# Google Tensor UFS Inline Storage Encryption Cryptographic Module

Software Version: 1.0

Hardware Version: de8b6c8621

FIPS 140-2 Non-Proprietary Security Policy

Google, LLC.

Documentation Version: 0.4

Date: October 24, 2022

# Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for the Google Tensor UFS Inline Storage Encryption cryptographic module. It contains a specification of the rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 Software-Hybrid cryptographic module. This security policy is for the validation of the Google Inline Storage Encryption cryptographic module.

In this document, the terms “Google Tensor UFS Inline Storage Encryption”, “ISE”, “cryptographic module” or “module” are used interchangeably to refer to the Google Tensor UFS Inline Storage Encryption Cryptographic Module with software version 1.0 and the hardware version de8b6c8621.

# Cryptographic Module Specification

## Description of Module

The module is a multi-chip standalone software-hybrid module designed for on-the-fly hardware encryption for a flash storage device. The logical cryptographic boundary for the module includes the following components:

- Tensor UFS ISE Driver CMVP Test Module
- Tensor UFS ISE

The Tensor UFS ISE Driver CMVP Test Module (hereafter referred to as ISE Driver) is the software component of the cryptographic module running in the Linux Kernel which calls the cryptographic algorithms implemented in the module's hardware component for power-up self-tests, and also sets the FIPS status of the entire cryptographic module once the power up self-test is completed successfully.

The Tensor UFS ISE is the hardware component of the cryptographic module which supports AES-XTS encryption and decryption. This hardware resides in the Google Tensor processor, located between external DRAM and the flash storage device so it can provide inline encryption/decryption while maintaining device performance (such as responsiveness and power consumption).

The hybrid cryptographic module is specified in the following table.

| Component | Type | Version | Location |
|---|---|---|---|
| Tensor UFS ISE Driver CMVP Test Module | Software | 1.0 | boot.img |
| Tensor UFS ISE | Hardware | de8b6c8621 | Tensor processor |

*Table 1 - Components of the Tensor UFS ISE Hybrid Module*

The following table shows the overview of the security level for each of the eleven sections of the validation.

| Security Component | Security Level |
|---|---|
| Cryptographic Module Specification | **1** |
| Cryptographic Module Ports and Interfaces | **1** |
| Roles, Services and Authentication | **1** |
| Finite State Model | **1** |
| Physical Security | **1** |

| | |
|---|---|
| Operational Environment | **1** |
| Cryptographic Key Management | **1** |
| EMI/EMC | **3** |
| Self Tests | **1** |
| Design Assurance | **3** |
| Mitigation of Other Attacks | **N/A** |
| Overall Level | **1** |

*Table 2 - Security Levels*

## Description of Operational State

The module supports only FIPS mode. When the module is initialized during the boot process (as part of the kernel start-up), the power-up self-tests are automatically executed without any operator intervention. If the self-tests complete successfully, the module enters its operational state.

Any failure in the outcome of the self-test shall result in a system panic. With no further operations permitted to the cryptographic services and requiring the system to restart. If the self-tests pass, all cryptographic services are available and the device will start/resume normally.

## Cryptographic Module Boundary

The physical boundary of the module is the physical boundary of the device that contains the module. Consequently, the embodiment of the module is a multi-chip standalone cryptographic module.

### Block Diagram

In the following diagram, the bidirectional arrows depict the flow of the status, control and data. The operations within the logical cryptographic boundary (the blue dotted region) use the cipher from the Tensor UFS ISE (hardware) that is included in the logical boundary. The CMVP Test Module (ISE Driver) is only used during initialization of the module as a whole (to perform all self-tests).

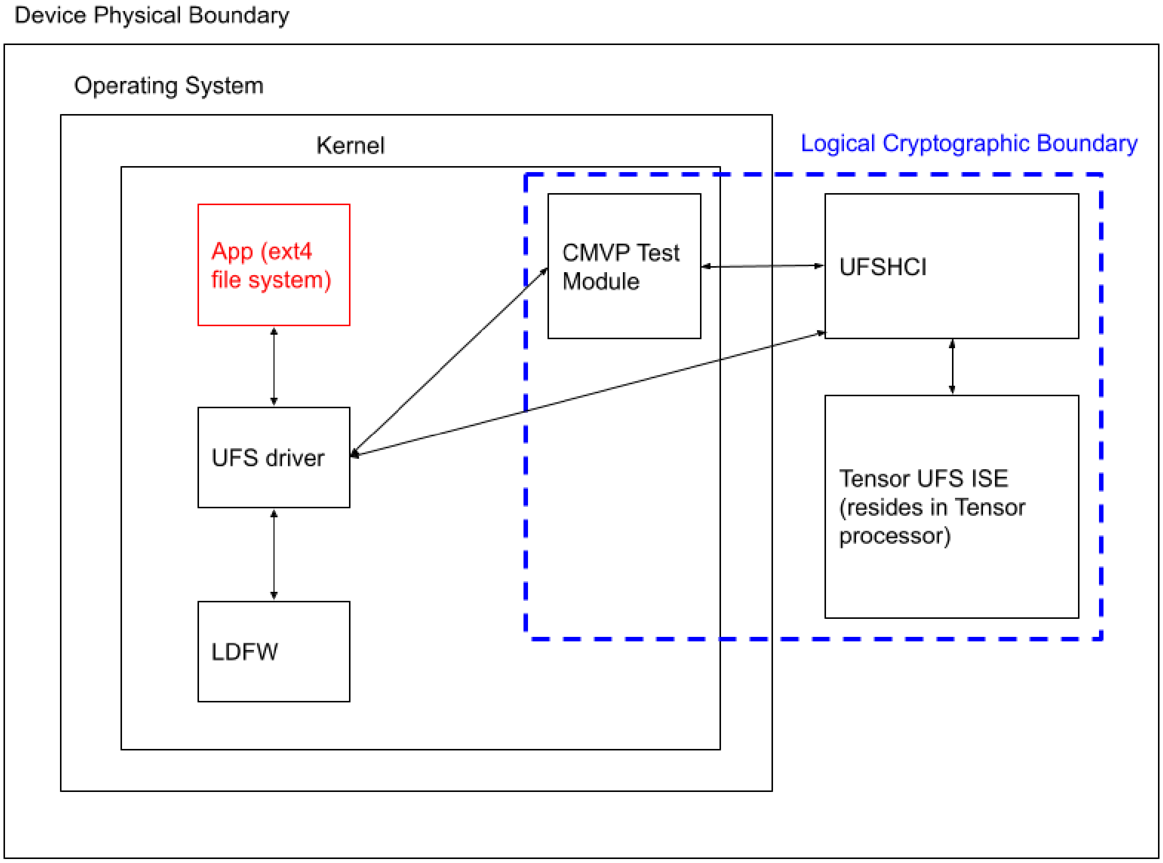

*Figure 1 - Cryptographic Boundary*

## Hardware Block Diagram

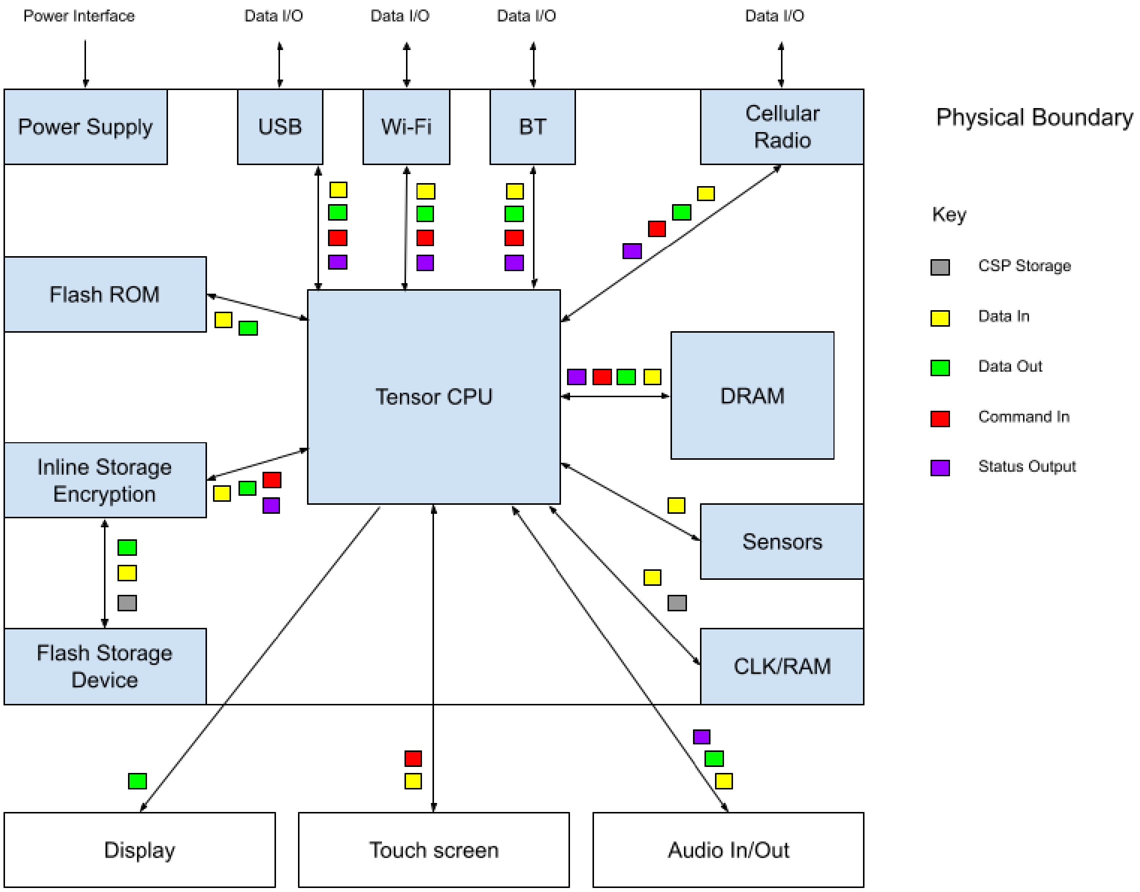

*Figure 2 - Device Physical Boundary*

In the following diagram, the bidirectional arrows depict the flow of the status, control and data. The CSPs and settings are passed via memory and processed by the Tensor UFS ISE hardware component.

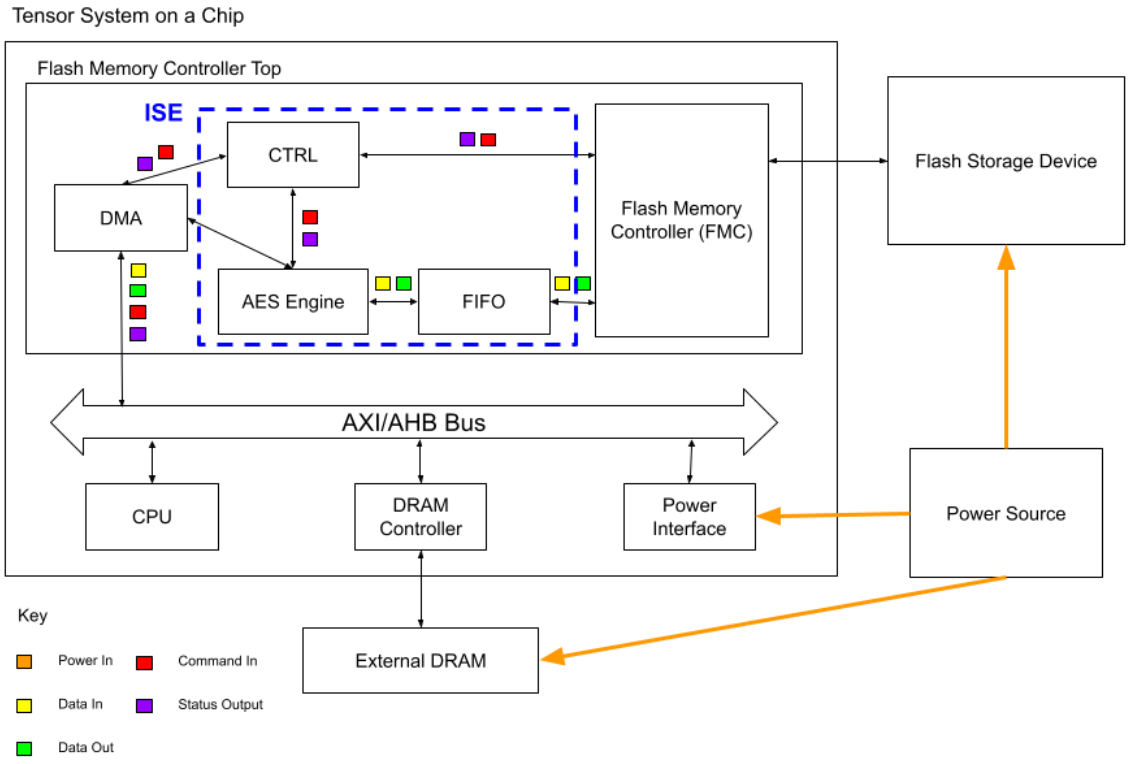

*Figure 3 - Hardware Block Diagram of Tensor UFS ISE*

# Cryptographic Module Ports and Interfaces

| FIPS Interface | API Interface | Physical Interface |
|---|---|---|
| Data Input | API input parameters | FIFO Interface |
| Data Output | API output parameters | FIFO Interface |
| Control Input | API function calls | CTRL Interface |
| Status Output | API return codes, kernel log messages | CTRL Interface |
| Power Input | Physical power connector | CPU power pins |

*Table 3 - Ports and Interfaces*

# Roles, Services and Authentication

## Roles

| Role | Description |
|---|---|
| User | Perform general security services, including cryptographic operations and other Approved security functions. |
| Crypto Officer (CO) | Perform module initialization. |

*Table 4 - Roles*

The module meets all FIPS 140-2 Level 1 requirements for Roles and Services, implementing both User and Crypto Officer roles. The module does not allow concurrent operators.

The User and Crypto Officer roles are implicitly assumed by the entity accessing services implemented by the module. No further authentication is required. The Crypto Officer can initialize the module.

## Services

The module does not support a bypass capability to the security functionality. When write requests are made to the ISE, all plaintext entered in the ISE buffers will be encrypted. For read requests, the ISE will place the decrypted plaintext into the specified buffers.

The following table describes the services available:

| Service | Roles | | Keys/CSPs | Access (Read/Write/ Execute) |
|---|---|---|---|---|
| | User | CO | | |
| **HARDWARE** | | | | |
| AES-XTS encryption and decryption for File Protection | X | | AES-XTS key for file protection | R, X |
| **SOFTWARE** | | | | |
| Self-Test<br>(Self-test is executed automatically when device is booted or restarted) | X | | HMAC keys for Integrity Tests, AES keys for Known Answer Tests | R, X |
| Check Status/Get State | X | | N/A | X |
| Zeroization | X | | DMA descriptor pointer to AES-XTS key for Encryption | W |
| Module Initialization | | X | N/A | N/A |

*Table 5 - Approved Services*

## Operator Authentication

There is no operator authentication; assumption of role is implicit by action.

## Mechanism and Strength of Authentication

No authentication is required at security level 1; authentication is implicit by assumption of the role.

# Physical Security

The Tensor UFS ISE is a software-hybrid module that operates on a multi-chip standalone platform, which conforms to the Level 1 requirements for physical security. The hardware portion of the cryptographic module is a production grade component. The cryptographic module must be used in a commercial off-the-shelf (COTS) mobile device. The mobile device shall comprise production grade components with standard passivation (a sealing coat applied over the chip circuitry to protect it against environmental and other physical damage) and a production grade enclosure that completely surrounds the cryptographic module.

# Operational Environment

The module operates in a modifiable operational environment per FIPS 140-2 Security Level 1 Specifications. The operating system shall be restricted to a single operator mode of operation. The procurement, build and configuring procedure are controlled. The module is installed into a commercial off-the-shelf (COTS) mobile device. The external application that makes calls to the cryptographic module is the single user of the cryptographic module, even when the application is serving multiple clients.

The module has been tested on the following platforms:

| Device | Processor | Operating Environment |
|---|---|---|
| Google Pixel 6 | Google Tensor with PAA | Linux Kernel 5.10 |

*Table 6 - Tested Platform*

# Cryptographic Algorithms

## Approved Cryptographic Algorithms

The module implements the following CAVP validated algorithms:

| Algorithm | Modes | Description | FIPS Approved Cert # |
|---|---|---|---|
| AES | XTS | Hardware Encryption/Decryption of data by Tensor UFS ISE (hardware); 256-bit keys | A1981 |
| HMAC | HMAC-SHA-256 | Software integrity test for ISE Driver; 344 bits | A2219 |
| SHS | SHA-256 | Pre-requisite algorithm of HMAC-SHA-256 used for software integrity test for ISE Driver | A2219 |

*Table 7 - CAVP Validated Algorithms*

**Notes:**

- According to SP 800-38E, the AES algorithm in XTS mode can be only used for the cryptographic protection of data on storage devices, as specified in SP800-38E. In addition, the length of a single data unit encrypted with the XTS-AES shall not exceed $2^{20}$ AES blocks, that is, 16 MiB of data. In addition, to meet the requirement in A.9, the module implements a check to ensure that the two AES keys (Key_1 and Key_2) used in XTS-AES algorithm are not identical.

- The SHA and HMAC implementations in the module are only used for the Software Integrity Test of the module during power-on.

The module does not have any non-approved or non-allowed algorithms.

# Cryptographic Key Management

## Keys and CSPs List

The CSPs are provided in (written to) a DMA (Direct Memory Access) descriptor that is setup by the third party application. The Flash Memory Controller (FMC) reads the information in the DMA descriptor and passes it to the ISE module. The DMA descriptor includes the following information:

- Address for the data blocks (the data to be encrypted)
- Data length
- Key length
- Mode
- Algorithm
- File Encryption IV
- File Encryption Key & File Tweak Key.

The following table lists the Keys and CSPs in the module:

| CSP/Key | Size | Entry/ Output | Storage | Zeroization |
|---|---|---|---|---|
| AES-XTS key for file protection | 256 bits | CSPs are read in the module from the DMA descriptor stored in the memory (figure 3), via the FMC, at the hardware level. | Third party applications store the keys in the DMA descriptor in plaintext. | Key is zeroized when the host device powers off. |
| **NOTE:** The following keys are not a CSP according to IG 7.4 as they are only used for the integrity and known answer tests during the module power-up sequence. | | | | |
| HMAC key for Integrity Test | 344 bits | N/A | Stored as plaintext within the ISE Driver binary. | These keys are not subject to key zeroization according to IG 7.4. |
| AES keys for Known-Answer Test | 256 bits | N/A | Stored as plaintext within the ISE Driver binary. | |

*Table 8 - Keys and CSPs*

## Key Generation

The module does not provide any key generation service or perform any key generation for any of its Approved algorithms. Keys are instead provided by third party applications and stored in a DMA descriptor located in memory. The module does not support any key establishment methods or asymmetric algorithms and hence no key generation services for them.

## Key Entry and Output

The module does not support manual key entry or key output. CSPs can only be exchanged in memory via a DMA descriptor inside the physical boundary of the device and may therefore be passed to the module as plaintext.

## Key Storage

All CSPs within the module are stored in a DMA descriptor located in memory. It is the user’s responsibility to destroy sensitive information in the DMA descriptor using procedures compliant to FIPS 140-2.

## Key Zeroization

As the CSPs are stored in a DMA descriptor located in the memory, it is the user’s responsibility to destroy the sensitive information in the DMA descriptor using FIPS Pub 140-2 compliant procedures. The cryptographic module itself does not destroy externally stored keys and secrets since it does not own these CSPs. The zeroization API function is provided by the module to clear out the sensitive information stored in the module with 0s. No CSP is passed in the module’s Driver.

# Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

The module is a software-hybrid module that has been tested on the test platforms listed in Table 2 - Tested Platforms. The Tensor UFS ISE hardware component cannot be certified by the FCC as it is not a standalone device. The test platforms are accepted by the FCC with the following information:

**Lab name:** Sporton International Inc. EMC & Wireless Communications Laboratory

**FCC ID:** A4RGLU0G (EMI/EMC testing done as declaration of conformity)

The test platforms conform to the EMI/EMC requirements specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital Devices, Class B (i.e., for home use).

# Self-Tests

The module performs a series of Power On Self-Tests (POSTs) that covers all of its FIPS-approved algorithms. The module automatically executes all self-tests when the module is initialized during the boot process without any operator intervention. If any of the self-tests fail, the module will output the specific error message in the kernel log, return the error codes to the calling application and trigger a kernel panic, halting the system. To recover from the error state, reinitialization is possible by doing a reboot to set the module to the power-on state.

If the power-up self-tests are completed successfully, the system may access the ISE cryptographic operations.

FIPS 140-2 explicitly allows that the on-demand test can be fulfilled with a power cycle of the module. Hence, a power cycle and its associated power-on self-test is the methodology used to perform the “on-demand” tests.

## Power On Self-Tests

The module implements each of the following Power On Self-Tests:

- AES-XTS encryption Known Answer Tests (KAT)
- AES-XTS decryption Known Answer Tests (KAT)
- SHA-256 KAT
- HMAC-SHA-256 KAT
- Integrity Test (HMAC-SHA256)

NOTE: The SHA-256 KAT is conducted in conjunction with the HMAC-SHA-256 KAT.

## Integrity Test

The Tensor UFS ISE Driver integrity test parameters are configured at build time and then executed at runtime. As the driver is included in the kernel as part of the vmlinux file, the HMAC-SHA-256 is calculated over the entire area of the FIPS APIs, but must take into account the relocatable addresses which should not be included in the HMAC calculation. At build time the gap begin/end addresses (`__fips140_text_start/__fips140_text_end` and `__fips140_rodata_start/__fips140_rodata_end`) are determined and used as the parameters along with the HMAC key (`ufs_pixel_fips_hmac_key`) when building the vmlinux ELF, storing the calculated digest (`ufs_pixel_fips_hmac_expected`) in the module.

When the device starts up and the self test is initiated, the memory containing the module, minus the addresses specified as parameters is used to calculate the HMAC value. This is then compared to the stored value (`ufs_pixel_fips_hmac_expected`) from the build time calculation. A non-match will place the module into the error state.

# Mitigation of Other Attacks

No other attacks are mitigated.

# Design Assurance

**Configuration Management**

The source code for the software component and the hardware design documents of the module are maintained in git repositories. The source code manifest includes a build configuration providing a list of the specific tools needed to reproduce the module.

Documentation related to the module is maintained in Google Docs. All documents (whether spreadsheets, documents, presentations or anything else) are automatically version tracked along with the owner. Like git, Docs uses access control lists to control access to the design documentation for the module.

**Delivery and Operation**

Google follows the same build steps to produce the binary version of the module as laid out in the configuration management solution when creating the software that will be installed on the device. The build process is strictly controlled within Google, including access controls on the generation and certification of the builds to be used on a device for production.

As part of the manufacturing process, the Linux kernel image (as part of the broader firmware that will be installed on the device/tested platform), will be securely transferred to the factory using a secured internal server. The factory will then install the binary (which includes the driver) on the device. Once the devices manufacturing is complete, they go to the sales channels for sale to customers. The hardware component is included as part of the manufacturing process for the Google Tensor.

# Secure Operation

The module is provided directly to solution developers and is not available for direct download or purchase by the general public. The module (hardware and software) is bundled together in the mobile device as specified in Table 6 or other platforms where that maintain portability is maintained. The End Users do not have the ability to install, modify or remove the module from the device.

# Glossary and Abbreviations

| | |
|---|---|
| AES | Advanced Encryption Specification |
| CAVP | Cryptographic Algorithm Validation Program |
| CMVP | Cryptographic Module Validation Program |
| CO | Crypto Officer |
| COTS | Commercial Off The Shelf |
| CSP | Critical Security Parameter |
| DMA | Direct Memory Access |
| FCC | Federal Communications Commission |
| FIPS | Federal Information Processing Standards Publication |
| FMC | Flash Memory Controller |
| HMAC | Hash Message Authentication Code |
| IV | Initialization Vector |
| KAT | Known Answer Test |
| NIST | National Institute of Science and Technology |
| OS | Operating System |
| SHA | Secure Hash Algorithm |
| SOC | System on Chip |
| UFS | Universal Flash Storage |
| UFSHCI | Universal Flash Storage Host Controller Interface |
| XTS | XEX Tweakable Block Cipher with Ciphertext Stealing |

*Table 9 - Abbreviations*