Bluechip Systems LLC

MicroCloud X4

FIPS 140-2 Cryptographic Module Security Policy
Non-Proprietary

Version: 3.10

Date: July 17, 2017

Table of Contents

List of Figures

List of Tables

1 Introduction

This document defines the Security Policy for the Bluechip Systems MicroCloud X4 module, hereafter denoted "the Module." The Module is a Linux computer in a microSD form factor, providing hardware isolated cryptographic services to host devices into which it is inserted. The Module meets FIPS 140-2 overall Level 2 requirements.

Table 1 Cryptographic Module Configurations

	Module	HW P/N and Version	Firmware Version	Manufacturable P/N
1	MicroCloud X4 4GB (licensed export, 256-bit encryption)	MCX4-004	X4 Linux 3.4.110.1 MicroCloud Manager 1.9	BCMCC4X4M64I-A1
2	MicroCloud X4 8GB (licensed export, 256-bit encryption)	MCX4-008	X4 Linux 3.4.110.1 MicroCloud Manager 1.9	BCMCC8X4M64I-A1

The Module is intended for use by US Federal agencies and other markets that require FIPS 140-2 validated data-at-rest (DAR) encryption. The Module is a multi-chip standalone embodiment; the cryptographic boundary is the exterior of the microSD package.

The FIPS 140-2 security levels for the Module are as follows:

Table 2: Security Level of Security Requirements

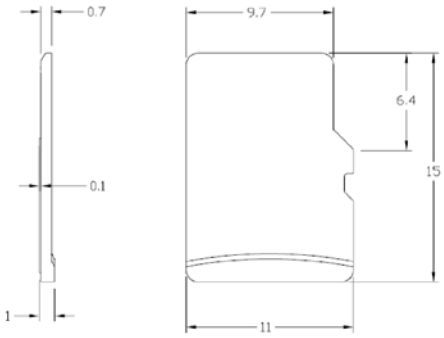
Security Requirement	Security Level
Cryptographic Module Specification	2
Cryptographic Module Ports and Interfaces	2
Roles, Services, and Authentication	2
Finite State Model	2
Physical Security	3
Operational Environment	N/A
Cryptographic Key Management	2
EMI/EMC	3
Self-Tests	2
Design Assurance	2
Mitigation of Other Attacks	N/A

The Module implementation is compliant with:

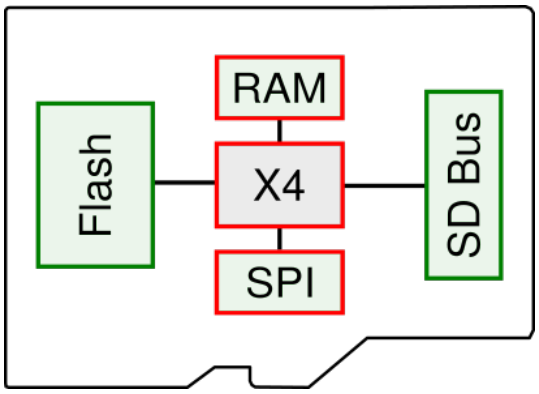
- Secure Digital Specification v2.0 (http://www.sdcard.org)

1.1 Hardware and Physical Cryptographic Boundary

The physical form of the Module is depicted in Figure 1; the physical cryptographic boundary is the exterior of the module enclosure. The MicroCloud X4 is packaged in the industry standard microSD format. The Module relies on a host Android handset to provide power and connectivity to the User and to the Internet (through proxy software running on the handset).



(a) Exterior dimensions



(b) Simplified internal block diagram

Figure 1: MicroCloud X4 Module

Table 3: Ports and Interfaces

Port	Description	Logical Interface Type
DAT2	SD Serial Data 2	Data In, Data Out, Control In, Status Out
DAT3	SD Serial Data 3	Data In, Data Out, Control In, Status Out
CMD	Command, Response	Control In, Status Out
VDD	Power	Power
CLK	Serial Clock	Control Input
VSS	Ground	Power
DAT0	SD Serial Data 0	Data In, Data Out, Control In, Status Out
DAT1	SD Serial Data 1	Data In, Data Out, Control In, Status Out

1.2 Firmware and Logical Cryptographic Boundary

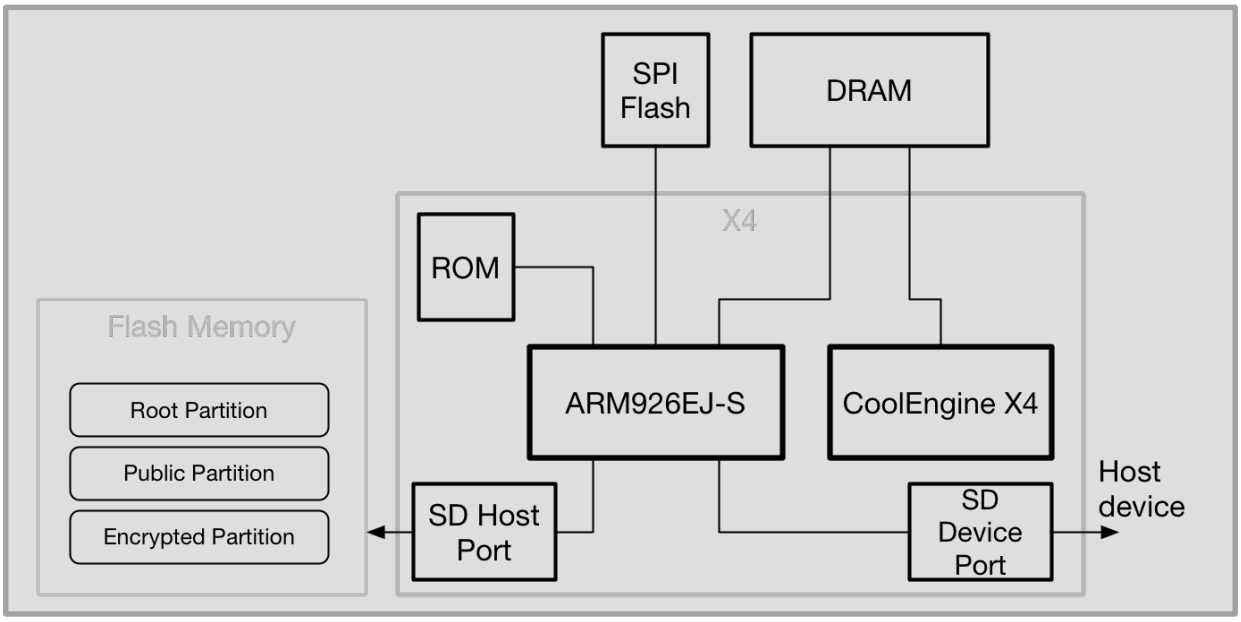


Figure 2: Module Firmware/Physical Block Diagram

As shown in Figure 2, the MicroCloud X4 (greyed area) contains:

- The X4 system on chip (SOC), which includes:
  - An ARM926EJ-S RISC CPU
  - A Bluechip Systems CoolEngine X4 multi-core DSP
  - A ROM containing the X4 boot code
  - SD Device and SD Host interface ports
- A SPI flash memory device
- A 64 MB low-power, mDDR DRAM
- A flash controller (not shown) connected to either 4 GB or 8 GB of flash memory
- A power converter and clock (not shown)

The only external port available is the SD device port, which is used to connect the MicroCloud X4 to a host device. The SD device port provides power to the MicroCloud X4. To the host device, the MicroCloud X4 appears as a normal SD mass-storage device. All traffic through the SD device port is mediated by firmware provided with MicroCloud X4 Linux and executed by the ARM CPU.

The flash memory is partitioned into multiple partitions. By default, only the public partition is made visible to the host device.

The SPI flash memory is a small (256 KB) flash memory connected directly to the X4 SoC. The SPI flash memory is used to store a small program (the boot loader) that is loaded and executed by the X4 at each power up. The SPI flash memory device also contains configuration registers that allow the device to be locked into a read-only state, preventing modification.

Simplified logical block diagrams of MicroCloud X4 operation are shown in Figure 3 and Figure 4. A detailed explanation of operation is given following the figures.

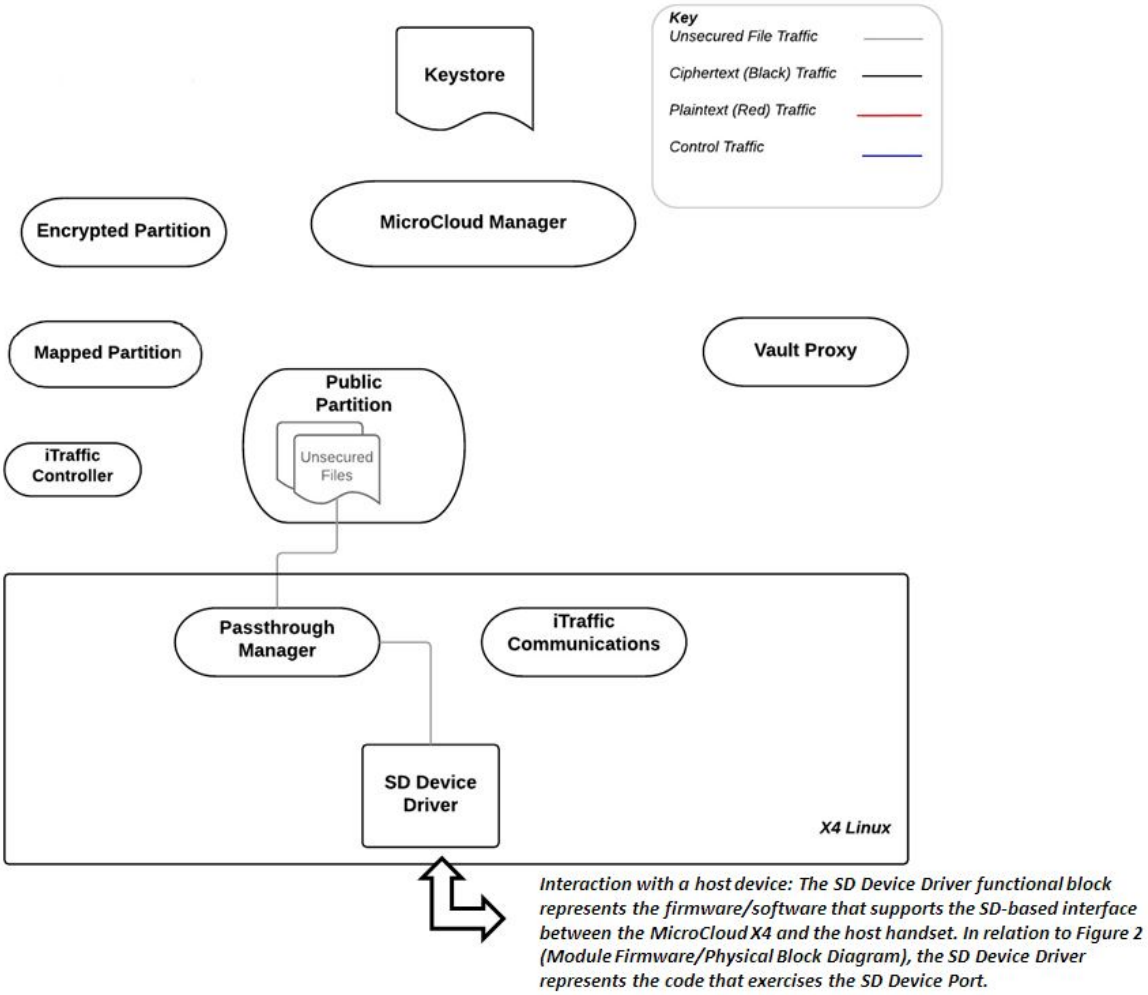


Figure 3: Public Partition access

Figure 3 shows the firmware components used in normal “pass-through” operation, which is available at all times as an unauthenticated service. SD commands issued by the host over the SD bus are processed by the Passthrough Manager, a firmware component of X4 Linux.

As shown in Figure 3, a host handset application can read a file in the MicroCloud X4’s public partition by using Android standard interfaces for reading to a microSD card. By default, the MicroCloud X4 product treats the public partition as read-only, and does not complete writes from the host (writes are allowed, but are not committed to the filesystem.) The one exception is the iTraffic Communication Shadow File (discussed below), to which writes are “committed.”

The MicroCloud Manager and Vault Proxy are not involved in this operation.

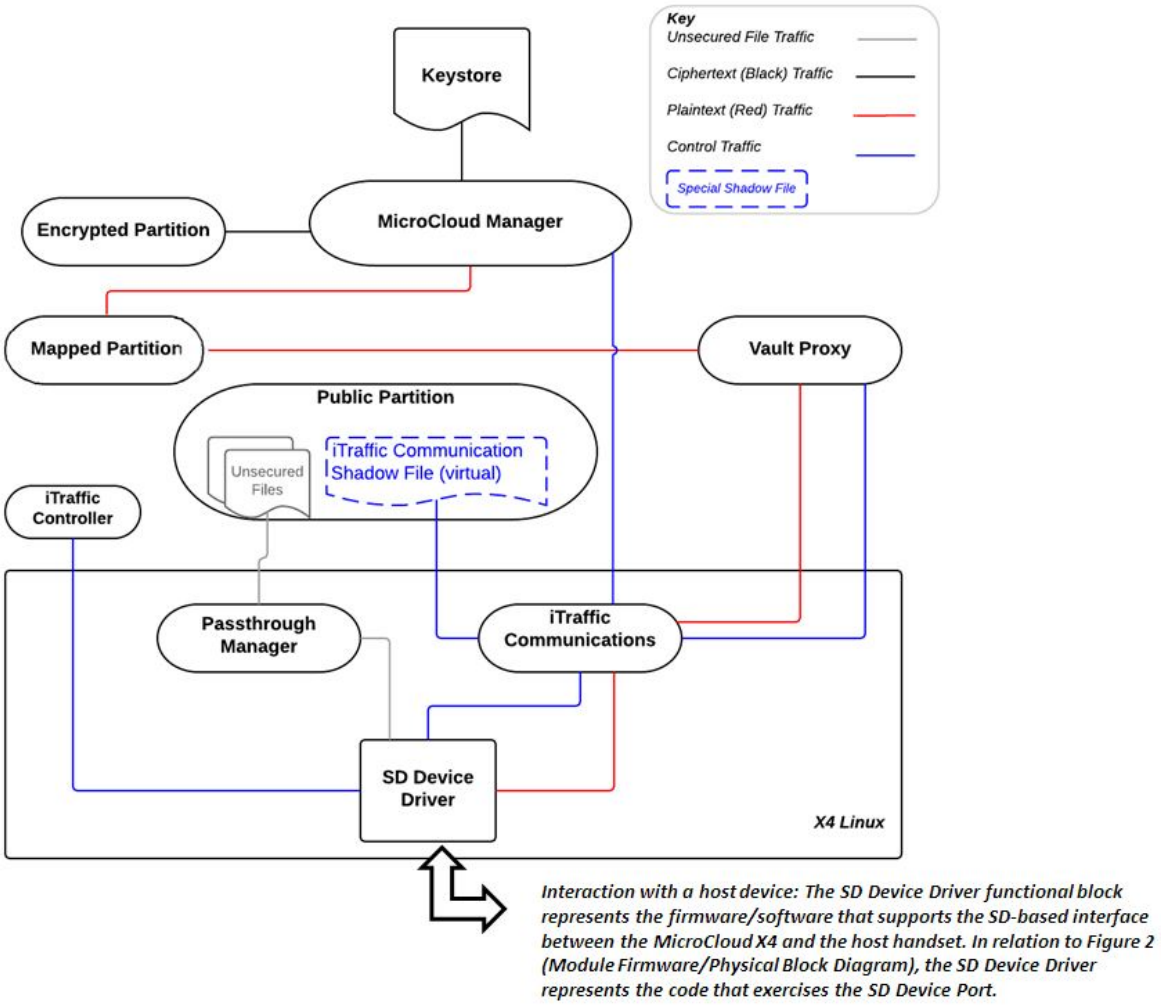


Figure 4: Vault mode

Figure 4 shows the steady state operation of Vault mode. The blue control traffic path shown from the SD Device Driver to the MicroCloud Manager represents a static iTraffic channel setup between a host handset application and the MicroCloud Manager. This channel is used for operations such as PIN entry and logout. iTraffic data is passed between MicroCloud Manager and host via reads and writes into the iTraffic Communication Shadow file (ICSF). The ICSF file is special: although it exists on the public partition, all accesses to it are intercepted by X4 Linux and pass through RAM buffers. That is, a write to the ICSF will never be written to the MicroCloud X4 flash memory and a read of the ICSF never returns data from the MicroCloud X4 flash memory.

The Mapped Partition is not made available directly to the host handset. Access to the Mapped Partition is controlled by the Vault Proxy. Thus all standard file operations (create, write, read, rename, delete, etc.) are mediated by the Vault Proxy.

In Vault mode, the iTraffic Communications module handles both control and user data which is sent to the Vault Proxy. This provides a level of exclusivity for secure data-at-rest services: a host handset application must be able to route both control and user data over an iTraffic channel. The details of

iTraffic channel setup and control are hidden from host applications by the Bluechip-proprietary *Velocity API*.

1.2.1 The Role of the Passthrough Manager and iTraffic Controller

The public partition is made available to the host device using the Passthrough Manager (PM), a firmware component of X4 Linux. A master boot record (MBR) is constructed and provided to the host by the PM shortly after the MicroCloud X4 has completed its boot process. The generated MBR informs the host of the location of the public partition, which is different from the physical location of the public partition in the MicroCloud X4 flash memory. The host may read and write any sector in the public partition at any time. The PM remaps the requests to match the actual sectors in public partition. Accesses out of bounds are discarded.

Applications running on the handset may be written using the Bluechip *iTraffic SDK* to allow communication with applications running on the MicroCloud X4. Communications are made through point-to-point, unidirectional *iTraffic channels* (defined by each application) that are managed by the *iTraffic Communications* module. iTraffic channel data does not have to be encrypted, but handset and MicroCloud X4 applications may encrypt data using any applicable scheme.

The iTraffic Communications module implements the iTraffic channels over the SD bus using standard file read and write operations ostensibly to a special file stored on the public partition. This special file is the iTraffic Communication Shadow File (iCSF). Note that while the iCSF is located in the Public Partition, X4 Linux intercepts reads and writes to the iCSF. All iTraffic data exchanged between the MicroCloud X4 and the host handset "through" the iCSF is transient and is never written to the MicroCloud X4 internal flash memory. When a Vault-enabled host handset application uses the Velocity API to write a file (e.g., a picture) to the MicroCloud X4 using the Vault Service, the Vault Service is communicating over an iTraffic channel to the Vault proxy application executing on the MicroCloud X4.

The iCSF is used for the following system security functions:

- Transfer of the PIN, authentication phrase, and entropy data used for random seeding is setup between an iTraffic enabled host handset application and the MicroCloud Manager via the iCSF
- Control plane information for the Vault mode of operation is exchanged via the iCSF. For example, when a host handset application requests secure storage of a file, the filename to be used is conveyed via the iCSF.

The MicroCloud Manager implements the cryptographic functions supported by MicroCloud X4. The MicroCloud Manager is built with the SAIFE cryptographic library, and the SAIFE cryptographic library incorporates an OpenSSL FIPS Object Module (version 2.0.9). The main function of the MicroCloud Manager is to provide data-at-rest encryption to Vault-enabled handset applications.

To use Vault mode, the user must first authenticate with the MicroCloud Manager. This is done via the iTraffic Communications block which receives the PIN/Passphrase from the SD Device Driver and forwards the information to the MicroCloud Manager. However, note that the iTraffic Communications block has no knowledge of the meaning of the data being passed over the iTraffic channel into the MicroCloud Manager.

The PIN/Passphrase is input to a Password Based Key Derivation Function based on NIST 800-132 (PBKDF). The output of the PBKDF is the Password Key (PW Key). The PW Key is used to seed the AES Key Wrap algorithm, for the purpose of encrypting/decrypting the Key Wrap Key. Decryption of the

associated Encrypted Key Wrap Key verifies the integrity of the decryption through the methods inherent to AES Key Wrap algorithm.

When the PIN/Passphrase has been successfully verified as described above, the system provides visual feedback to the user by displaying a character string that the user (or officer responsible for initial provisioning of the MicroCloud X4) programmed into the MicroCloud X4 during provisioning. This character string is called the "authentication confirmation phrase", but this name should not be taken to imply that it is a cryptographic authentication mechanism; it is a simple visual feedback mechanism to the user that the MicroCloud X4 just unlocked through correct PIN/Passphrase entry was programmed, during its provisioning, with the authentication confirmation phrase that the user expects. Procedurally the user verifies, on each unlock attempt, that the authentication phrase programmed during provisioning is reproduced when the PIN/passphrase is entered; this serves as a non-cryptographic verification to the user that he/she is dealing with the MicroCloud X4 originally provisioned, not with a possible substituted imposter device. The authentication confirmation phrase is sent to the ITraffic Communications block, which forwards it to the host device applications via the SD Device Driver.

On successful PIN/password entry, the *Encrypted Partition* (which lies in the MicroCloud X4 flash memory) is mapped to the Mapped Partition by the Linux device mapper. The device mapper provides a data filter interface to which the MicroCloud Manager is connected. The filter lies completely between the Encrypted Partition and the Mapped Partition such that all data that is read from or written to the Mapped Partition must pass through this filter. The Mapped Partition is then mounted within the X4 Linux filesystem. The Mapped Partition is mounted to a mount point that is not exported by the PM. The only access is via the Vault Proxy. The Vault Proxy receives commands from the *Vault Service* via iTraffic. Figure 4 shows the data flow for Vault Service reads and writes.

Communication between the MicroCloud Manager and the user is done using a handset application that is out of scope of this document.

Vault Service

The Vault Service is an Android Service that runs on the Android handset. Vault-enabled applications communicate with the Vault Service using the *Velocity API*, which provides an API that is compatible with the Java File API. The Vault Service communicates with the Vault Proxy via iTraffic.

1.3 Modes of Operation

The MicroCloud X4 has the following approved modes of operation:

- **Vault Mode with Firmware Update**. After the User has been authenticated to the MicroCloud X4, an internal partition is connected to the encryption/decryption filter provided by the MicroCloud Manager. Only an iTraffic-enabled application running on the handset, which has been connected to the Vault Proxy, is granted access to the encrypted data on the MicroCloud X4. Firmware Update is possible using properly signed packages.
- **Vault Mode without Firmware update**. Identical to Vault Mode with Firmware Update, except that Firmware Update is not possible.
- **Firmware Update-only Mode**. In this mode, firmware updates are possible using the Firmware update module. Vault Mode is not possible.
- **Status-Only Mode**. The host device may read from the public partition, but may not write data to it that will persist across MicroCloud X4 power cycle events.

The MicroCloud X4, on being supplied power, automatically executes a series of tests intended to establish its health. Details of the tests are provided in Section 4. Self testing occurs in three phases:

1) boot testing,
2) post-boot integrity testing, and
3) module self-testing.

During the boot testing phase, the following tests are executed:

CRC Check of Secure Boot Loader,
SHA message digest verification of X4 Linux image[1]
ECDSA signature verification of X4 Linux image

A failure in any of these boot testing phase tests halts all further operation - the MicroCloud X4 enters an infinite loop.

During the post-boot integrity testing phase, the following module tests are executed:

SHA-256 message digest verification of the MicroCloud Manager module
SHA-256 message digest verification of the Vault Proxy module
SHA-256 message digest verification of the iTraffic controller module, and
SHA-256 message digest verification of the Firmware Update module

The consequence to MicroCloud X4 operation of a failure, or a set of failures, during post-boot integrity testing varies depending on the mix of test failures; see the text below accompanying Table 3.5 for details.

During the Module self-testing phase, the following tests are executed as a result of launching the MicroCloud Manager module:

OpenSSL FIPS Object Module SE AES KAT health check
OpenSSL FIPS Object Module SE AES KW KAT health check
OpenSSL FIPS Object Module SE CTR_DRBG KAT health check
OpenSSL FIPS Object Module SE ECDSA Keygen, Sign, Verify PCT health check
OpenSSL FIPS Object Module SE SHA-256 KAT health check, and
CoolEngine AES XTS KAT health check

The consequence of a failure in any of the above tests is the loss of access to all Authenticated Services (see Table 12 for details), with the exception of Firmware Update. In other words, all Authenticated Services dependent on MicroCloud Manager and/or CoolEngine are unavailable. In addition, the “Show Status” unauthenticated service is unavailable.

[1] SHA digest verification occurs during calculation of the “Alpha Hash,” which is stored with the Linux object. For more details on the Alpha Hash, see the MicroCloud X4 Firmware Update Process document.

Table 3.5: Post-Boot Integrity Testing Dependent Entry Conditions for Operating Modes

Each row below captures a combination of Post-Boot Integrity Testing results...				...that causes the X4 to enter the Operating Mode in this column
MicroCloud Manager Check	Vault Proxy Check	Firmware Update Check	iTraffic Controller Check	
Pass	Pass	Pass	Pass	Vault Mode with Firmware Update
Pass	Pass	Fail	Pass	Vault Mode without Firmware Update
Pass	Fail	Pass	Pass	Firmware-Update Only Mode
Fail	Pass			
Pass	Fail	Fail	Pass	Status-Only Mode
Fail	Pass			
Fail	Fail			
Pass or Fail	Pass or Fail	Pass or Fail	Fail	

As noted above, the supported mode of operation depends on the result of post-boot integrity checks performed by X4 Linux on the following modules:

1. MicroCloud Manager
2. Vault Proxy
3. Firmware Update
4. iTraffic

The integrity tests are performed by computing SHA-256 checksums for the binary components as described in Section 4 of each module and comparing the results against expected values generated when the modules are built and packaged and stored in the read-only root partition of X4 Linux. If the computed value does not match the expected value, then the module is erased from the MicroCloud X4 device. Each module may fail the integrity check and be removed independently of the others.

If either or both of the MicroCloud Manager or Vault Proxy modules fail, then Vault Mode is non-operational and any data stored in the encrypted partition is inaccessible. The Firmware Update module may be used to replace the failed modules, so that recovery to an operational Vault Mode is possible.

If the Firmware Update module fails, Vault Mode operation may still be possible (provided the MicroCloud Manager and Vault Proxy modules have not failed), but firmware updates will not be possible.

If the iTraffic module fails (with or independently of the other modules) neither Vault Mode nor Firmware Update modes will be functional. Status information may be read from the read-only public partition.

The post-boot integrity check conditions under which the modes of operation are entered are detailed in Table 3.5, above.

To verify that a module is in an Approved mode of operation, special software (for example, the "Android Controller") is run on the handset. The Android Controller software communicates with the MicroCloud Manager firmware running on the MicroCloud X4, which in turn provides status. The Android Controller only requests state and state changes, and provides the information from the User required to effect the state change (e.g., the PIN or passphrase obtained from the user via the handset UI). The Android Controller can use the status information provided by the MicroCloud Manager to provide indication of the current state, for example, by using colored borders or toast[2] messages. The key APIs used by software running on the handset to query Module state are shown in Appendix A.

2 Cryptographic Functionality

The Module implements the FIPS Approved and Non-Approved but Allowed cryptographic functions listed in the table(s) below.

Table 4: Approved and CAVP Validated Cryptographic Functions

Algorithm	Modes of Operation	Description	Cert #
AES CoolEngine	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[FIPS 197, SP 800-38A, SP 800-38E] Functions: Encryption, Decryption Modes: ECB, XTS Key sizes: 256 bits	4250[3]
AES SAIFE Lib	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[FIPS 197, SP 800-38A, SP 800-38F] Functions: Encryption, Decryption Modes: ECB, KW, KWP Key sizes: 256 bits	4251
DRBG OpenSSL-FIPS (AES CTR_DRBG)	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[SP 800-90A] Functions: AES CTR_DRBG Security Strengths: 256 bits	1329
ECDSA OpenSSL (FW Update)	• Firmware-Update Only • Vault Mode with Firmware Update	[FIPS 186-4 and ANSI X9.62-2005] Functions: Signature Verification	991

[2] In the Android OS, a toast message is a small, popup message that is displayed briefly on behalf of an application before being automatically dismissed by the OS.
[3] 128 bit keys were CAVP tested but are not used by any of the module's services.

		Curves/Key sizes: P-256, P-384	
ECDSA OpenSSL-FIPS (P-384)	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[FIPS 186-4 and ANSI X9.62-2005] Functions: Key Pair Generation, Signature Generation, Signature Verification Curves/Key sizes: P-384	992
ECDSA Secure Boot Loader	• Not Applicable (used during POST, before the module is in an Approved Mode of operation)	[FIPS 186-4 and ANSI X9.62-2005] Functions: Signature Verification Curves/Key sizes: P-256 Executed during Power Up Self Testing	990
HMAC SHA-256 OpenSSL-FIPS	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[FIPS 198-1] Functions: PBKDF Primitive SHA sizes: SHA-256	2789
SHS Open SSL (FW Update)	• Firmware-Update Only • Vault Mode with Firmware Update	[FIPS 180-4] Functions: Message Digest Generation SHA sizes: SHA-256	3488
SHS OpenSSL-FIPS	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[FIPS 180-4] Functions: Message Digest Generation SHA sizes: SHA-256	3489
SHS Secure Boot Loader (CoolEngine SHA)	• Not Applicable (used during POST, before the module is in an Approved Mode of operation)	[FIPS 180-4] Functions: Message Digest Generation SHA sizes: SHA-256 Executed during Power Up Self Testing	3487

Table 5: Approved Cryptographic Functions Tested with Vendor Affirmation

Algorithm	Mode(s) of Operation	Description	IG Ref.
KDF, Password-Based	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	[SP 800-132] Options: PBKDF with Option 2a Functions: HMAC-based KDF using SHA-256 Key sizes: The 256 bit MK, the key derived from the PBKDF, is never used directly as a data protection key (DPK). Per Option 2a, it instead seeds the AES Key Wrap Algorithm, which encrypts the Key Wrap Key and stores it in non-volatile memory as the Encrypted Key Wrap Key. The Key Wrap Key is the DPK. Key sizes: The key derived from the PBKDF is, per option 2a of SP 800-132, used to seed the AES Key Wrap Algorithm, which encrypts the system's Key Encrypt Key (KEK), called the "Key Wrap Key." The encrypted Key Wrap Key is stored in non-volatile memory. The Key Wrap Key is the DPK called out in SP 800-132. Keys derived from passwords may only be used in storage applications.	Vendor Affirmed IG D.6

Table 6: Non-Approved but Allowed Cryptographic Functions

Algorithm	Description
None	The module does not implement Allowed Cryptographic functions.

Table 7: Protocols Allowed in FIPS Mode

Protocol	Description
None	The module does not implement cryptographic protocols.

Non-Approved Cryptographic Functions for use in non-FIPS mode only: None.

2.1 Critical Security Parameters

All CSPs used by the Module are described in this section. All usage of these CSPs by the Module (including all CSP lifecycle states) is described in the services detailed in Section 4.

Table 8: Critical Security Parameters (CSPs)

CSP	Description / Usage
Password	A minimum of four octets (bytes) that are used as input to a password based key derivation function (PBKDF). 256 bits of the PBKDF output is the PW Key.
DRBG SEED	1024 octets (bytes) of seeding data provided to the MicroCloud X4 by a host device application. Based on an assumed entropy strength per byte of 7 bits, the module is assumed to be seeded with at least 384 bits of entropy. The module is seeded with a minimum of 384 bits of entropy.
DRBG STATE	Value V of the outlen bits (128 bits), the 256 bit Key (AES 256), and the reseed_counter counter value are maintained within the OpenSSL FIPS Module
Key Wrap Key	256 bit key that serves as Key Encrypt Key (KEK) for the Private Key and Volume Key. The Key Wrap Key is stored in non-volatile memory encrypted by the PW Key
PW Key	256 bit key that enables decrypting the Key Wrap Key. The PW Key is not persistently stored, it is derived using Password Based Key Derivation Function (PBKDF) operating on the Password input.
Private Key	ECDSA P-384 Private Key for possible future Data In Transit capability. Not required or involved in providing secure DAR service. The Private Key is not persistently stored in plaintext, it is stored encrypted using the Key Wrap Key. The Public Key counterpart is the public key contained in the X4 SAIFE Certificate. The Private Key is created during SMSI CA Provisioning. It may play a role in possible new features such as data in transit; the Private Key is not, however, required for the purposes of DAR processing in the X4.
Volume Key	256 bit Key for decrypting Ciphertext Data At Rest. The Volume Key is stored in non-volatile memory in encrypted form.

2.2 Public Keys

Table 9: Public keys

Key	Description / Usage
SMSI CA Public Key	Public component of an Elliptic Curve (EC) key pair (based on the NIST P-384 curve) used by the SAIFE library built into the MicroCloud Manager to verify signature of provisioning data received from the SAIFE Management Services Instance Certificate Authority (SMSI CA). Interactions with the SMSI CA involve authentication only, in the form of Elliptic Curve Digital Signature Algorithm (ECDSA) signature verification. The SMSI CA Public Key is used to verify signatures of provisioning data received from the SAIFE Management Services Instance Certificate Authority (SMSI CA).
X4 SAIFE Certificate	Certificate containing a SMSI-signed Public Key (NIST P-384 Elliptic Curve) for possible future Data In Transit capability. Not used or required for data at rest

	services. Provisioning with the SAIFE Management Services Instance (SMSI) Certificate Authority (CA) establishes a SMSI-CA-signed certificate which is stored in the X4; this certificate is called the "X4 SAIFE Certificate". The X4 SAIFE Certificate may be used for possible new features such as data in transit. The X4 SAIFE Certificates capability is not, however, required for the purposes of DAR processing in the X4.
Linux Secure Boot Public Key	Public component of an EC key pair (based on the P-256 curve) used to verify the validity of the MicroCloud X4 Linux kernel as it is loaded during device power-up, in the form of Elliptic Curve Digital Signature Algorithm (ECDSA) signature verification. This public key is stored in the MicroCloud X4's Serial Peripheral Interface (SPI) Flash memory. The SPI flash memory is a small (256 KB) flash memory connected directly to the X4 SoC. The SPI flash memory contains configuration registers that are used during MicroCloud X4 production to lock the device into a read-only mode, preventing field modification. Public component of an EC key pair (based on the P256r curve) used to verify the validity of the MicroCloud X4 Linux kernel as it is loaded during device power-up, in the form of Elliptic Curve Digital Signature Algorithm (ECDSA) signature verification.
Root Public Key	Public component of a key generated using the ECDSA P-384 curve. The private component is used to create the CASoC System Certificate. This public key is stored in the X4 Linux root file system, which is read-only. See the discussion on key signing hierarchy following Table 12 for details on how firmware updates are verified.
CASoC System Certificate	Certificate containing a public key generated using the ECDSA P-384 curve, used to verify the signature in the X4 System Installation Certificate. The CASoC System Certificate is signed by the private key counterpart of the Root Public Key. See the discussion on key signing hierarchy following Table 12 for details on how firmware updates are verified.
X4 System Installation Certificate	Certificate containing a public key generated using the P-256 curve, used in the X4 Linux Package verification process. The X4 System Installation Certificate is signed by the private key counterpart of the CASoC System Public Key contained in the CASoC System Certificate. See the discussion on key signing hierarchy following Table 12 for details on how firmware updates are verified.
X4 Package Signing Certificate	Certificate containing a public key generated using the P-256 curve used to verify signatures of X4 Packages. The X4 Package Signing Certificate is signed by the private key counterpart of the X4 System Installation Public Key contained in the X4 System Installation Certificate. See the discussion on key signing hierarchy following Table 12 for details on how firmware updates are verified.

3 Roles, Authentication and Services

3.1 Assumption of Roles

The module supports two operational roles, that of the Cryptographic Officer and User. Operationally, the two roles are identical. The module has a single operator, so both the CO and User roles are assigned to the single operator.

A factory maintenance function is performed by personnel, who configure the MicroCloud X4 prior to its entering field operations, but factory maintenance functions are not subject to any cryptographic access control; i.e., there is no kind of credential or authentication mechanism required. Once the MicroCloud X4 has been initially configured by a factory maintenance operation, the customer creates the User Role by provisioning the MicroCloud X4 and defining a PIN that will be required to access security services. Once the User Role has been established, the MicroCloud X4 will erase all user data if the MicroCloud X4 is re-initialized.

Table 10 lists all operator roles supported by the module. The Module does not support a maintenance role. The Module does not support concurrent operators.

Table 10: Roles Description

Role ID	Role Description	Authentication Type	Authentication Data
Cryptographic Officer	Same as User	Same as User	Same as User
User	User – able to authenticate to MicroCloud X4 using PIN code, able to invoke security services.	Identity-based	PIN

3.2 Authentication Methods

3.2.1 PIN

The PIN authentication method collects a multi-character PIN code using an application on the handset. The PIN is passed securely to the MicroCloud Manager, which uses the PIN to unlock the keystore (which is stored encrypted on the MicroCloud X4 flash memory), as well as to access the confirmation passphrase. The MicroCloud Manager enforces a minimum four character PIN length—i.e., a four octet minimum, each octet capable of assuming a value in the hexadecimal range (0x00...0xFF). No composition check within a PIN is performed, i.e., the MicroCloud Manager does not check how many characters in a PIN are, or are not, in the ASCII character code numeric range (0x30...0x39).

The PIN is collected using a user interface presented to the user via the handset display. Longer PINs take longer to enter. Transmitting an invalid PIN to the MicroCloud X4 and using it to attempt to unlock the keystore takes very little time, typically just a few hundred milliseconds. Thus, the rate of PIN attempts is dominated by PIN entry, which is proportional to the length of the PIN as well as the composition of each individual character.

Only one PIN can be provided to the MicroCloud X4 per authentication (keystore unlock) attempt. The MicroCloud Manager allows a maximum of ten consecutive failed PIN attempts. After ten consecutive

failed attempts, the device will be sanitized (zeroized), and must be reprovisioned. If the correct PIN is entered before the tenth consecutive failed attempt, the number of failures is reset.

In order for MicroCloud X4 operation to satisfy the requirement that for any given PIN entry attempt, the probability shall be less than 1 in 1,000,000 that a random attempt will succeed, the customer must enforce the following recommendations for PIN composition—the MicroCloud X4 will not enforce these PIN composition rules, the MicroCloud X4 only enforces a minimum PIN length of four characters. The PIN is used in the PBKDF function, so the length and complexity of the PIN is the upper bound for the probability of having this parameter guessed at random.

A four digit PIN composed of 8-bit characters results provides 2^8 * 2^8 *2^8 *2^8 or 4,294,967,296 possible combinations, so the probability of a random authentication attempt succeeding is less than 1 in 1,000,000.

If the entered PIN will be comprised solely of numbers taken from the set [0...9], the recommendation to customers is that the PIN be comprised of a minimum of seven numbers so that the random probability is less than 1 in 1,000,000 (the probability is lowered to 1 in 10,000,000 through the requirement for a minimum seven number long PIN).

If the entered PIN will be a mix of numbers and non-numeric characters, the recommendation to customers is that the PIN must be comprised of a minimum of four characters, since, for the simplest case, each entry represents one in 36 possibilities [0...9 augmented by a...z], and 36^4 is 1,679,616, which is greater than 1,000,000. If the entered PIN is comprised only of only non-numeric characters, e.g., [a...z], then the PIN must be comprised of a minimum of five characters, since 26^5 is equal to 11,881,376. Note that allowing for characters beyond the simple set of [0...9 / a...z] - e.g., allowing upper case characters and punctuation characters - could arguably lower the minimum PIN composition, for simplicity's sake the above composition rules for a PIN containing non-numeric characters is proposed.

The MicroCloud Manager allows a PIN length of 64 8-bit characters or more, which will allow for a numeric-only PIN composition, a non-numeric-only PIN composition, or an alphanumeric PIN composition, as discussed above.

The MicroCloud X4 meets the requirement that multiple attempts in a one-minute period will succeed with a probability of less than 1 in 100,000 due to the limit on consecutive failed PIN attempts (10 consecutive failed attempts). If the initial PIN is constructed to meet or exceed a 1 in 1,000,000 probability, then the probability of guessing the correct PIN in 10 attempts is lower than the required probability of 1 in 100,000.

Table 11: Authentication Description

Authentication Method	Probability	Justification
PIN	1 in 10,000,000 for a seven digit numeric only PIN 1 in 1,679,616 for a four character alphanumeric PIN 1 in 11,881,376 for a five character non-numeric PIN	Customers should adhere to the PIN composition rules described above.

3.3 Services

All services implemented by the Module are listed in the table(s) below. Each service description also describes all usage of CSPs by the service.

PIN or passphrase values must be specified at the time the MicroCloud X4 is keyed. The PIN/Passphrase must have a minimum length of four characters.

In the context of providing secured data at rest (DAR) services for user information, there is no asymmetric key generation involved.

Table 12: Authenticated Services

Service	Description	Modes of Operation	CO/U
PIN/passphrase Verification	Provides cryptographic verification of a user entered PIN/Passphrase; on successful verification, a user-programmed phrase is provided by the MicroCloud X4 to the Android Controller for display to the user.	● Vault Mode with Firmware Update ● Vault Mode without Firmware Update	X
Read/Write Data	Reads or writes data to/from the encrypted storage area. Executes using AES XTS.	● Vault Mode with Firmware Update ● Vault Mode without Firmware Update	X
Zeroize	Destroys all CSPs except the Root Public Key, the Linux Secure Boot Public Key, the CASoC System Certificate, the X4 System Installation Certificate and the X4 Package Signing Certificate.	● Vault Mode with Firmware Update ● Vault Mode without Firmware Update	X
Firmware Update	Updates to the X4 Linux OS and MicroCloud X4 applications using signed packages.	● Vault Mode with Firmware Update ● Firmware Update-only Mode	X
Entropy Load	A Vault-enabled Application running on the host device harvests entropy which is then passed to the MicroCloud X4 for the purpose of seeding the DRBG.	● Vault Mode with Firmware Update ● Vault Mode without Firmware Update	X
SMSI CA Provisioning	During provisioning, the MicroCloud Manager generates an elliptic curve (EC) keypair through operations over the NIST P-384 curve. Operations involving the EC keypair are not involved in the MicroCloud X4's secure DAR processing, they are solely for registration with the SAIFE Management Services Instance Certificate	● Vault Mode with Firmware Update ● Vault Mode without Firmware Update	X

	Authority (SMSI CA). The MicroCloud X4 makes calls to the Deterministic Random Bit Generator (DRBG) for the purpose of forming a random variable that will serve as the Elliptic Curve Private Key (EC Private Key). The EC Private Key is used to multiply the EC generator point, the end result of which is an EC point (coordinate pair) which serves as the Elliptic Curve Public Key (EC Public Key). The MicroCloud X4 forms a Certificate Signing Request (CSR) which contains the EC Public Key. The CSR is self-signed, i.e., the MicroCloud X4 signs the CSR containing its public key by using its EC Private Key to generate an Elliptic Curve Digital Signature Algorithm (ECDSA) signature over the CSR (P-384 curve, SHA-256 digest). The CSR is submitted, via the Host Device's network stack, to the SMSI CA. The SMSI CA returns a certificate which it has signed (P-384 curve, SHA-256 digest) with its private key; the MicroCloud X4 uses the SMSI CA Public Key (factory programmed into the MicroCloud X4) to verify the signature. Provisioning with the SAIFE Management Services Instance (SMSI) Certificate Authority (CA) establishes a SMSI-CA-signed certificate which is stored in the X4. This registration with the SMSI CA enables future certificate management capability for possible new features such as data in transit; this capability is not, however, required for the purposes of DAR processing in the X4.		

PIN/Passphrase verification is initiated by the user via the Android Controller. The output of a password based key derivation function (PBDKF) based on NIST SP 800-132 seeds the AES Key Wrap algorithm, which decrypts the Encrypted Key Wrap Key, resulting in the plaintext Key Wrap Key. Note that the Key Wrap algorithm has built-in integrity verification; therefore, on successful decryption of the Encrypted Key Wrap Key, the MicroCloud X4 provides the authentication phrase to the Android Controller for display to the user. The authentication phrase was established during provisioning.

Reading and writing data to the MicroCloud X4 is only possible after successful PIN/Passphrase verification. Data-at-rest (DAR) is secured using the AES algorithm running in XTS mode. Decrypted DAR information is provided to host applications via the SD device interface.

Firmware Update covers updates to the X4 Linux OS and MicroCloud X4 applications.

X4 Linux OS updates must be signed by the private key cryptographically coupled to the Linux Secure Boot Public Key stored in the MicroCloud X4 SPI memory. Signatures are created in accordance with the Elliptic Curve Digital Signature Algorithm (ECDSA) using the P-256 curve and SHA-256 digest. The Linux Secure Boot Public Key cannot be modified via firmware update.

Updates to X4 Linux or MicroCloud X4 applications must be signed by the private key cryptographically coupled to the X4 Package Signing Certificate stored in the X4 Linux read-only filesystem. Signatures are created in accordance with the ECDSA algorithm.

A firmware update signing hierarchy is implemented in the X4 as follows:

1. The private key corresponding to the Root Public Key signs the CASoC System Certificate.
2. The private key corresponding to the public key in the CASoC System Certificate signs the X4 System Installation Certificate.
3. The private key corresponding to the public key in the X4 System Installation Certificate signs the X4 Package Signing Certificate.
4. The private key corresponding to the public key in the X4 Package Signing Certificate signs X4 firmware update packages.

For details on the keys and certificates in the firmware update signing hierarchy, reference the "MicroCloud X4 Key Management" document. These keys/certificates can be modified via firmware update.

The MicroCloud Manager can be queried by the host at any time for status using software protocols carried via the iTraffic control channel. Status may or may not be made visible to the User, depending on the operational state of the host. For example, when the MicroCloud X4 is first inserted into the handset, it is in the unauthenticated state and appears as a simple mass storage device. In this case, the cryptographic module status is not visible to the User. If the User chooses to run software capable of authenticating the User to the Module, this software would query for and indicate module status.

Table 13: Unauthenticated Services

Service	Mode(s) Of Operation	Description
Show Status	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	Provides MicroCloud Manager status to the Android Controller via iTraffic control channel. Does not affect current status.
Secure Boot	• Not Applicable (part of POST, before the module is in an Approved Mode of operation)	Ensures that the X4 Linux image is correct and has not been tampered with. Initiated by Power Up
Public Partition Access	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	Access to the unencrypted public partition.

	● Firmware Update Only ● Status Only	
Self-test	● Not Applicable	Self-test is performed at power-up and first execution of the MicroCloud Manager firmware.

The status query does not require authentication to operate. It does require the Android Controller or Vault-enabled software to be executed by the User.

Secure Boot is performed at each power up to validate the X4 Linux image. An image that does not meet the SHA-256 hash check and ECDSA P-256 signature check will not be executed, preventing further operation of the Module.

The public partition is a required function of the MicroCloud X4, being essential to its operation as a microSD-compatible device.

Table 14 defines the relationship between access to CSPs and the different module services. The modes of access shown in the table are defined as:

- G = Generate: The module generates the CSP.
- R = Read: The module reads the CSP. The read access is typically performed before the module uses the CSP.
- E = Execute: The module executes using the CSP.
- W = Write: The module writes the CSP. The write access is typically performed after a CSP is imported into the module, when the module generates a CSP, or when the module overwrites an existing CSP.
- Z = Zeroize: The module zeroizes the CSP.

Table 14: CSP Access Rights within Services

Service	Mode(s) of Operation	CSPs							Public Keys						
		DRBG SEED	DRBG STATE	Password	KEY WRAP KEY	PW KEY	PRIVATE KEY	VOLUME KEY	SMSI CA Public Key	X5 SAIFE Certificate	Linux Secure Boot Public Key	Root Public Key	CASoC System Certificate	X4 System Installation Certificate	X4 Package Signing Certificate
PIN/Passphrase Verification	• Vault Mode with Firmware Update • Vault Mode without Firmware Update			*WE*	*GE*	*GE*	*R*	*R*							
Read / Write Data	• Vault Mode with Firmware Update • Vault Mode without Firmware Update				*RE*			*E*							
Zeroize	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	*Z*	*Z*	*Z*	*Z*	*Z*	*Z*	*Z*							
Firmware Update	• Vault Mode with Firmware Update • Firmware-Update Only											*RE*	*RE*	*RE*	*RE*
Show Status	• Vault Mode with Firmware Update • Vault Mode without Firmware Update • Status-Only Mode														
Secure Boot	• Not Applicable										*RE*				
Public Partition Access	• Vault Mode with Firmware Update • Vault Mode without Firmware														

	Update														
Self-Test[4]	• Not Applicable														
SIMSI CA Provisioning	• Vault Mode with Firmware Update • Vault Mode without Firmware Update		*E W*				*G E W*	*G*	*R E*	*RE*	*GW*				
Entropy load	• Vault Mode with Firmware Update • Vault Mode without Firmware Update	*R E W*	*G W*												

4 Self-tests

Each time the Module is powered up it tests that the cryptographic algorithms still operate correctly and that sensitive data have not been damaged. Power up self–tests are available on demand by power cycling the module.

Cryptographic function self-tests are provided by the OpenSSL FIPS Object Module (version 2.0.9), and by additional tests built into the MicroCloud Manager.

On power up or reset, the Module performs the self-tests described in Table 15 below. All Known Answer Tests (KATs) must be completed successfully prior to any other use of cryptography by the Module. If one of the Secure Boot Loader KATs fails, the Module enters the *Secure Boot Error* state. If the Secure Boot Loader KATs pass, then the device will perform the post-boot package integrity KATs. These are performed on a per-package basis. Packages that fail the post-boot package integrity KATs are deleted from the module. If the MicroCloud Manager is not uninstalled following these KATs, then the MicroCloud Manager performs the remaining KATs listed in the table. If one of these KATs fails, then MicroCloud Manager will abort execution and no cryptographic services will be available for use by the host device. The MicroCloud X4 will continue to offer read-only access to the public partition. Appendix A describes how an operator determines if the module is in an error state.

Table 15: Power Up Self-tests

Test Target	Description	Relationship to Operational Modes
AES (Cert. #4251) SAIFE Lib	Test the AES KW algorithm in the SAIFE Library KAT: Known Answer for	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, and Firmware-Update Only Mode

[4] See section 1.3 for a list of which self-tests are performed when during Power Up Self Testing.

	encrypt and decrypt Key sizes: 256 bit	
AES (Cert. #4250) CoolEngine	Test CoolEngine AES-XTS KATs: Encrypt and decrypt tests for XTS_AES (uses AES ECB mode) performed on CoolEngine Key sizes: 256 bits	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, and Firmware-Update Only Mode
CRC (Secure Boot Loader)	Tests the integrity of the Secure Boot Loader stored in the SPI flash. KATs: CRC	This test runs for all modes: Vault Mode with Firmware Update, Vault Mode without Firmware Update, Firmware-Update Only, and Status-Only Mode , and determines whether the MicroCloud X4 can enter any operational mode
DRBG (Cert. #1329) OpenSSL-FIPS	Tests the integrity of the DRBG used by the SAIFE Library KAT: OpenSSL FIPS Object Module SE CTR_DRBG, AES, 256 bit without derivation function	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, and Firmware-Update Only Mode
ECDSA (Cert. #991) / SHS (Cert. #3488) OpenSSL	Check signature against a known-good signature. Used in Firmware Update-only & Vault Mode with Firmware Update KAT: ECDSA Signature Verification Curves/Key sizes: P-256 / SHA256	This test is run for Firmware-Update Only Mode and Vault Mode with Firmware Update
ECDSA (Cert. #992) OpenSSL-FIPS	Test the integrity of the ECDSA used by the SAIFE Library	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, and Firmware-Update Only Mode

	PCT: Sign, Verify Curves/Key sizes: P-384 / SHA256	
ECDSA (Cert. #990) / SHS (Cert. #3487) Secure Boot Loader	Computes ECDSA P-256 signature on the message digest generated from the X4 Linux image. KATs: ECDSA Signature Verification Curves/Key sizes: P-256 / SHA256	This test runs for all modes: Vault Mode with Firmware Update, Vault Mode without Firmware Update, Firmware-Update Only, and Status-Only Mode , and determines whether the MicroCloud X4 can enter any operational mode
SHS (Cert. #3489) Firmware Integrity (Post-Boot Check)	Tests the integrity of the MicroCloud Manager, Vault Proxy, Firmware Update, and iTraffic modules. KATs: SHA-256 on module components. Hash: SHA-256	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, Firmware-Update Only Mode, and Status-Only Mode
HMAC (Cert. #2789) OpenSSL-FIPS	Test the HMAC used by the SAIFE Library KAT: OpenSSL FIPS Object Module SE HMAC Hash: SHA-256	This test is run for Vault Mode with Firmware Update, Vault Mode without Firmware Update, and Firmware-Update Only Mode

For more information, see Table 3.5 Post-Boot Integrity Testing Dependent Entry Conditions for Operating Modes

As noted in Section 1.1, the MicroCloud X4 includes a SPI flash memory that is connected directly to a SPI (serial interface) port of the X4 SoC. The SPI flash memory is loaded with the Secure Boot loader. The X4 ROM loads and executes this code at each power up. During production, the SPI flash memory is loaded with the Secure Boot loader firmware, and then locked into read-only mode by writing to configuration registers internal to the SPI flash memory device. This prevents any further modification of the SPI flash memory.

Table 16: Conditional Self-tests

Test Target	Description
DRBG	SP800-90 Health Tests: Tested as required by [SP 800--90] Section 11.3 upon Instantiate and Generate.
DRBG	Continuous RNG Test: OpenSSL FIPS Object Module SE FIPS 140-2 continuous test for stuck fault. Test performed every time a random value is requested from the DRBG.
ECDSA	OpenSSL FIPS Object Module SE ECDSA Pairwise Consistency Test performed on every ECDSA key pair generation.
Package Update	Firmware Load Test: ECDSA P-256 (SHA-256) signature verification performed before a MicroCloud X4 update package may be installed.

Table 17: Critical Function Tests

Test Target	Description
None	None applicable

5 Physical Security Policy

The microSD package has no moving parts. It contains miniature electronic components and integrated circuits that are bonded together and completely sealed within a plastic epoxy case shaped per the microSD standard package dimensions. Access to the components requires removal of the epoxy, which is a destructive process. Hardness testing was performed at a single temperature (ambient room temperature ≈ 21° C) and no assurance is provided for Level 3 hardness conformance at any other temperature.

Table 18: Physical Security Inspection Guidelines

Physical Security Mechanism	Recommended Frequency of Inspection/Test	Inspection/Test Guidance Details
Tamper Evident Package	1 months	The package is sealed in epoxy. Removal of part or all of the epoxy would be immediately apparent on visual inspection.

6 Operational Environment

The Module is designated as a limited operational environment under the FIPS 140-2 definitions. The Module includes a firmware load service to support necessary updates. New firmware versions within the scope of this validation must be validated through the FIPS 140-2 CMVP. Any other firmware loaded into this module is out of the scope of this validation and require a separate FIPS 140-2 validation.

7 Mitigation of Other Attacks Policy

Not applicable.

8 Security Rules and Guidance

The Module design corresponds to the Module security rules. This section documents the security rules enforced by the cryptographic module to implement the security requirements of this FIPS 140-2 Level 2 module.

1. The module provides two identical operator roles: Cryptographic Officer and User.
2. The module does not provide identity-based authentication.
3. The module clears previous authentications on power cycle.
4. When the module has not been placed in a valid role, the operator does not have access to any cryptographic services.
5. The operator may command the module to perform the power up self-tests by cycling power or resetting the module.
6. Power up self-tests do not require any operator action.
7. Data output is inhibited during key generation, self-tests, zeroization, and error states.
8. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the module.
9. There are no restrictions on which keys or CSPs are zeroized by the zeroization service.
10. The module does not support concurrent operators.
11. The module does not support a maintenance interface or role.
12. The module does not support manual key entry.
13. The module does not have any external input/output devices used for entry/output of data.
14. The module does not output plaintext CSPs.
15. The module does not output intermediate key values.
16. The operator must ensure that the 1024 bytes of entropy input provided during provisioning contains at least 384 bits of security strength.

9 References and Definitions

The following standards are referred to in this Security Policy.

Table 19: References

Abbreviation	Full Specification Name
[FIPS140-2]	*Security Requirements for Cryptographic Modules*, May 25, 2001
[SP800-131A]	*Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, January 2011

Table 20: Acronyms and Definitions

Acronym	Definition
SD	Secure Digital
SMC	SAIFE© Management Console, SAIFE's web-based management portal for managing SAIFE enabled software and hardware.
SMSI	SAIFE Management Services Instance
SPI	Serial Peripheral Interface
TIPRNET	Trusted Internet Protocol Relay NETwork, SAIFE© branded and operated all-black trusted network.

Appendix A: Show Status API

The Controller Service Client runs on the handset and facilitates communication with the Module. The following table shows the key APIs available via the Controller Service Client to query the status of the Module.

`public void requestServiceState();`	Request the Module service state.
`public void handleServiceState(final SaifeState state);`	Override if your activity needs to handle changes in the SAIFE state. This is frequently used during the provisioning and login process. SAIFE States: ● UNINITIALIZED ● UNKEYED ● LOCKED ● PROVISIONED ● UNPROVISIONED ● UNKNOWN