datasheet and technical specifications for the device's secure boot process, including key management and signature validation steps.
2. With the OEM team present, provide a valid boot image with a proper signature as expected by the device.
3. Boot the device and observe the boot process, ensuring that all steps align with the documented secure boot sequence.
4. Verify that the boot loader is checking the signature against the correct public key and that all cryptographic verifications are performed.

Expected Results for Pass:

- The device boots up successfully, confirming that the secure boot process is correctly implemented and functional when the valid image is provided.
- The successful verification of the boot image's signature and any other related security checks performed during the process is provided.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.8.2
Test Name: TC_VERIFY_BOOT_IMAGE_SIGNATURE_INVALID

Objective:
Testing, in presence of OEM team, to verify that the device does not boot up when a tampered boot image (like with missing signature, invalid signature) is provided.

Tools used:

Test Execution Steps:

1. In collaboration with the OEM team, attempt to boot the device with a modified boot image. This image should have an altered signature, or the signature should be completely removed to simulate tampering.
2. Observe the device's response to the tampered boot image.
3. Document any error messages, system behaviors, or lack of boot progress that indicates the device recognizes the image as tampered.

Expected Results for Pass:

- The device does not boot with the tampered image. It rejects the image based on the failed signature validation.
- The device's reaction to the tampered image confirms that the secure boot mechanism is protecting against unauthorized software execution is reported.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

1.9 Check Cryptographic PRNG Utilization

Requirement Description
Verify usage of cryptographically secure pseudorandom number generator (PRNG) on embedded device (e.g., using chip provided random number generators).

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Documentation regarding the random generators (either hardware based, or software based or both) being used in the device with their intended usage.
- In case hardware based random number generators are being used, vendors shall submit the following:
- Datasheet of the SoC
- Technical specifications of the device regarding random generators
- In case software based random number generators are being used, vendors shall provide the libraries being used for the same.

Test Plan
Total number of test cases: 2

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-1.9.1
Test Name: TC_CHECK_CRYPTOGRAPHIC_PRNG_UTILIZATION_DOC

Objective:
Verification of the documentation provided by the vendor regarding the random number generators being used in the device.

Tools used:
Test Execution Steps:

1. Acquire and review the documentation provided by the vendor, including the datasheet of the SoC and technical specifications regarding the random number generators.
2. Check if the hardware-based random number generators' documentation details their compliance and describes the entropy source and generation algorithms.
3. For software-based generators, review the documentation for the libraries being used and ensure they meet industry-standard criteria for cryptographic use.

Expected Results for Pass:
The documentation review confirms that the random number generators, whether hardware or software, are suitable for cryptographic purposes according to industry standards.

Test Observations:
Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-1.9.2
Test Name: TC_CHECK_CRYPTOGRAPHIC_PRNG_UTILIZATION_CODE

Objective:
Verification through code review that random number generators or related libraries as applicable are being used in the device.

Tools used:

Test Execution Steps:

1. Run Static Code Analysis tool against the device's codebase, specifically focusing on the implementation and invocation of random number generation functions.
2. Review the analysis results from the tool for any potential issues regarding the usage of random number generators, such as the use of deprecated functions, insufficient entropy, or patterns that could lead to predictability.
3. Manually review segments of code that handle critical security functions, including the initialization and calling of PRNG (Pseudo Random Number Generators) libraries, to supplement the automated analysis and to understand the context which automated tools might miss.

Expected Results for Pass:
The tool yielded a report detailing the security of the PRNG-related code within the device's software, confirming that the PRNGs are used correctly and follow best practices for security. The manual review validated Software tool's findings to ensure that the PRNGs not only function correctly but also provide the necessary security guarantees.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

2. Software/Firmware

2.1 Verify Memory Protection Mechanisms

Requirement Description
Verify that memory protection controls such as ASLR and DEP are enabled by the embedded/IoT operating system, if applicable.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Declaration of the memory protection controls available and enabled in the device.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.1.1
Test Name: TC_VERIFY_MEMORY_PROTECTION_MECHANISMS

Objective:
Testing, in presence of OEM team, to verify the declared memory protection controls available and enabled in the device using command line-based tools/commands or any other open-source tool like DEP, EMET tool.

Tools used:

Test Execution Steps:

1. In the presence of the OEM team, use the prepared tools to check the status of ASLR and DEP.
2. Execute the appropriate commands to determine if ASLR is active, which typically involves checking if the system is using randomized memory addresses.
3. Check if DEP is enabled by looking for kernel messages or system properties that indicate execution prevention measures are active.
4. Document the tool outputs, system responses, and any OEM inputs.

Expected Results for Pass:

- The tools and commands used confirm the enablement of ASLR by showing that memory addresses are being randomized. The status of DEP is verified to be enabled, indicating that the system is preventing the execution of code from non-executable memory pages.

Test Observations:
Evidence Provided:
Test Case Result:

Overall Test Result

2.2 Test Firmware Data-in-Transit Security

Requirement Description
Verify that the firmware apps protect data in using layer transit transport security.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Specifications and documentation related to the configurations available in the applications and firmware related to transport layer security.

Test Plan
Total number of test cases: 5

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.2.1
TestName: TC_TEST_FIRMWARE_DATA-INTRANSIT_SECURITY_ENCRYPTION

Objective:
Verifying that strong encryption algorithms and secure TLS version is supported by the device to establish secure communication.

Tools used:

Test Execution Steps:

1. Review the specifications and documentation to identify the supported TLS versions and encryption algorithms.
2. Use the testSSL tool to verify the TLS version supported and algorithms used by the DUT.

Expected Results for Pass:
The device establishes connections using only strong, industry-accepted encryption algorithms and the latest secure TLS version.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 2

Test Case: BIS-2.2.2
Test Name: TC_TEST_FIRMWARE_DATA-IN-TRANSIT_SECURITY_SERVER

Objective:
Verifying that device properly validates the server's TLS certificate to ensure that it is trusted and has not been tampered with.

Tools used:

Test Execution Steps:

1. Attempt to establish a TLS connection to the device using a test server with both valid and invalid certificates.
2. Use a network sniffer to observe the device's response to the certificates presented by the server during the handshake process.

Expected Results for Pass:
The device successfully establishes a connection with a server that presents a valid certificate and reject connections where the certificate is invalid or tampered with.

Test Observations:
Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-2.2.3
Test Name: TC_TEST_FIRMWARE_DATA-IN-TRANSIT_SECURITY_VULNE

Objective:
Testing for vulnerabilities which can affect the security of TLS connection such as padding oracle attacks, or weak cipher suites.

Tools used:

Test Execution Steps:
1. Use vulnerability scanning tools like Scantist, Nessus, testssl.sh, SSLyze, or tls-scan, to test the device's TLS implementation for known weaknesses, such as padding oracle attacks or the use of weak cipher suites.
2. Document any vulnerabilities discovered during the scanning process.

Expected Results for Pass:
The scan did not find any critical vulnerabilities in the TLS implementation.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 4

Test Case: BIS-2.2.4
Test Name: TC_TEST_FIRMWARE_DATA-IN-TRANSIT_SECURITY_NMAP

Objective:
Using tools such as Nmap to identify open ports through which device can be accessed leading to unintended data retrieval.

Tools used:

Test Execution Steps:
1. Use Nmap or similar network scanning tools to identify open ports on the device.
2. Analyze the services running on those ports to determine if they are necessary and properly secured.

Expected Results for Pass:
Nmap listed all open ports with details on the services running. The report indicates that the open ports are justified and secure, suggesting closure or additional protection for any unnecessary or insecure services.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 5

Test Case: BIS-2.2.5
Test Name: TC_TEST_FIRMWARE_DATA-IN-TRANSIT_SECURITY_TLS

Objective:
Verifying that the TLS session(s) are resistant to attempts of interception and decryption of network traffic using man-in-the- middle attacks using tools like Burpsuite.

Tools used:

Test Execution Steps:
1. Use tools like Burp Suite or Ettercap to attempt man-in-the-middle (MITM) attacks on the TLS sessions.
2. Attempt to intercept, decrypt, and modify the traffic between the device and the test server.

Expected Results for Pass:
The TLS session resisted MITM attacks, and encrypted traffic remained secure against interception and decryption attempts.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

2.3 Test Server Connection Signature Validation

Requirement Description
Verify that the firmware apps validate the digital signature of server connections.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Document mentioning the use-cases when the device establishes server connections with the external world, with detailed information about the security measures in place while validating the digital signatures of the server connections.

Test Plan
Total number of test cases: 5
Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.3.1
Test Name:
TC_TEST_SERVER_CONNECTION_SIGNATURE_VALIDATION _ENCRYPTION

Objective:
Identifying the scenarios when the device establishes the server connections with the external world and verifying security features, related to secure server connections and digital signature validation as implemented like strong cipher suites, secure TLS version, SSL pinning etc. supported by code walkthrough.

Tools used:

Test Execution Steps:

1. Conduct a detailed review of the source code to check for the implementation of security features such as strong cipher suites and the latest secure TLS version. Look for SSL pinning implementation.

Expected Results for Pass:
Source code includes robust security measures, and that SSL pinning is correctly implemented to prevent man-in-the-middle (MITM) attacks.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 2

Test Case: BIS-2.3.2
Test Name: TC_TEST_SERVER_CONNECTION_SIGNATURE_VALIDATION_ID

Objective:
Identifying the scenarios when the device establishes the server connections with the external world and verifying proper certificate validation, certificate chain validation and certificate revocation checks are implemented in the device.

Tools used:

Test Execution Steps:

1. Set up test servers with various configurations of TLS and invalid/expired certificate and try to establish the connection using the DUT as client.

Expected Results for Pass:
The device establishes connections using secure configurations, DUT rejects the connections from servers with invalid/expired certificates.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 3

Test Case: BIS-2.3.3
Test Name:
TC_TEST_SERVER_CONNECTION_SIGNATURE_VALIDATION_VULNE

Objective:
Testing for vulnerabilities which can affect the security of TLS connection such as padding oracle attacks, or weak cipher suites.

Tools used:

Test Execution Steps:

1. Utilize vulnerability scanning tools specifically tailored for TLS, such as Nessus, testssl.sh, SSLyze, or tls-scan, to test the device's TLS implementation.
2. Evaluate the device for common TLS vulnerabilities and misconfigurations, such as susceptibility to padding oracle attacks and use of weak cipher suites.

Expected Results for Pass:
The testing tools reported that the device's TLS implementation is free of known vulnerabilities and is configured to use strong ciphers, with no critical issues found.

Test Observations:
Evidence Provided:

Test Case Result:

TEST 4

Test Case: BIS-2.3.4
Test Name: TC_TEST_SERVER_CONNECTION_SIGNATURE_VALIDATION _NMAP

Objective:
Using tools such as Nmap to identify open ports through which device can be accessed leading to unintended data retrieval.

Tools used:

Test Execution Steps:

1. Obtain a list of all standard and non-standard ports that the device is expected to use.
2. Use Nmap to perform a port scan targeting the device's IP address to identify all open ports.
3. Analyze the services running on the open ports to assess whether they are intended and properly secured.
4. Document any unexpected open ports or services that do not align with the documented use-cases.

Expected Results for Pass:

- The Nmap scan reported ports that are documented and necessary for the device's operation. There are no unexpected open ports.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 5

Test Case: BIS-2.3.5
Test Name: TC_TEST_SERVER_CONNECTION_SIGNATURE_VALIDATION _TLS

Objective:
Verifying that TLS session(s) are resistant to attempts of interception and decryption of network traffic using man-in-the- middle attacks using tools like Burpsuite.

Tools used:

Test Execution Steps:

1. Use tools like Burpsuite or ettercap to attempt man-in-the-middle (MITM) attacks on the TLS sessions.
2. Attempt to intercept, decrypt, and modify the traffic between the device and the test server.

Expected Results for Pass:
The TLS session is resistant to MITM attacks, and encrypted traffic remained secure against interception and decryption attempts.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

2.4 Check for Safe Alternatives to Banned C Functions

Requirement Description
Verify that any use of banned C functions is replaced with the appropriate safe equivalent functions.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Firmware binaries for code review.
- Internal code review reports.

Test Plan
Total number of test cases: 4

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.4.1
Test Name:
TC_CHECK_FOR_SAFE_ALTERNATIVES_TO_BANNED_C_FUNCTIONS_RECOMMENDED

Objective:
Secure code review [both automated and manual], in presence of OEM team, using a licensed static analysis tool through any of the following approaches in BIS-2.4.1, BIS-2.4.2, BIS-2.4.3 and BIS-2.4.4.
Visit to the evaluation agency by the vendor with the firmware code and install the licensed static analysis tool available with the evaluation agency in their systems.

Tools used:

Test Execution Steps:

1. The vendor visits the evaluation laboratory with the firmware code.
2. Install the licensed static analysis tool provided by the evaluation agency on their system.
3. Perform the code analysis in the presence of the OEM team and evaluation agency representatives, demonstrating the code review activity live.

Expected Results for Pass:
Code review report showing that no banned C functions are present and that safe equivalents are correctly used.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

2.5 Validate Firmware Software Bill of Materials

Requirement Description
Verify that each firmware maintains a software bill of materials (SBOM) cataloging third party components, versioning, and published vulnerabilities.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Documentation for information on software bill of materials, including third-party components and versions.
- Organization process and policies for the following:
  - Addressing and patching any identified vulnerabilities in third-party components.
  - Informing the customers about the security issues or vulnerabilities and providing security updates and patches for the same.
- Configuration management system and related policies for maintaining firmware and third-party binaries, libraries, and frameworks along with the patches/fixes issued to the devices.

Test Plan
Total number of test cases: 3

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.5.1
TestName:
TC_VALIDATE_FIRMWARE_SOFTWARE_BILL_OF_MATERIALS_FACT

Objective:
Verification of the submitted list of third-party components by running automated tools like FACT on the firmware.

Tools used:

Test Execution Steps:

1. Acquire the SBOM from the vendor, which should list all third-party components and their versions.
2. Use an automated tool like FACT (Framework for Analysis of COTS) to scan the firmware and verify that it accurately reflects the SBOM.
3. Ensure that the tool checks for discrepancies between the listed versions and the actual versions in the firmware.

Expected Results for Pass:
The report generated by the automated scan confirms the integrity of the SBOM with no discrepancies found.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 2

Test Case: BIS-2.5.2
Test Name: TC_VALIDATE_FIRMWARE_SOFTWARE_BILL_OF_MATERIALS_ID

Objective:
Identifying vulnerabilities in the third-party component(s) through publicly available vulnerability databases.

Tools used:

Test Execution Steps:

1. Use publicly available vulnerability databases, such as the National Vulnerability Database (NVD) or CVE database, to check for known vulnerabilities in the listed third-party components.
2. Verify vulnerabilities using Nessus.
3. Document any known vulnerabilities along with their severity scores and potential impact.

Expected Results for Pass:
A comprehensive list of all identified vulnerabilities in the third-party components used within the firmware.

Test Observations:

- 15 vulnerabilities were found in the system.
- 1 is critical, 5 are high, 8 are medium and 1 is low severity.
- The vulnerabilities were obtained using the CVE database.

Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-2.5.3
Test Name:
TC_VALIDATE_FIRMWARE_SOFTWARE_BILL_OF_MATERIALS_PROCESS

Objective:
Verification and validation of the process defined by the vendor for providing regular security updates and patches for the firmware to address any known vulnerabilities in third- party components.

Tools used:

Test Execution Steps:

1. Review the organization's processes and policies for addressing and patching vulnerabilities, as provided by the vendor.
2. Validate these processes by checking the history of security updates and patches released for the firmware.
3. Confirm that the vendor has a system to inform customers about security issues and provide timely updates and patches.

Expected Results for Pass:

- The vendor's documented processes for handling vulnerabilities are proven to be effective and consistent with best practices.

Test Observations:

Test Case Result:

Overall Test Result

2.6 Audit Code for Hardcoded Credentials

Requirement Description
Verify all code including third-party binaries, libraries, frameworks are reviewed for hardcoded credentials (backdoors).

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Firmware binaries for code review.
- Internal code review reports.

Test Plan
Total number of test cases: 4

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.6.1
Test Name:
TC_AUDIT_CODE_FOR_HARDCODED_CREDENTIALS_RECOMMENDED

Objective:
Independent secure code review [both automated and manual] using a licensed static analysis tool through any of the following approaches in BIS-2.6.1, BIS-2.6.2, BIS-2.6.3 and BIS-2.6.4.
Visit to the evaluation agency by the vendor with the firmware code and install the licensed static analysis tool available with the evaluation agency in their systems.

Tools used:

Test Execution Steps:

1. The vendor visits the evaluation agency with firmware binaries.
2. Install the agency's licensed static analysis tool on the agency's system.
3. Conduct a thorough automated scan followed by a manual review to identify any hardcoded credentials or backdoors.

Expected Results for Pass:

- No hardcoded credentials, such as usernames or passwords, are found within the codebase in the code review.
- A manual review supports the findings of the automated tools, ensuring no hidden or obfuscated credentials are in the code

Test Observations:

Evidence Provided:

Test Case Result:

2.7a Test Firmware Digital Signature Pinning

Requirement Description
Verify that the firmware apps pin the digital signature to a trusted server(s).

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Document mentioning the use-cases when the device establishes server connections with the external world, with detailed information about the security measures in place while validating the digital signatures of the server connections.

Test Plan
Total number of test cases: 2

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.7a.1
Test Name: TC_TEST_FIRMWARE_DIGITAL_SIGNATURE_PINNING_TLS

Objective:
Identifying the scenarios when the device establishes the server connections with the external world and verifying security features, related to secure server connections and digital signature validation as implemented like strong cipher suites, secure TLS version, SSL pinning etc. supported by code walkthrough.

Tools used:

Test Execution Steps:

1. Obtain the use-case documentation detailing how the device establishes secure server connections.
2. Review the documentation to understand the security features implemented for secure server connections, including digital signature validation, strong cipher suites, secure TLS versions, and SSL pinning.
3. Perform a code walkthrough to ensure these security features are correctly implemented within the firmware.
4. Simulate server connections in a controlled environment to observe and verify the implementation of these security features. Use Wireshark to observe the behavior.

Expected Results for Pass:
The code walkthrough and simulated connections confirm that the device uses strong cipher suites, a secure TLS version, and properly implements SSL pinning as outlined in the documentation.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 2

Test Case: BIS-2.7a.2
Test Name: TC_TEST_FIRMWARE_DIGITAL_SIGNATURE_PINNING_CERT

Objective:
Identifying the scenarios when the device establishes the server connections with the external world and verifying proper certificate validation, certificate chain validation and certificate revocation checks are implemented in the device.

Tools used:

Test Execution Steps:

1. Verify the device's process for validating server certificates, including the certificate chain and revocation status.
2. Test the device's reaction to various certificates by simulating secure connections using certificates that are valid, expired, revoked, and from untrusted authorities.
3. Confirm that the device checks the entire certificate chain, up to the root certificate, and properly handles certificate revocation statuses.

Expected Results for Pass:
The device uses strong protocols and ciphers to establish the sessions.
The device correctly validates server certificates and rejects connections from servers with expired, revoked, or untrusted certificates.

Test Observations:
Evidence Provided:

Test Case Result:
Overall Test Result

2.7b Assess Firmware Reverse Engineering Protections

Requirement Description
Verify security controls are in place to hinder firmware reverse engineering (e.g., removal of verbose debugging symbols).

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Documentation regarding the security controls in place to hinder firmware reverse engineering.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.7b.1

TestName:
TC_ASSESS_FIRMWARE_REVERSE_ENGINEERING_PROTECTIONS

Objective:
Testing, in presence of OEM team, to verify the security controls as provided by the vendor to hinder firmware reverse engineering.

Tools used:

Test Execution Steps:

1. Arrange a session with the OEM team to oversee and validate the testing process.
2. Use open-source tools like Binwalk or Ghidra to analyze the firmware binary for signs of security controls such as stripped debugging symbols, obfuscation, or encryption.
3. Binwalk can be used to scan the firmware for embedded files and code, whereas Ghidra offers functionality for disassembly that could reveal whether verbose debugging symbols have been removed.
4. Perform a thorough examination to check for anti-reversing measures, such as control flow obfuscation or the presence of anti-tamper checks.

Expected Results for Pass:

- The analysis tools did not find verbose debugging symbols or other informational artifacts that could aid reverse engineering.
- The firmware has demonstrated evidence of security measures like obfuscation and encryption that would deter reverse engineering.

Test Observations:

Evidence Provided:
Test Case Result:
Overall Test Result

2.8 Evaluate Firmware Update Security

Requirement Description
Verify that the firmware update process is not vulnerable to time-of-check vs time-of-use attacks (TOCTOU).

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Measures implemented in the device to make it resistant to time-of-check vs. time-of-use attacks.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.8.1
Test Name: TC_EVALUATE_FIRMWARE_UPDATE_SECURITY

Objective:
Testing, in presence of OEM team, to verify the measures implemented in the device to make it resistant to time-of-check vs. time-of-use (TOCTOU) attacks.

Tools used:

Test Execution Steps:

1. Replicate the update process in a way that allows you to introduce changes between the check and use phases. This could involve simulating an update being authenticated but modified just before deployment.
2. Monitor the update mechanism to see if it detects and prevents an attack where the update package is altered after passing an authenticity check but before being applied.
3. Employ techniques to attempt TOCTOU attacks, such as intercepting the firmware update transmission and trying to inject or modify the update after its integrity check has been completed but before it is executed.

Expected Results for Pass:

- The device demonstrates that it has appropriate measures in place, such as real-time integrity checks, to thwart TOCTOU attacks.
- The device detects and blocks the attempt of TOCTOU attack.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

2.9 Confirm Device Code Signing and Validation

Requirement Description
Verify the device uses code signing and validates firmware upgrade files before installing.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Process of achieving secure firmware upgrade which should consist of keys involved and their management life cycle*, signature validation process and any other secure mechanisms if implemented.

Test Plan
Total number of test cases: 2

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.9.1
Test Name: TC_CONFIRM_DEVICE_CODE_SIGNING_AND_VALIDATION_POS

Objective:
Testing, in presence of OEM team, to verify that the device gets successfully updated with the documented secure upgrade process when a valid update package is provided.

Tools used:

Test Execution Steps:

1. Review the documentation outlining the secure firmware upgrade process, including the role of code signing and key management.
2. Prepare an environment to simulate the firmware update in the presence of the OEM team.
3. Provide a valid update package with a legitimate signature for the upgrade.
4. Proceed with the upgrade process and observe the device behavior and validation logs to ensure the update is applied only after successful signature verification.

Expected Results for Pass:

- The device successfully updates with the validly signed package, and the integrity of the code signing process is upheld.
- Logs and other system outputs confirm that signature validation was performed and passed.

Test Observations:

Evidence Provided:
Test Case Result:

TEST 2

Test Case: BIS-2.9.2
Test Name: TC_CONFIRM_DEVICE_CODE_SIGNING_AND_VALIDATION_NEG

Objective:
Testing, in presence of OEM team, to verify that the device does not boot up when a tampered update package (like with missing signature, invalid signature) is provided.

Tools used:

Test Execution Steps:

1. In the presence of the OEM team, attempt to upgrade the device with a tampered update package (one with a missing or invalid signature).
2. Monitor the device's response to the tampered package, looking for rejection based on signature validation failure.
3. Document the process and outcome, noting any system messages or logs that indicate the invalidity of the update package.

Expected Results for Pass:

- The device does not apply the update, recognizing the missing or invalid signature and thereby preventing a potentially unauthorized firmware change.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

2.10 Test Anti-Rollback Firmware Protection

Requirement Description
Verify that the device cannot be downgraded to old versions (anti-rollback) of valid firmware.

DUT Confirmation Details

DUT Software Details:

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Process of achieving secure firmware upgrade which should consist of keys involved and their management life cycle*, signature validation process and any other secure mechanisms if implemented.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.10.1
Test Name: TC_TEST_ANTI-ROLLBACK_FIRMWARE_PROTECTION

Objective:
Testing, in presence of OEM team, to verify that the device cannot be downgraded to old versions (anti-rollback) of valid firmware.

Tools used:

Test Execution Steps:

1. In the presence of the OEM team, attempt to install an older version of the firmware on the device.
2. Monitor and document the device's response, noting any rejection messages or behaviors that indicate an anti-rollback mechanism is in place.
3. If the device has version tracking (like a secure monotonic counter), verify that it correctly identifies, and blocks attempts to install outdated firmware.
4. Check system logs to confirm that the downgrade attempt was logged as an unauthorized action.

Expected Results for Pass:

- The device does not provide the option to roll back to the older firmware version, regardless of its signature's validity, due to the anti-rollback measures.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

2.11 Verify Scheduled Firmware Update Capability

Requirement Description
Verify that firmware can perform automatic firmware updates upon a predefined schedule.

DUT Confirmation Details
DUT Software Details:

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Modes of updates available i.e., automatic, manual or both.
- Organizational process and policies regarding the issuing of updates to the devices.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-2.11.1
Test Name: TC_VERIFY_SCHEDULED_FIRMWARE_UPDATE_CAPABILITY
Objective:
Verification shall be done as per the applicable scenario:

- **CASE 1:** Automatic OTA updates are available: A standard operating procedure for issuing automatic updates/upgrades to the in-field devices is required to be submitted by the vendor which can then be evaluated by the evaluation agency as per C20, C21 and C22 security requirement of OWASP open standard.
- **CASE 2:** Automatic OTA updates are not available, and vendor provides manual updates: A standard operating procedure for issuing manual updates/upgrades to the in-field devices is required to be submitted by the vendor which can then be evaluated by the evaluation agency as per C20, C21 and C22 security requirement of OWASP open standard.
- The security requirements from OWASP are:
    - C.20 Verify that the firmware update process is not vulnerable to time-of-check vs time-of-use attacks.
    - C.21 Verify the device uses code signing and validates firmware upgrade files before installing.
    - C.22 Verify that the device cannot be downgraded to old versions (anti-rollback) of valid firmware

Tools used:

Test Execution Steps:

- **CASE 1: Automatic OTA Updates Available**
    1. Review the provided standard operating procedure (SOP) from the vendor for issuing automatic updates to in-field devices.
    2. Validate the modes of updates as per documentation, ensuring both automatic and manual methods are covered if available.

3. Conduct a test to simulate an automatic OTA update, preferably in a controlled environment. This test should follow the documented SOP and use the firmware update mechanisms as they would operate in a real-world scenario.
4. Check the system logs and other relevant outputs to confirm the successful deployment of the update as per the predefined schedule.

- **CASE 2: Automatic OTA Updates Not Available, Manual Updates Provided**
  1. Obtain the SOP from the vendor regarding the manual update process.
  2. Evaluate the SOP to ensure it meets the security requirements outlined by OWASP, paying particular attention to the integrity of the update package and the authentication of the update source.
  3. Manually initiate an update process following the SOP and monitor the process to confirm that all security steps are observed, and that the device does not execute an update without proper validation.

Expected Results for Pass:

- **CASE 1: Automatic OTA Updates Available**
  The device successfully completes an automatic OTA update following the predefined schedule and in accordance with the SOP. The OTA updates are meeting the OWASP C20, C21, and C22 security requirements.
- **CASE 2: Automatic OTA Updates Not Available, Manual Updates Provided**
  The device updates only when the manual process is followed correctly, and all security measures are verified as per the SOP.

Test Observations:

Evidence Provided:
Test Case Result:

<u>Overall Test Result</u>

3. Secure Process Conformance

3.1 Verify Mutual Authentication of Wireless Communications

Requirement Description
Verify that wireless communications are mutually authenticated.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- The documentation regarding the process of mutual authentication as implemented in the device when wireless communications are initiated.
- In case the device does not support wireless communications, the vendor shall provide a declaration for the same.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-3.1.1
Test Name:
TC_VERIFY_MUTUAL_AUTHENTICATION_OF_WIRELESS_COMMUNICATIONS

Objective:
Testing, in presence of OEM team, to verify the process of mutual authentication as laid down in the documentation by the vendor.

Tools used:

Test Execution Steps:

1. In collaboration with the OEM team, set up a test environment that allows for the initiation of wireless communications with the device.
2. Follow the documented mutual authentication process step by step, which could involve the use of certificates, pre-shared keys, or other cryptographic methods that ensure both the device and the communication partner authenticate each other before establishing a connection.
3. Attempt to establish wireless communication with and without proper authentication credentials to test the robustness of the authentication mechanism.
4. Monitor the exchange of authentication messages and verify that the communication is only successful when proper authentication takes place from both ends.

Expected Results for Pass:

- The device establishes the wireless connection when the mutual authentication process is correctly followed.
- Attempts to communicate without proper mutual authentication are failed, confirming the device's compliance with the documented security procedures.

Test Observations:
Evidence Provided:
Test Case Result:

Overall Test Result

3.2 Test Encryption of Wireless Communication Channels

Requirement Description
Verify that wireless communications are sent over an encrypted channel.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Documentation regarding the security measures implemented in the device to prevent tampering of the data being sent through wireless mode of communication.
- In case the device does not support wireless communications, the vendor shall provide a declaration for the same.

Test Plan
Total number of test cases: 3

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-3.2.1
Test Name:
TC_TEST_ENCRYPTION_OF_WIRELESS_COMMUNICATION_CHANNELS_TEST

Objective:
Testing, in presence of OEM team, to verify the process of mutual authentication as laid down in the documentation by the vendor.

Tools used:

Test Execution Steps:

1. In collaboration with the OEM team, set up a test environment that allows for the initiation of wireless communications with the device.
2. Follow the documented mutual authentication process step by step, which could involve the use of certificates, pre-shared keys, or other cryptographic methods that ensure both the device and the communication partner authenticate each other before establishing a connection.
3. Attempt to establish wireless communication with and without proper authentication credentials to test the robustness of the authentication mechanism.
4. Monitor the exchange of authentication messages and verify that the communication is only successful when proper authentication takes place from both ends.

Expected Results for Pass:

- All the wireless communications happen over encrypted channels.
- The attempts to capture and read the communication failed.

Test Observations:
Evidence Provided:
Test Case Result:

TEST 2

Test Case: BIS-3.2.2
Test Name:
TC_TEST_ENCRYPTION_OF_WIRELESS_COMMUNICATION_CHANNELS_CODE

Objective:
Identifying all the security mechanisms being used in the communication process verification through code review.

Tools used:

Test Execution Steps:
1. Perform a thorough code review of the device's firmware, focusing specifically on the implementation of wireless communication features.
2. Look for the use of encryption libraries and functions within the code.
3. Verify that the code properly implements best practices for secure encryption, such as the use of strong cryptographic protocols (e.g., WPA2, WPA3, or TLS for wireless communication).

Expected Results for Pass:
- The code review confirms that the device uses strong, up-to-date cryptographic protocols for encrypting wireless communications.

Test Observations:

Evidence Provided:

Test Case Result:

TEST 3

Test Case: BIS-3.2.3
Test Name:
TC_TEST_ENCRYPTION_OF_WIRELESS_COMMUNICATION_CHANNELS_PROCESS

Objective:
Identifying all the security mechanisms being used in the communication process verification through process audit of the key-life cycle process.

Tools used:

Test Execution Steps:

1. Review the process documentation provided by the vendor related to key management, including generation, distribution, storage, rotation, and revocation of encryption keys.
2. Audit the actual key management practices in place to verify they conform to the documented procedures.
3. Ensure that keys are handled securely throughout their lifecycle to maintain the integrity and confidentiality of the encryption process.

Expected Results for Pass:

- The key management practices for wireless communications are secure and follow the documented procedures without deviation.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

3.3 Assess Trusted Supply Chain for Critical Components

Requirement Description

Verify that whether trusted sources are being used for sourcing the components of the device i.e., trusted supply chain through a managed Bill of materials for critical hardware components (related to security functions like SoC) is in use.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions

The vendor shall provide the following:

- Bill of materials for critical hardware components (related to security functions like SoC).

Test Plan

Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-3.3.1
Test Name:
TC_ASSESS_TRUSTED_SUPPLY_CHAIN_FOR_CRITICAL_COMPONENTS

Objective:
Verify that whether trusted sources are being used for sourcing the components of the device i.e., trusted supply chain through a managed Bill of materials for critical hardware components (related to security functions like SoC) is in use.

Tools used:
Test Execution Steps:
Only validate the documentation.

Expected Results for Pass:

- Documentation confirms that all critical hardware components are sourced from trusted suppliers.
- The Bill of Materials (BOM) reflects a managed and secure supply chain process.
- The hardware components in use are authentic and procured through reputable and secure channels.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

3.4 Evaluate Supply Chain Risk Management Process

Requirement Description
Supply chain risk identification, assessment, prioritization, and mitigation shall be conducted. Supply chain risk/business continuity planning policy documents, playbooks reflecting how to handle supply chain disruption, post-incident summary documents need to be submitted and demonstrate the same.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image
DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Supply chain risk identification, assessment, prioritization, and mitigation documents.
- Supply chain risk/business continuity planning policy documents, playbooks reflecting how to handle supply chain disruption, post-incident summary documents.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-3.4.1
Test Name:
TC_EVALUATE_SUPPLY_CHAIN_RISK_MANAGEMENT_PROCESS

Objective:
Supply chain risk identification, assessment, prioritization, and mitigation shall be conducted. Supply chain risk/business continuity planning policy documents, playbooks reflecting how to handle supply chain disruption, post-incident summary documents need to be submitted and demonstrate the same.

Tools used:

Test Execution Steps:

Expected Results for Pass:

- Comprehensive documentation is available, showing a robust supply chain risk identification, assessment, prioritization, and mitigation process.
- Policy documents and playbooks clearly outline procedures for addressing and recovering from supply chain disruptions.
- Post-incident reports or summaries (if any) demonstrate the effectiveness of these policies in practical scenarios, illustrating the organization's readiness and resilience against supply chain threats.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

3.5 Confirm Use of Non-Proprietary Network Protocols

Requirement Description
Verify that no proprietary network protocols are being used in the device. If yes, then complete implementation details and the source code for the same shall be provided.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Document for Network protocols used in the device.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-3.5.1
Test Name:
TC_CONFIRM_USE_OF_NON-PROPRIETARY_NETWORK_PROTOCOLS

Objective:
Supply chain risk identification, assessment, prioritization, and mitigation shall be conducted. Supply chain risk/business continuity planning policy documents, playbooks reflecting how to handle supply chain disruption, post-incident summary documents need to be submitted and demonstrate the same.

Tools used:

Test Execution Steps:
Only validate the documentation. You could use Wireshark to validate the existence of proprietary network protocols.

Expected Results for Pass:

- Documentation regarding all the network protocols used in the device is verified.
- If proprietary protocols are in use, complete documentation, including implementation details and source code, is provided, demonstrating that these protocols have been developed with security considerations and have been thoroughly tested.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

4. Security Conformance at Product Development Stage

4.1 Assess Design and Architecture for Counterfeit and Malware Risks

Requirement Description
Design and architecture details till the PCBA and SoC level to be provided to aid in counterfeit mitigation and malware detection.

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Design and architecture documents till the PCBA and SoC level.

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-4.1.1
Test Name:
TC_ASSESS_DESIGN_AND_ARCHITECTURE_FOR_COUNTERFEIT_AND_MALWARE_RISKS

Objective:
Design and architecture details till the PCBA and SoC level to be provided to aid in counterfeit mitigation and malware detection.

Tools used:

Test Execution Steps:
Only validate the documentation.

Expected Results for Pass:

- Detailed design and architecture documentation that covers the PCB and SoC levels are provided.
- Documentation includes measures and controls implemented to prevent the use of counterfeit components and to detect malware.

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

4.2 Test Threat Mitigation Strategies for Tainted and Counterfeit Products

Requirement Description
Threat mitigation strategies for tainted and counterfeit products shall be implemented as part of product development.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
None

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-4.2.1
Test Name:
TC_TEST_THREAT_MITIGATION_STRATEGIES_FOR_TAINTED_AND_COUNTERFEIT_PRODUCTS

Objective:
Process and method artifacts need to be submitted and demonstrate the same.

Tools used:

Test Execution Steps:

1. Request and review all process and method artifacts related to threat mitigation strategies from the vendor.
2. The documentation should detail the approaches taken during the product development lifecycle to identify, assess, and mitigate the risks associated with tainted and counterfeit products.
3. Request a demonstration from the vendor showing the actual implementation of these strategies during the product development phase. This could involve walk-throughs of systems or processes put in place to prevent the integration of tainted or counterfeit components.
4. Verify that there are systems in place for continuous monitoring and periodic auditing of supply chains and ensure there are response protocols for when tainted or counterfeit components are detected.

Expected Results for Pass:

- The review and demonstration confirm that comprehensive mitigation strategies are part of the product development process.
- Vendor documentation and demonstrations show clear, actionable procedures that are regularly followed to prevent the inclusion of tainted or counterfeit products.

Test Observations:

Evidence Provided:

Test Case Result:
Overall Test Result

4.3 Verify Deployment of Malware Detection Tools in Development

Requirement Description
One or more up-to-date malware detection tools shall be deployed as part of the code acceptance and development processes. Malware detection techniques shall be used before final packaging and delivery (e.g., scanning finished products and components for malware using one or more up-to-date malware detection tools).

DUT Confirmation Details
DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
None

Test Plan
Total number of test cases: 1

Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-4.3.1
Test Name:
TC_VERIFY_DEPLOYMENT_OF_MALWARE_DETECTION_TOOLS_IN_DEVELOPMENT

Objective:
List of components identified as requiring tracking targets of tainting/counterfeiting, CM tool. Quality assurance process needs to be submitted and demonstrate the same.

Tools used:

Test Execution Steps:

1. Obtain a list of components identified as targets for tainting or counterfeiting and which require tracking.
2. Review the Configuration Management (CM) tool and processes that are in place for version control and change management of these components.
3. Examine the quality assurance process documents that detail the integration of malware detection tools into the product lifecycle.
4. Verify the actual deployment of malware detection tools by observing a demonstration of the tools in action, scanning finished products and components.

Expected Results for Pass:

- The documentation and demonstration show that up-to-date malware detection tools are being used at key stages of product development, particularly before code acceptance, and prior to final packaging and delivery.
- The CM tool has a clear audit trail for changes in components, particularly for those identified as high-risk for tampering or counterfeiting.
- The quality assurance process documents show a robust procedure for integrating malware detection into the development and delivery process,

Test Observations:

Evidence Provided:

Test Case Result:

Overall Test Result

4.4 Evaluate Supply Chain Risk Management Practices

Requirement Description
Supply chain risk identification, assessment, prioritization, and mitigation shall be conducted.

DUT Confirmation Details

DUT Software Details

Hash Checksum Verification for DUT's Software Image

DUT Configuration

Pre-Conditions
The vendor shall provide the following:

- Supply chain risk/business continuity planning policy documents, playbooks reflecting how to handle supply chain disruption, post-incident summary documents need to be submitted and demonstrate the same.

Test Plan
Total number of test cases: 1
Test-bed Diagram with Interfaces and IP's:

TEST 1

Test Case: BIS-4.4.1
Test Name:
TC_EVALUATE_SUPPLY_CHAIN_RISK_MANAGEMENT_PRACTICES

Objective:
Supply chain risk identification, assessment, prioritization, and mitigation shall be conducted.

Tools used:

Test Execution Steps:
Only validate the documentation.

Expected Results for Pass:

- Comprehensive supply chain risk management documentation is available and verified, detailing the policies and procedures for identifying, assessing, prioritizing, and mitigating risks.

Test Observations:

Evidence Provided:
Test Case Result:

Overall Test Result

ANNEXURE - A

Product Images (Model No.:)

FRONT
BACK
INSIDE
BOTTOM
LABEL:

---END OF REPORT---