SASE01EPZS
SESIP Security Target

Revision 2.5 November 21, 2024 **Evaluation Document**
PUBLIC

Based on SESIP methodology, version "Public Release v1.1 [1]

Document Information

Information	Content
SESIP1	Evaluated TOE version 2.5
Keywords	SESIP, Security Target, SASE01EPZS
Abstract	Evaluation of the SASE01EPZS developed and provided by IIST Inc. According to SESIP Assurance Level 1 (SESIP1)

Contents

1. Introduction

The Security Target describes the SASE01EPZS platform and the exact security properties of the platform that are evaluated against SESIP [1] methodology, version "Public Release v1.1" in chapter [3. Security Requirements and Implementation] that a potential consumer can rely upon the product upholding if they fulfill the objectives for the environment described in chapter [2. Security Objectives for the Operational Environment].

1.1. ST Reference

SASE01EPZS, SESIP Security Target, Revision 2.5, IIST Inc., November 21, 2024.

1.2. TOE Reference

Table 1.1 TOE Reference

Reference	Value
TOE Name	SASE01EPZS
TOE version	Rev. 2.5
TOE identification	SASE01EPZS
TOE Type	Secure co-processor platform for embedded systems

1.3. Included guidance documents

The following documents are included with the platform:

Table 1.2 Guidance Documents

Document	Reference
Datasheet	SASE01EPZS Datasheet Rev. 2.4
Guidance Manual	SASE01EPZS Instruction Manual Rev. 1.0 (including source codes references and appendices)

1.4. Platform functional overview and description

The SASE01EPZS platform is a co-processor secure element introducing the feature of having on-board multiple root-of-trust secure cryptography credentials sourcing mechanism. This platform is designed with predefined security functionality without the use of any firmware entity. This high-efficiency first product of a secure element series leverages the advanced design of a physically unclonable function (PUF) using a dynamically measurable source of entropy independent from powering, initialization or secret injection sequences. The nature of the integrated innovative PUF design allows SASE01EPZS to dynamically manage multiple truly random digital identities and set of unique credentials, as if there were multiple physically separated root-of-trust on the target host system.

SASE01EPZS enables embedded system, and by extension IoT host processor devices, to possess an isolated physical secure element used as the system multiple root-of-trust engine, comparable to a Digital Signal Processor specifically designed for network and IoT security.

SASE01EPZS is interfaced with a host processor unit, MCU or MPU, via standard communication protocols either UART or SPI up to 12MHz. The platform operates with additional necessary components including a clock oscillator source of up to 48MHz frequency, a non-volatile memory storage (NVM) of at least 40 Kbits and power supply source of 3.3V. The clock oscillator source of up to 48MHz oscillating can be sourced from the host MCU. The non-volatile memory storage (NVM) of at least 40 Kbits can be sourced from the host MCU or from AT24C128 and AT24C256 Atmel EEPROMS.

The NVM external component is accessed via an I2C bus interface, in which SASE01EPZS is the master and the NVM unit is the slave. SASE01EPZS follows the I2C master protocol as defined in [4] and is by default compatible with both AT24C128 and AT24C256 Atmel EEPROMS. A different source of NVM, EEPROM or Flash may be used given the adequate I2C slave protocol implementation and memory allocation on the target NVM slave, necessary but not sufficient to fulfill all SASE01EPZS security operations functions.

SASE01EPZS offers the following features:

- True Random Numbers Generator
- Intrinsic Multiple Root Identities Management
  - Creation with/without External User-defined Input: Automatic on first power-up
  - Loading (also referred as Retrieval)
  - Updating (for extreme environments cases or extra safety measures)
  - Flush (before switching to another Root Identity)
- Functions After Root Identity in-use, also referred as loaded
  - Unique Response:

    Hardware Root Key response to input 32 bytes seed to generate a unique 32 bytes response. The unique response is generated using a proprietary Key-Derivation Function and Format Preserving Encryption (KDF-FPE) algorithm, combined with unsalted SHA-256 (SHA-2). The KDF-FPE algorithm uses a 1024 bits random key recovered internally by the PUF root key(s) engine.
  - Encryption/Decryption:

    Local Encryption by 4 bytes or 32 bytes user input data. The input are encrypted/decrypted in two steps: first through the KDF-FPE, same KDF-FPE as for the Unique Response function, and then combined with AES-256 using a 256 bits secret internally derived from the KDF-FPE 1024 bits key.
    - Usage 1: Volatile responses to input plain text or cipher text for local data encryption and decryption
    - Usage 2: Save encryption results into NVM user codes section, 4 bytes or 32 bytes code. There are a total of 256 bytes of user codes memory mapping reserved in NVM for each Root Key.
    - Usage 3: Read a saved encrypted code from the NVM. The code will be read from the NVM and decrypted by the TOE before returning the result.

The functional block diagram is shown in the Fig.1.1 . This diagram provides a view of the chip's major functional components and core complexes.

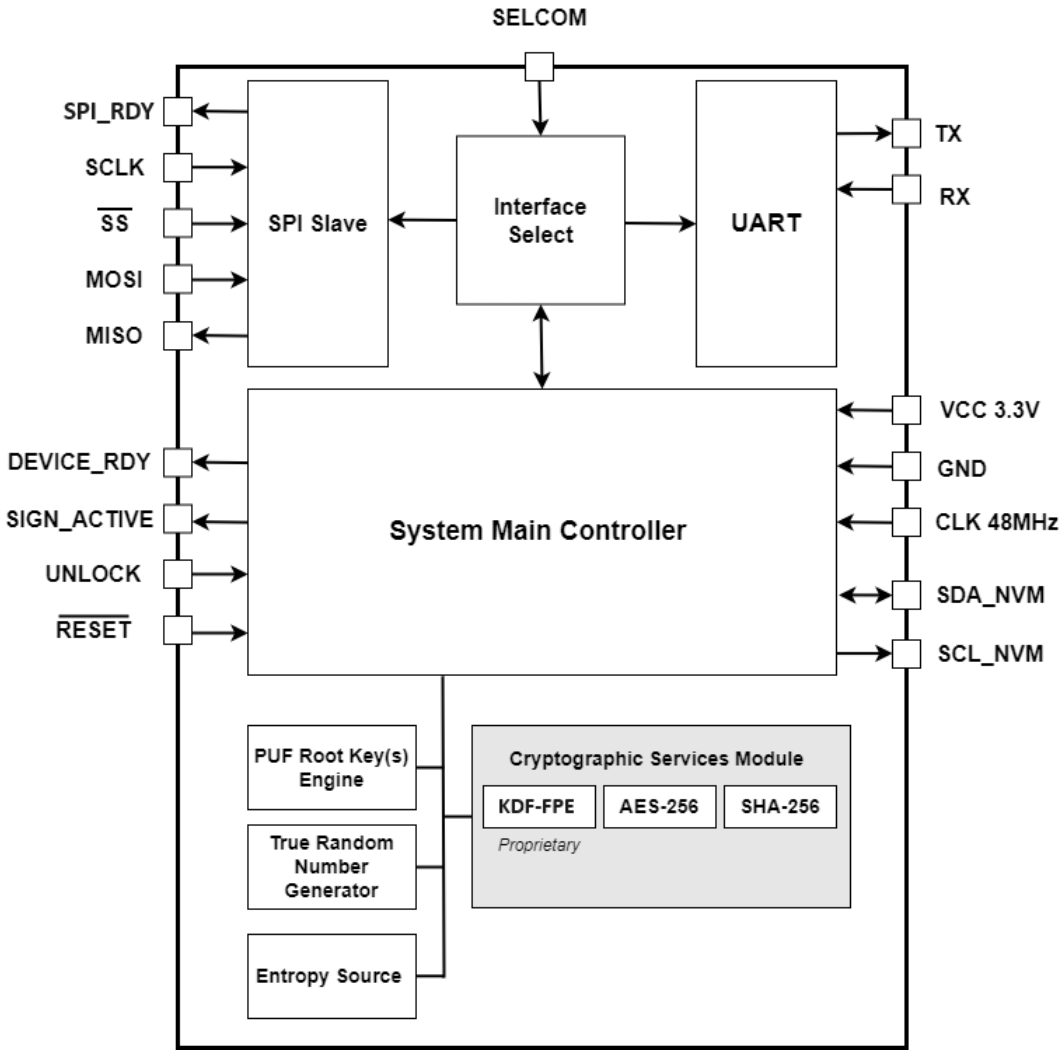

Figure 1.1: Functional Block Diagram

The TOE consists of a core security engine (SASE01EPZS) integrated circuit functioning and assembled with external non-passive elements: System clock oscillator (via CLK pin), non-volatile memory of at least 40Kbits capacity (via SDA_NVM and SCL_NVM pins) and power supply source (via VCC and GND pins).

The TOE is deterministic, in other words non-programmable, digital processor implementing a dynamic (i.e power and life cycle independent) management of multiple root identities derive from: a unique unpredictable entropy source, root keys generation and recovery mechanisms (PUF Root Key(s) Engine) and a True Random Number Generator. The term hardware root identity, or root identity refers to the set of recovered cryptographic keys. The TOE uses an external NVM to store relevant recovery data for each root identity, referred to as checkpoint data, that are only relevant when combined with the internal algorithms and entropy source confined within the TOE.

On top of the root identity management mechanisms, associated with each root identity,

the TOE supports for cryptographic primitives security functions AES-256, SHA-256,random number generation, combined with proprietary key derivation and encryption mechanism Key-Derivation Function Format Preserving Encryption (KDF-FPE).

The TOE's external non-volatile memory constitutes the persistent storage used by the TOE to store relevant checkpoint data to recover each unique root identity, and stores encrypted data to recover secure credentials. i.e root keys. The TOE associated NVM, does not contain directly relevant data that constitutes a security breach by itself and all its content are entangled with the entropy source confined within SASE01EPZS and optional user-defined inputs. In other words, the NVM component of the TOE is outside the physical TOE scope, however it should remain included for adequate description of vulnerability survey (AVA VAN.1).

The TOE, along with a clock source and NVM, is intended to be used as a secure element co-processor interfaced with standard communication interfaces: UART/SPI, selected by digital input of SASE01EPZS (via SELCOM pin). In most cases, the TOE is used with an MCU or MPU already embedding the required NVM with I2C (SDA_NVM and SCL_NVM) capabilities to be both the host and the NVM associated entity.

The TOE as a secure element platform is intended to be used to provide and safeguard cryptographic sensitive credentials, such as secret and shared private keys, to source the host with trusted credentials. The TOE is also intended to be used as a trusted execution environment to perform a set of cryptographic primitives included in the Cryptographic Services Module. The sensitive credentials and recovery data are stored hashed and/or encrypted in the NVM and can only be used, authenticated and decrypted by the TOE, SASE01EPZS. Any tampering attempts on the NVM could damage the recovery of sensitive credentials but would not leak actual secrets.

The TOE, being able to manage multiple root identities, enables the host (i.e the user) to manipulate the TOE as Secure Element as if the host system possessed multiple physically independent secure elements. In order to do so, the TOE has integrated an optional function for custom secure digital input for each root key to be entangled with, i.e each hardware identity cryptographic rules and keys entangled with a role-based like digital authentication from the connected host or master.

For example, the integrator may choose "123456" as unique volatile input to create and load Root Identity no.1, "abcdef" for Root Key no.2 and no defined input for Root Key no.3.
As a result, Root Identity no.1 and 2 will be used with role-based critical security authentication, while the Root Identity no.3 will have less security entangled layer. Root Identity no.1 and no.2 will be used for applications and functions with high risk of attacks and a large attack surface, typically networked connections, while Root Identity no.3 can be confined for local usage and internal host device operation. Each Root Identity no.1 , 2 and 3 will obtain uniquely and truly random different cryptographic behavior from the Cryptographic Services Module, and will not share correlated data stored inside the NVM.

The multiplicity of root identities and associated recovered root keys is a functional feature, not an additional security feature claimed by the TOE. The TOE does not claim to enhance

its global security level with this feature. For vulnerability assessment and side-channel attacks immunity evaluation, vulnerabilities assessed or found for a single root identity can be automatically transferred to the others identities, although tampering with multiple identities does increase the potential attack identification time.
The platform is intended to be used by an integrator that deploys it into a embedded system connected solution and/or application together with its own operating systems and/or user applications.

The main security features of the platform are listed in [3. Security Requirements and Implementation] of this document.

1.4.1. Physical Scope of the TOE

The physical scope is the SASE01EPZS as identified in Table.1.1 and whose functional blocks are identified in Fig.1.1. It includes communication interfaces with host via UART, SPI and I2C. It includes a physically unclonable function engine and cryptographic primitives, all managed by a system main controller.

1.4.2. Logical Scope of the TOE

The logical scope includes the hardware communication interfaces and GPIOs configuration options defined in the guidance documents, [2].
The communication with the host for data transactions is selected using SELCOM input to choose between UART (TX,RX,RTS,CTS) or SPI(RDY,SCLK,SS,MOSI.MISO). The communication with the NVM entity is done by I2C protocol defined in [2].
The input Reset, active low, resets the TOE to initial state and shall be implemented by the host.
Device_Rdy and Sign_active are output indicating the current TOE status. Device_rdy is pulled up once the TOE has completed its self-check in power-up. The Sign_active is pulled up when a current root key is being used and buffered inside the TOE.
The Unlock pin allows the host to remove condition constraints on creating and loading a particular root identity. It can be tight low or high or managed by the host.

1.4.3. Required non-TOE Hardware/Software/Firmware

The TOE requires certain non-TOE hardware/software/firmware to function as intended. This includes an associated non-volatile memory accessible via I2C protocol as defined in the TOE's datasheet [2], and a stable clock source input to operate up to 48MHz.

It is imperative to clarify that these non-TOE components, while essential for the TOE's operation, are not encompassed within the scope of the TOE evaluation. Consequently, the TOE reference does not guarantee or imply the security, reliability, or functionality of these non-TOE components. Users are advised to ensure these components meet their specific requirements and standards, while the TOE guarantees their non-sufficient property to expose the host or users secrets in plaintext.

Furthermore, it is important to understand the role of the data stored in the required non-TOE NVM. This data is integral for the recovery of certain cryptographic keys and parameters. However, it does not pose a risk of secrets leakage. The TOE has the capability to renew and reset this data.

In the event of an adversarial tampering with this data, the associated keys retrieved by the TOE may be lost, but they remain neither predictable nor leaked. Predicting the future behavior of the TOE and the retrieved key based on tampering with this data would necessitate predicting the TOE's internal source of entropy output. This is a complex task, given that the TOE's internal source of entropy does not directly interact with any component outside the TOE's secure boundaries.

Therefore, while the data stored in the non-TOE NVM plays a critical role in the operation of the TOE, it does not pose a risk of secrets leakage. Its integrity is important for the proper functioning of the TOE, but even if compromised, it does not allow an attacker to predict or influence the TOE's behavior or retrieved keys.

2. Security Objectives for the Operational Environment

2.1. Platform Objectives for the Operational Environment

For the platform to fulfill its security requirements, the operational environment (technical or procedural) shall fulfil the following objectives.

External Functional Components

Each instance of the TOE must be associated with a non-volatile memory instance interfaced by the I2C master protocol defined by SASE01EPZS, with power supply sources and clock signal input source. These external components are necessary for the functions of SASE01EPZS and their associated security protection described in 3.2.6.

Unique Host Control Interface

Each instance of the TOE must be associated with a single host interface via communication port, either UART and SPI, and shall not be shared by multiple host on a shared bus, so to ensure integrity and efficiency of claims described in [3. Security Requirements and Implementation].

3. Security Requirements and Implementation

3.1. Security Assurance Requirements

The claimed assurance requirements package is: SESIP1 as defined in [1]. The assurance requirements are as follows:

Table 3.1 SESIP1 Assurance Requirements

Assurance Class	Assurance Families
ASE: Security Target Evaluation	ASE_INT.1 ST Introduction ASE_OBJ.1 Security requirements for the operational environment ASE_REQ.3 Listed security requirements ASE_TSS.1 TOE summary specification
AGD: Guidance documents	AGD_OPE.1 Operational user guidance AGD_PRE.1 Preparative procedures
ALC: Life-cycle Support	ALC_FLR.2 Flaw reporting procedures
AVA: Vulnerability Assessment	AVA_VAN.1 Vulnerability survey

3.1.1. Flaw Reporting Procedure (ALC_FLR.2)

In accordance with the requirement for flaw reporting procedures (ALC_FLR.2), the developer has defined the following procedure that consist in 3 or 5 steps according to the case:

Step 1: Flaw reporting

A dedicated team is in charge of monitoring via e-mail (service@iist.com.tw) the IIST products-related reported bugs and flaws by customers. Every flaw is then reported to the product development team for analysis.

Step 2: Flaw Analysis

Each reported flaw is analyzed, evaluated and categorized according to its nature and origin. After risks evaluation and identification of associated threats for the customer products, decisions are made based on feasibility and necessity: a bug fix will be produced, an update version of the product is necessary or the product has to be recalled.

When the hardware design of a product, such as SASE01EPZS, is concerned by a security flaw, it cannot be related to a firmware that can be updated, in that case a design flaw report will lead to the need of a new product design in step 3, should the flaw be found critical. In the eventuality that the flaw is found non-critical, it is kept as a record for future products and adequate communication with customers is arranged.

In other cases, with flaws related to any software development kit, user manual, or documentation, the solution will be provided at step 3.

Step 3: Design Solution

The step design solution corresponds to the activity of the design team in creating a fix for an identified bug, expected to be related to instruction manual or software development kit. The product design team may also at this step updates the design of the product to deliver a fixed version to customers.

Step 4: Solution Delivery and Update

Once a solution to the flaw reported is found, the solution, if applicable, is delivered and a verification procedure is launched with the customer to close the opened flaw reporting ticket and validate the new solution.

Step 5: Customers Support and Communication

The customer reporting the flaw, receives updates throughout all the necessary steps to treat flaw reporting: the reported flaw summary, the results of the flaw analysis and the decision made to answer it, the solution put in place and expected lead time, and finally the solution delivery and verification report.

3.1.2. Vulnerability Survey (AVA_VAN.1)

In accordance with the requirement for a vulnerability analysis survey (AVA_VAN.1) the developer has performed a vulnerability survey and submits the following test results to demonstrate the consideration of publicized potential vulnerabilities relating to the TOE:
The secure cryptographic engine, isolation mechanism, debugger interface, and device lifecycle management mechanism components of the platform were analyzed to determine their vulnerability and attack potential. Side channel attacks, fault injection, abuse of the debugger interface, RNG attacks, and physical extraction of device information by decapping the device were considered and deeemed to have an attack rating of 21 or greater. Attacks that are within scope for the SESIP1 assurance level are those attacks that have an attack rating of 0-15. Therefore, these attacks are out of scope for this evaluation.
Also, public databases were searched for publicly known vulnerabilities:

- Common Vulnerabilities and Exposures
  (CVE, https://cve.mitre.org)
- Common Weakness Enumeration
  (CWE, https://cwe.mitre.org)
- Common Attack Pattern Enumeration and Classification
  (CAPEC, http://capec.mitre.org/data/index.html)

No vulnerabilities were found.

3.2. Security Functional Requirements

3.2.1. Verification of Platform Identity

The platform provides a unique identification of the platform type, including all its parts and their versions.

Conformance Rationale:

The platform contains a 32 bytes default input values fixed by design, referred to as default token that can be used in conjunction with the validation of certain TOE cryptographic functions to verify the platform identity through usage of regular operating functions. Each specific TOE hardware version is associated a different 32 bytes default token to ensure version tracking. Each lot and production information is indicated on the TOE package.
There will be only one 32 bytes default token per TOE version that can exhibit specific cryptographic properties and serve as platform identity verification process for a specific TOE version. The procedure is described below.
The platform is able to provide, after manufacturing and associated with an NVM, means to extract unique identifiers through SASE01EPZS functions. SASE01EPZS on the first power-up, will attempts to generate first 5 sets of application root keys by default for 5 hardware root identities, which can be used for initial identification and identification tracking mechanism, should the root identities be updated at a later time of the product life cycle.
Each of the initial set is created by entangling a unique PUF response of 798 bits strength, a random bit generator and a default arbitrary random value referred to as token of 256 bits length. The default token is hard-wired in registers constitutes a mean to verify the platform identity.
The root identity generation procedure can be used to identify SASE01EPZS platform. The root identities generation procedure allows the host to add an optional additional input sequence, input token, of up to 32 bytes to replace the default hard-wired token value and entangle it with the hardware identity to further increase the security level. The replacement of the default token by a user-defined input undergoes a shift-register like function and does not undergo any cryptographic transformation.
Should the host decides not to use this additional input option, SASE01EPZS will use the default 32 bytes sequences pre-registered in the hardware internal registers. Therefore, there exists a unique 32 bytes digital sequence, common to all SASE01EPZS platforms from the same IC mask, that will make the platform able to successfully generate and retrieve the same target root identity whether that specific 32 bytes input is used or not.
This specific 32 bytes sequence default token pre-registered into an IC mask or IC design block is provided under non-disclosure agreement to each integrator of the platform.

Performing the following test sequence serves as platform identification during the first power-up:

1. First power-up: Generate 5 sets of root identities set without optional additional input, referred to as token in [2].
2. Perform Start command without token, on one of the set of application root keys, according to [2].

3. Perform unique response generation with any defined input 32 bytes seed sequence.
4. Perform Flush command, according to [2]
5. Perform Perform Start command with the specified token on the same target root identity, according to [2].
6. Repeat step 3 with same the same conditions and verify that the unique response is identical to the one obtained for step 3.

If step 3 and step 6 generate the same unique key response, the platform identification process is successful.

Performing the following test sequence serves as platform identification at any moment during product lifecycle:

1. Select an indexed set of application root keys that can be renewed (i.e overwritten) without damaging the final product functions
2. Send the adequate Start command for the target root identity
3. Renew the target root identity without input token option.
4. Perform Start command without token input, on the target root identity, according to [2].
5. Perform unique response generation with any defined input 32 bytes seed sequence.
6. Perform Flush command, according to [2]
7. Perform Perform Start command with the specified unique token on the same target root identity, according to [2].
8. Repeat step 5 with same the same conditions and verify that the unique response is identical to the one obtained for step 5.

If step 5 and step 8 generate the same unique key response, the platform identification process is successful.

The mathematical creation of each hardware root identity relies on a true random number generator and unique PUF response for every different instances of the TOE.

The uniqueness of each root identity is statistically guaranteed with true random bits generator meeting the requirements defined by [6] and a PUF entropy source response characteristics meeting requirements defined by [10] for bits distribution, inter and intra chips hamming distances.

The default 32 bytes token value used when the host generate a new hardware root identity without specifying its own input token, is a fixed sequence written in non-volatile hardware registers that takes effect in a deterministic manner, previously validated by wafer-level testing on each die.

3.2.2. ~~Secure Update of Platform~~

~~The platform can be updated to a newer version in the field such that the integrity, authenticity and confidentiality of the platform is maintained.~~

The platform does not contain any memory or firmware, it does not support the update or patching of hardware nor internal boot firmware. It does offer features that enable a customer to implement secure update mechanisms for their own code and platform.

Conformance Rationale:

SASE01EPZS TOE is a deterministic digital signal processing chip without executed firmware, it is solely composed by hardware designs and enable the creation and use of unique volatile cryptography rules when connected to an accessible non-volatile memory.

3.2.3. Cryptographic Operation

The platform provides the application with the cryptographic operations from Table 3.2 functionality with the specified algorithms listed in Table 3.2 as specified in relevant specifications shown in Table 3.2 for key lengths listed in Table 3.2 and modes listed in Table 3.2.

Table 3.2 List of cryptographic algorithms used for cryptographic operations

Operations	Algorithm	Specification	Key(s) length	Modes
Unique Response	KDF-FPE	*Block Size 256 bits	1024 bits	Proprietary
	SHA-256	FIPS PUB 180-2	N/A	Unsalted
Encryption, Decryption	KDF-FPE	*Block Size 256 bits	1024 bits	Proprietary
	AES-256	NIST FIPS 197	256 bits	OFB

Conformance Rationale:

The proprietary function KDF-FPE, although mathematically a cryptographic operation, has not been considered for the vulnerability analysis due its closed-source nature. For vulnerability analysis, the KDF-FPE function is treated as a plaintext function.
The platform offers a proprietary hash-based key derivation function that generates 256-bit cryptographic unique responses based on user commands. This function utilizes a proprietary key-derivation function format-preserving encryption (KDF-FPE) algorithm in conjunction with SHA-256 (SHA-2), as specified in [7].

The platform provides functions for encrypting and decrypting data stream sequences using the combination of KDF-FPE and AES-256 OFB, as specified in [8]. These operations utilize an AES secret encryption key derived internally from the 1024 bits KDF-FPE secret key.

The combination of two type of ciphering algorithms reinforce the robustness against attacks, having two mixed encryption ciphers to decipher with two different secret keys to extract.

The proprietary KDF-FPE algorithm uses a 1024 bits keys generated by random bits generator, recoverable using unique ciphertext data inside the NVM checkpoint data, the associated hardware identity and optional user-defined input token.
The KDF-FPE used in the TOE respects a proprietary implementation including a Key Derivation Function (KDF) mechanism and a Format Preserving Encryption mechanism. To ensure standardized security compliance, it is coupled with a NIST standardized security algorithms for cryptographic operations.

The output of KDF-FPE algorithm has been made and verified so that its output statistically validates the true randomness behavior defined by [6] and can be considered or extrapolated as a pseudo-random number generator mechanism where the input seeds are the input keys.

To ensure the reliability and conformity of the SHA-256 implementation, it has been designed and tested in accordance with the Cryptographic Algorithm Validation Program (CAVP) NIST requirements for SHA-2 [7].

During the manufacturing process, the deterministic cryptographic functions from KDF-FPE undergo chip deterministic testing. This testing is performed in a chip test mode with known parameters, ensuring the function's performance and adherence to specifications.

The rigorous testing approach guarantees the functionality and conformance of the platform's key wrapping, encryption and derivation functions.

To ensure the reliability and conformity of the AES-256 OFB implementation, it has been designed and tested in accordance with the Cryptographic Algorithm Validation Program (CAVP) NIST requirements for AES.

During the manufacturing process, the AES-256 OFB encryption and decryption functions undergo chip deterministic testing. This testing is performed in a chip test mode with known parameters, ensuring the functions' performance and adherence to specifications.
Additionally, the validity of the encryption key is derived from the successful validation of random numbers and hardware identities. This comprehensive testing approach guarantees the functionality and conformance of the platform's AES-256 OFB encryption and decryption functions.

3.2.4. Cryptographic Key Generation

The platform provides the application with a way to generate cryptographic keys for use in AES as specified in NIST FIPS PUB 197 [8] for key lengths 256 bits, and in proprietary KDF-FPE for key lengths 1024 bits.

Conformance Rationale:

SASE01EPZS provides multiple hardware identities, each associated with application root keys. These identities are generated based on the dynamic measurement of the entropy source,

user-defined role-based like authentication input, and PUF recovery engine algorithms, with acceptance criteria as specified in [6]. Each application root key undergoes internal verification and authentication using a proprietary Key Derivation Function Format Preserving Encryption (KDF-FPE) algorithm, performing a symmetrical encryption algorithm with a hash function.

The security keys and functions of the platform rely on a physically unclonable function (PUF) entropy source designed to meet the recommended statistical behavior as defined in [10]. This entropy source design renders any physical tampering destructive without user information or secret leakage, and non-invasive power and electromagnetic radiation analysis inconsistent.
To perform the cryptographic functions described in 3.2.3, the TOE generates two application cryptographic keys, the second one derived from the first one. A first 1024 bits KDF-FPE cryptographic key is created using random bits generator. A second 256 bits key for AES-256 OFB operations is derived using the proprietary KDF. The first 1024 bits KDF-FPE cryptographic key is recovered using externally stored checkpoint data, their respective stored data properties are further described in 3.2.6.

Table 3.3 List of cryptographic algorithms used for key generation

Key Label	Generation Algorithm	Specification (*Proprietary)	Operations Algorithms	Key Length	Modes
Application Root Key 1	Random Bit Generator	NIST SP800-22	KDF-FPE	1024 bits	Re-seeded
Application Root Key 2	KDF	*Block Size 256 bits	AES-256 OFB	256 bits	Proprietary

The TOE uses a PUF mechanism to generate and recover cryptographic keys, therefore for better clarity in the TOE claims, the conformance rationale describes the intermediary values generated and collected that can be considered as keys and that take active part in the generation of the final two hardware identity application root keys.

A initial raw PUF entropy source response is collected. The response is a binary array with a certain level of high entropy to generate random bits and a certain level of low entropy to use as a stable unique base to further generate recoverable secrets. The PUF entropy source characteristics were validated according to recommended characteristics described in [10].

The initial PUF entropy source response is then randomly shuffled using a shuffle function and a shuffle seed generated by random bit generator. The random bit generator statistical output validated against [6] and [5] non-iid statistical test suite. The shuffle seed is then kept within the checkpoint data associated with that specific shuffling, i.e that specific hardware root identity.

The shuffled PUF response is used as input of the proprietary KDF function to create an

intermediary FPE encryption key of the same bit size. This intermediary FPE encryption key is used to encrypt 1024 random bits kept in checkpoint data. The decryption result of this 1024 random bits provide the final value of the 1024 random bits application root key. Without disclosing the exact detailed characteristics of the PUF response bit length, the TOE guarantees a PUF response length above 512 bits.

To ensure sufficient level of data security, the output of the proprietary KDF and FPE functions were also tested against [6] to ensure enough randomness to prevent brute-force decryption. In terms of actual application 1024 bits root key generation, it uses a random bit generator meeting the specifications requirements defined by [6]. Finally the second used application root key for AES-256 OFB encryption is derived from the 1024 bits random key using the proprietary KDF section of the KDF-FPE algorithm.

Since the KDF-FPE algorithms are not part of NIST standardized algorithms suite, ensuring compliance with [6] is essential. It can be assimilable to a pseudo-random generator or to a shuffling data engine. Ultimately, the keys generated for cryptographic operations are always using a [6] compliant random bits generator.

3.2.5. Cryptographic Random Number Generation

The platform provides the application with a way based on physical noise to generate random numbers to as specified in SP800-90B[5] and SP800-22 [6].

Conformance Rationale:

The platform provides function of generating true random numbers based on the integrated dynamically measurable entropy source as specified in [5].The platform provides true random numbers compliant with the statistical randomness tests as specified in [5] and [6].

The PUF entropy source response expected entropy according to non-iid NIST SP800-90B [5] is greater than 0.79.

The PUF entropy source raw number of output entropy is greater or equal than 325.

The platform adheres to the SP800-90B [5] standard noise source description for generating necessary entropy to generate true random numbers with a lowest average of 0.8 entropy value measured and estimated according to statistical non-iid tests from [5] for a set of physical IC samples exposing relevant measurable outputs to ensure the target subsystem is compliant for production.

For security purposes and to keep the source of trusted entropy generator exclusively confined within the physical boundaries of the TOE, the entropy value cannot be physically estimated with production ICs. The generated true random numbers undergo compliance testing against statistical randomness criteria outlined in SP800-22 [6].

The PUF entropy result from IC testing sampling have not had an entropy value measured

below 0.8, as per the SP800-90B non-IID test method. Moreover, the PUF response characteristics adheres to the recommendations guidelines provided by [10].

After manufacturing process and production, in normal operation, generated true random numbers can be tested against statistical randomness criteria outlined in [6] by using the regular true random number generator function provided by the TOE as operations function.

3.2.6. Secure External Storage

The platform ensures that all data stored outside the direct control of the platform, except for user-defined input tokens, is protected such that the confidentiality and binding to the platform instance is ensured.

Conformance Rationale:

The TOE requires externally stored, generated 700-byte checkpoint data per root identity version created in order to recover and decrypt stored credentials necessary for the TOE operations for the target root identity.

This 700-byte data block, considered a single asset, includes the application root key in its 1024-bit encrypted form. The TOE ensures that this data, stored outside its direct control, maintains authenticity, integrity, confidentiality, and binding to the platform instance.

The protection and encryption of the data stored externally reuses the cryptographic operations described in 3.2.3.
Parts of the 700-bytes checkpoint are random numbers, for instance the PUF entropy source shuffling seeds, and they can be considered as data stored in plaintext.

Encryption: The 700-byte checkpoint data, including the 1024-bit secret application root key, is encrypted using strong cryptographic algorithms before being shared and stored outside the TOE. This ensures that the data remains confidential and cannot be accessed by unauthorized entities.

Key Management: The keys and values required to unwrap and decrypt the 1024-bit application root key are securely managed within the TOE. These keys retrieved upon correct provided checkpoint data never leave the TOE, ensuring that the host or application cannot decrypt the data outside the TOE.

Integrity Verification: The TOE requires exact, unmodified checkpoint data to recover the application root keys. Any modification to the data by the application will prevent the recovery of the root keys, thus maintaining data integrity. Any modification of the 700-byte checkpoint data will result in a recovery mechanism mismatch and all operations will be rejected.

Platform Binding: The checkpoint data is bound to the specific platform unique PUF entropy source instance, ensuring that it cannot be used on a different instance. This binding ensures that the data is authentic and specific to the intended platform.

Response Mechanism: If the data is tampered with or destroyed, the TOE will not process the data, effectively protecting the integrity and confidentiality of the cryptographic operations.

Impact of Data Destruction: While the application can modify or erase the checkpoint data, this action results in the loss of means to recover the unciphered data, not in the exposure of

the actual value of the asset. This ensures that the authenticity, integrity, and confidentiality of the data are not compromised. The destruction of data by the application will prevent the TOE from performing cryptographic operations, but it will not lead to any predictable or malicious behavior. The TOE will simply reject the invalid data.

Important Note: The TOE datasheet document [2] does mention operating functions to save user code and read user code. However, the labeling save and read has been chosen for reading convenience. From a security assessment perspective and cryptographic operations it should not be considered different than encrypt and decrypt operations functions described in 3.2.3 as it in fact simply use I2C port instead of UART/SPI to output the results.

4. Mapping and sufficiency rationales

4.1. SESIP1 sufficiency

Table 4.1 SESIP1 Sufficiency

Assurance Class	Assurance Family	Covered by	Rationale
ASE: Security Target Evaluation	ASE_INT.1 ST Introduction	Section 1	The ST reference is in section 1.1, the TOE reference in section 1.2, the TOE overview and description in section 1.4.
	ASE_OBJ.1 Security requirements for the operational environment	Section 2	The objectives for the operational environment in Section 2 refer to the guidance document.
	ASE_REQ.3 Listed security requirements	Section 3.1	All SFRs in this ST are taken from [1]. SFR "verification of platform identity" is included. SFR "Secure update of platform" is mentioned in Section 3.
	ASE_TSS.1 TOE summary specification	Section 3	All SFRs are listed per definition, and for each SFR, the implementation and verification is defined in the SFR.
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Section 1.3	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement.
ALC: Life-cycle Support	ALC_FLR.2 Flaw reporting procedures	Section 3.1.1	The flaw reporting and remediation procedure is described.
AVA: Vulnerability Assessment	AVA_VAN.1 Vulnerability survey	Section 3.1.2	The vulnerability survey and associated test results are described

5. Developer's Deliverables and Procedures

The delivery procedures provided by the developer outline the means to ensure that SASE01EPZS remains secure throughout the shipping and delivery process or further integrated products. These include:

- **Traced Packaging and handling** of SASE01EPZS to prevent tampering or damage during transit.
- **Secure private delivery channels** that ensure SASE01EPZS is not modified or replaced during shipment.
- **Documentation** to assist the user in verifying that SASE01EPZS is an evaluated instance and has not been altered or masqueraded.

While these procedures are outside the control of the user, adherence to them ensures that the delivered SASE01EPZS remains trustworthy.
List of deliverables associated with SASE01EPZS TOE version 2.5:

- SASE01EPZS IC reel (*Integrated Circuit*)
- SASE01EPZS Datasheet (*Documentation*)
- SASE01EPZS Development Kit and User Manual (*Documentation*)
- SASE01EPZS SDK support for integration and verification of TOE (*Software and reference codes*)

6. References

6.1. Evaluation Documents

[1] Security Evaluation Standard for IoT Platforms (SESIP) Version 1.1

6.2. Developer Documents

[2] SASE01EPZS Datasheet Rev. 2.4

[3] SASE01EPZS Instruction Manual Rev. 1.0

[4] Two-wire Serial EEPROMs Datasheet, AT24C128 AT24C256, Atmel, https://ww1.microchip.com/downloads/en/devicedoc/doc0670.pdf, last access November 21, 2024.

6.3. Standards

[5] NIST Special Publication SP800-90B: Recommendation for the Entropy Sources Used for Random Bit Generation,US Department of Commerce/National Institute of Standards and Technology, January 2018.

[6] NIST Special Publication 800-22r1: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, US Department of Commerce/-National Institute of Standards and Technology, April 2010.

[7] FIPS PUB 180-2: Secure Hash Standard, Federal Information Processing Standards Publication 180-2, U.S. Department of Commerce, National Institute of Standards and Technology , Information Technology Laboratory, February 2004.

[8] FIPS 197: Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, US Department of Commerce/National Institute of Standards and Technology, May 2023.

[9] FIPS PUB 140-3: Security Requirements for Cryptographic Modules, Federal Information Processing Standards Publication 140-3, US Department of Commerce/National Institute of Standards and Technology, March 2019.

[10] ISO/IEC 20897-2:2022(E): Information security, cybersecurity and privacy protection — Physically unclonable functions —, International Standard, May 2022

List of Figures

List of Tables

Revision History

Version	Date	Content	Author
1.0	August, 2023	Initial draft version for scheme discussion	IIST Inc.
2.0	December,2023	Update TOE description with Clock,NVM described as external components	IIST Inc.
2.1	January, 2024	Change "Verification of platform identity" 3.2.1 Update "SESIP1 sufficiency" 4.1	IIST Inc.
2.2	April, 2024	Update Guidance document version Rev. 2.2	IIST Inc.
2.3	June, 2024	Update conformance rationales	IIST Inc.
2.4	July, 2024	Proofread and Update conformance rationales	IIST Inc.
2.5	November 21, 2024	SFRs and Developer's Deliverables edited	IIST Inc.