QuiddiKey 300
v1.0

Security Target

Doc version **1.3**
Status **Approved**
Reference **IID-QK300-1-0-PSA-ST**

www.intrinsic-id.com

This product is subject to EU export restrictions according to Council Regulation (EC) No. 428/2009, dual-use control category 5E002.

This document contains information which is proprietary and confidential to Intrinsic ID B.V. and is intended for internal use only. The document is provided with the express understanding that the recipient will not divulge its content to other parties or otherwise misappropriate the information contained herein. Please destroy this document if you are not the intended recipient. Thank you.

History Information

Version	Date	Change Description
1.3	2023-07-05	• Updated QuiddiKey version to v1.0.2 • Updated version numbers of referenced documents
1.2	2023-06-09	• Updated QuiddiKey version to v1.0.1 • Updated version numbers of referenced documents
1.1	2023-05-08	• Moved QuiddiKey driver out of scope for the evaluation • Added Cryptographic Key Generation to list of SFRs in Section 1.6.3 • Updated description for SFR Secure Update • Updated format for SFRs secure communication • Fixed some typos and added minor clarifications here and there
1.0	2023-03-30	First approved release

Table of Contents

1. Introduction

The objective of this Security Target (ST) document is to state the security claims of the hardware IP **QuiddiKey 300 v1.0** as a PSA RoT Component on Level 3.

The PSA website states [PSA-RoTComponent]:

"… we recognize that Root of Trust (RoT) components, offered by IP providers, are a crucial part of the security story. Over time we've introduced a scalable approach for IP providers who invest significant time and resources into security. We enable you to showcase product security expertise to your customers, with a stamp of trust. Meaning that your customers can fast-track full PSA Certified chip evaluations by reusing your existing certifications."

In line with this philosophy, the scope of this ST document is to present the security properties and functionality of the Intrinsic ID hardware IP component QuiddiKey 300 v1.0 for certification as a **PSA RoT Component**. This ST document states the security objectives, security assurance requirements and security functional requirements of QuiddiKey 300 v1.0 as an RoT component, and presents pointers to the details of its implementation to assist the evaluation.

The PSA RoT Component under evaluation refers to the QuiddiKey 300 v1.0 HW IP block (QK IP), namely the gate-level netlist description of the QK IP, or an equivalent representation or implementation thereof.

For the sake of facilitating the evaluation, the evaluated QK IP is integrated in a basic SoC target reference platform based on a commercial off-the-shelf AMD/Xilinx Zynq XC7Z020 FPGA [AMD-XC7Z020] on a commercial development board [AMD-ArtyZ7-20]. This target reference platform is not part of the certification, but it represents an exemplary integration and use of the evaluated RoT component following Intrinsic ID additional documentation, driver software and guidelines.

The Security Target describes the Platform (in this chapter) and the exact security properties of the Platform that are evaluated against **[PSA-SESIP-L3-RoTComponent]** (in chapter "Security requirements and implementation") that a potential consumer can rely upon the product upholding if they fulfil the objectives for the environment (in chapter "Security Objectives for the operational environment").

1.1. ST Reference

See title page.

1.2. SESIP Profile Reference

PP name	*SESIP Profile for PSA Certified RoT Component Level 3*
PP version	*01 (JSADEN018 1.0 REL)*
Assurance Claim	*SESIP Assurance Level 3 (SESIP 3)*
Optional and additional SFRs	*Additional SFRs (mandatory for Trusted Subsystem platform):* • *Secure Communication Support* • *Secure Communication Enforcement*

Table 1: SESIP Profile Reference

1.3. Platform Reference

Platform name	*QuiddiKey 300*
Platform version	*v1.0*
Platform identification	*HW IP module:* ***QK353-1.0.2-apb*** *(order number as defined in Section 1.5 of* [IID-QK300-1-0-DS]*)*
Platform Type	*Digital HW IP module as a "PSA RoT Component"*

Table 2: Platform (RoT Component) Reference

1.4. Included Guidance Documents

The following documents are delivered with the Component:

Reference	Name	Version
[IID-QK300-1-0-DS]	QuiddiKey 300 v1.0 Datasheet	1.3
[IID-QK300-1-0-IM]	QuiddiKey 300 v1.0 Integration Manual	1.1
[IID-QK300-1-0-DRV-RM]	QuiddiKey 300 v1.0 Driver Reference Manual	1.2
[IID-QK300-1-0-CCR]	QuiddiKey 300 v1.0 Coverage Reports	1.1
[IID-QK300-1-0-RN]	QuiddiKey 300 v1.0 Release Notes	1.2
[IID-AN102]	Application Note 102: QuiddiKey Use Case: Embedded Key Vault	1.0
[IID-AN100]	Application Note 100: Multiple Key Generation with QuiddiKey	1.0
[IID-QK300-1-0-SG]	QuiddiKey 300 v1.0 Security Guidance	1.2
[IID-QK300-1-0-SD]	QuiddiKey 300 v1.0 Security Documentation	1.2
[IID-QK300-1-0-ACVP]	QuiddiKey 300 v1.0 ACVP parameters document to support functional conforming testing.	1.1
[IID-ESVG-NIST90B]	SRAM PUF-based Entropy Source, validation Guide for NIST SP800-90B	1.0

[IID-QK300-1-0-ESVGA]	QuiddiKey 300 v1.0 Entropy Source Validation Annex (Annex to the "SRAM PUF-based Entropy Source – Validation Guide for NIST SP800-90B")	1.0

Table 3: Guidance Documents

1.5. Other Certification

1.5.1. NIST CAVP

The immediate predecessor of the platform (QuiddiKey v4.1) has been evaluated in the Cryptographic Algorithm Validation Program (CAVP) by NIST. The evaluated algorithms have not been changed in the current platform (QuiddiKey 300 v1.0), and all provisions for their (renewed) CAVP validation and certification are in place [IID-QK300-1-0-ACVP].

Scheme	CAVP Cryptographic Algorithm Validation Program
Certification body	NIST
Certification number	A2516 https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/details?product=15005
Certificate date	First validated:4/11/2022
Implementation name	QuiddiKey
Version	QK_RELEASES/v4.1.0
Description	Secure key generation and storage IP module based on SRAM PUF. Extracts device-unique cryptographic keys from the chip's unique silicon properties through a connected SRAM memory. Providing secure key vault functionality (key wrapping/unwrapping with device unique keys) without the need for embedded non-volatile memory.
Operating Environment	Xilinx Zynq XC7Z020 [AMD-XC7Z020]
Tested algorithms	AES-CTR, HMAC DRBG, HMAC-SHA2-256, KDF SP800-108, SHA2-256

Table 4: QK IP version 4.1 CAVP certification

1.6. Platform Functional Overview and Description

1.6.1. Platform type

The considered platform is a RoT Component as intended by PSA [PSA-SESIP-L3-RoTComponent].

1.6.1.1. RoT Component Type

The platform is **QuiddiKey 300 v1.0**, a HW IP component (delivered as gate level netlist to a customer), which is targeted for certification as a **PSA RoT Component**, on Level 3. The use of the QK IP can be supported by the QuiddiKey 300 SW driver of the IP block in SoC or MCU integrations (not in scope).

QuiddiKey is a HW security subsystem for secure key generation, storage and management, and for secure random-number generation. QuiddiKey also offers cryptographic engine functionality in the form of wrapping/unwrapping of externally provided secrets.

1.6.1.2. RoT Component Integration

As a demonstration of the typical integration and use of the QK IP, and for assisting the evaluation, the QK IP is preintegrated on an FPGA-board with an embedded microprocessor (a COTS development board [AMD-ArtyZ7-20] with FPGA chip Xilinx Zynq XC7Z020 FPGA board [AMD-XC7Z020]). An example application including the QuiddiKey driver is preinstalled on the microprocessor. Using a serial communication tool on a general-purpose PC system, the evaluator can interface with this application to demonstrate the functionality of the QK IP.

As a RoT component, the QK IP is intended to be integrated by the customer/integrator as an immutable (HW) RoT into security architecture of their platform, i.e., it cannot be modified after manufacturing. The QK IP is integrated as a module in the HW design of the customer/integrator (e.g., an MCU or SoC). The customer/integrator (e.g., chip designer, device maker) is responsible for the correct and secure integration of the QK IP into its system, which shall comply with the requirements detailed in the QuiddiKey Integration Manual [IID-QK300-1-0-IM] and Security Guidance [IID-QK300-1-0-SG].

1.6.2. Physical Scope

1.6.2.1. Description of the QuiddiKey Component

The QuiddiKey HW product is a digital HW IP module delivered to a customer as an RTL netlist, which is a synthesized version of a VHDL HW description. As shown in Figure 1, the top-level QuiddiKey module physically contains the QuiddiKey Engine, the QuiddiKey register interface, and a protocol converter for connecting the register interface to a standard system bus.

1.6.2.2. Physical Interfaces of the QK IP

Figure 1 shows the top-level block diagram of the QK IP component and its typical interconnection with the rest of the system when it is integrated in an SoC or MCU. The physical interfaces to QK IP are all digital HW input and/or output signals which need to be connected physically into the integrator’s HW platform. When integrated in an SoC/MCU, a number of these physical interfaces can be made accessible in the SW environment of the platform, preferably through the use of the QuiddiKey driver library (not in scope).

A very typical integration of QuiddiKey is as a HW IP block in an MCU system with a standard system bus. QuiddiKey can be accessed from the MCU’s SW environment executing on the central microcontroller (CPU) through its memory-mapped register interface.

The full physical details of all these interfaces are presented in Section 2 of the QuiddiKey datasheet [IID-QK300-1-0-DS]. The details on how to properly connect these interfaces in a

platform are presented in Section 2.1 of the QuiddiKey integration manual [IID-QK300-1-0-IM]. A brief overview:

1. **PUF SRAM** interface: a dedicated digital memory interface used to communicate directly with the (dedicated) PUF SRAM which should be part of the trusted immutable subsystem.
2. **EC SRAM** interface: an (optional) digital memory interface used by QuiddiKey to automatically add error detection and correction protection on the data stored in PUF SRAM.
3. **QuiddiKey register interface**: a slave register interface for data transfer, control commands and status information; through an included protocol converter, this register interface can be connected to a standard bus interface of an SoC/MCU system (supported: APB or TileLink) and becomes accessible to the system's CPU in a memory-mapped interface.
4. **Secure Output interface**: a dedicated HW signal interface to securely transfer key or random data directly from QuiddiKey to a HW security or crypto module; this eliminates the need to transfer this data via a potentially untrusted shared bus.
5. **Secure Input interface**: dedicated interface to (optionally) transfer external random entropy to the DRBG during DRBG reseed.
6. **Interrupt interface**: signal interface which indicates events (command finished, error occurred, etc.) to the interrupt controller.
7. **DMA interface**: signal interface which indicates events (request input data, request output data) to the DMA Engine.
8. **Other** (not shown in Figure 1): several direct signal interfaces: module reset and clock, (static) configuration of QuiddiKey, direct zeroization and mode selection.

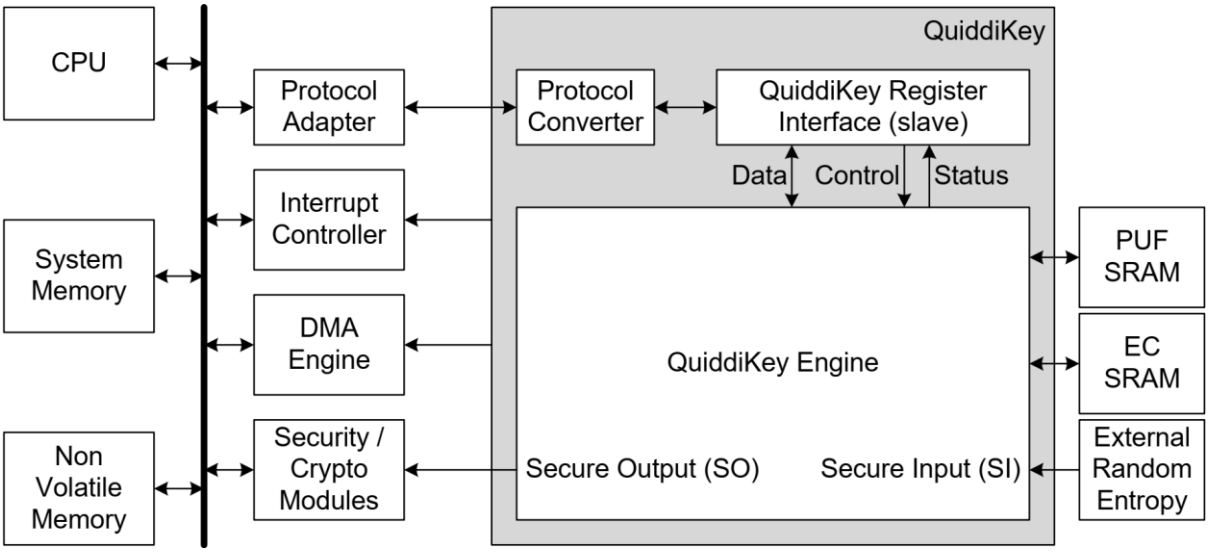

Figure 1: QuiddiKey integrated in microcontroller system

1.6.3. Usage and Major Security Features

QuiddiKey takes an SRAM PUF response, a noisy silicon fingerprint[1], as input and turns it into a high-quality and secure device-intrinsic key by further processing. QuiddiKey reliably reconstructs the same device-intrinsic key under a wide range of environmental circumstances. It generates an Activation Code (AC) which, in combination with the SRAM start-up behaviour (SRAM PUF response), is used to reconstruct on demand and in real time the device-intrinsic key which is never stored permanently. When the key is no longer needed, it can be removed from working memory. When it is needed later it can be reconstructed.

The device-intrinsic PUF key is used as the root key for cryptographic key derivation of unique application keys, and for the derivation of wrapping keys used for the secure cryptographic wrapping of external application secrets (secure storage of other keys). A key derived or wrapped by QuiddiKey can only be retrieved on the same device and is completely inaccessible on other devices. In addition, QuiddiKey also uses the entropy extracted from the SRAM PUF noise behaviour to seed a cryptographically secure random bit generator.

Based on this description, the following security functional requirements are considered in the scope of evaluation for this platform:

- **Verification of Platform Identity**
- **Verification of Platform Instance Identity**
- **Cryptographic Operation**
- **Cryptographic Random Number Generation**
- **Cryptographic Key Generation**
- **Cryptographic KeyStore**
- **Physical Attacker Resistance**

In addition, the following additional security functional requirements are considered as mandatory for the platform as a RoT trusted subsystem:

- **Secure Communication Support**
- **Secure Communication Enforcement**

1.6.4. Logical Scope

Figure 2 reiterates the logical scope hierarchy of a PSA evaluation. In a typical and correct integration, the QK IP will reside in the **Trusted Subsystem** part of this hierarchy: *"Any*

[1] SRAM Physical Unclonable Functions or SRAM PUFs use the behaviour of standard SRAM, available in any digital chip, to differentiate chips from each other. They are virtually impossible to duplicate or predict. Due to deep sub-micron process variations in the production process, every transistor in an SRAM cell has slightly random electrical properties. This randomness is expressed in the start-up values of uninitialized SRAM. These values form a unique chip fingerprint, called the SRAM PUF response.

Trusted subsystems that the host processor relies on for protection of its assets, or that implement some of its services." [PSA-SESIP-L3].

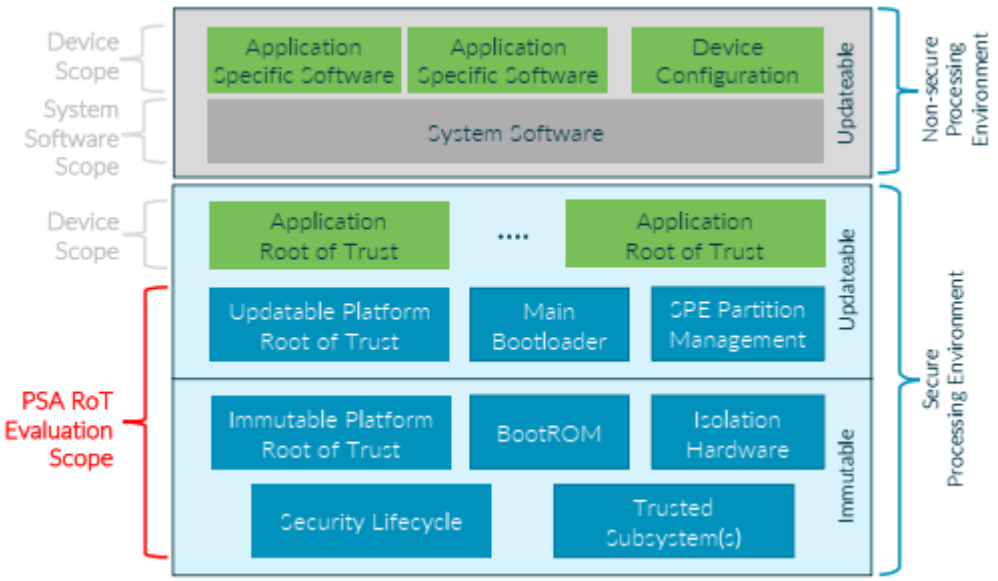

Figure 2: Scope of PSA Certified Level 3 (from [PSA-SESIP-L3])

1.6.5. Required Hardware/Software/Firmware

QuiddiKey has the following minimal requirements regarding external components:

- The PUF SRAM interface needs to be directly connected to a dedicated (fully isolated) SRAM which is not initialized upon power-on, and hence contains an uninitialized SRAM state (i.e., it operates as an SRAM PUF).
- The EC SRAM interface needs to be connected to a digital memory which is isolated from other processes.
- In a system integrating QuiddiKey, non-volatile (NV) storage is required for storing the activation code (AC) of QuiddiKey. The AC is a non-sensitive, but essential data packet which QuiddiKey needs to reliably reconstruct the device-intrinsic key. The required NV storage can be anywhere in the logical scope, since it does not require external protection. It can even reside outside the physical boundary of the system (e.g., in the cloud).

2. Security Objectives for the Operational Environment

For the platform to fulfil its security requirements, the operational environment (technical or procedural) must fulfil the objectives listed in Table 5.

ID	Description	Reference
SRAM_PUF	The SRAM connected to PUF SRAM and EC SRAM interfaces shall be implemented and integrated correctly, following all requirements listed in the QuiddiKey Integration Manual and Security Guidance. In particular: – PUF SRAM region shall not be initialized after power-on – PUF SRAM and EC SRAM regions shall be dedicated to their use for QuiddiKey – Integrity and Confidentiality of data R/W to PUF SRAM and EC SRAM shall be ensured by the integration	[IID-QK300-1-0-IM] [IID-QK300-1-0-SG]
AC_AVAILABILITY	The activation code generated by QuiddiKey during the Enroll process needs to be available later to successfully run the Start process. Availability of the AC shall be ensured by the integration, generally by storing it in a reliable and accessible non-volatile memory.[2]	[IID-QK300-1-0-IM]
TRUSTED_ INTEGRATION	The integrator shall ensure that the integration and security guidance requirements are followed. The integrator is trusted by default, and hence shall not attempt to thwart the component's security functionalities.	[IID-QK300-1-0-SG]

[2] This is an availability objective rather than a security objective, but it is essential to the operation of QuiddiKey.

ID	Description	Reference
ACCESS_CONTROL	QuiddiKey on its own has no direct access control mechanisms. Logical access to its interfaces and functionality shall be controlled by the integrating system in accordance with the security use case(s).	[IID-QK300-1-0-SG]
FAULT_DETECTION_SUPPORT	QuiddiKey includes methods for detecting common operational faults, e.g., due to a fault injection attack on its I/O interface. These methods shall be integrated with the proper support from the integrating system, e.g., to assess the criticality of detected faults, or to detect faults in external I/O behavior. See Section 4 in [IID-QK300-1-0-DS] for all details.	[IID-QK300-1-0-SG] [IID-QK300-1-0-DS]

Table 5: Security Objectives for the Operational Environment

3. Security Requirements and Implementation

3.1. Security Assurance Requirements

The claimed assurance requirements package is SESIP3 as defined in [GP-SESIP].

3.1.1. Flaw Reporting Procedure (ALC_FLR.2)

In accordance with the requirement for a flaw reporting procedure (ALC_FLR.2), including a process to generate any needed update and distribute it, the developer has defined procedure outlined in the following subsections.

3.1.1.1. Flaw Reporting: Contact

In case a flaw or bug is found in an Intrinsic ID product, customers are asked to report this as soon as possible to the Intrinsic ID technical team, via one of the following ways:

- Contact the official technical contact that handles your account by e-mail (preferred).
- Send an e-mail to customer support: customer.service@intrinsic-id.com.
- Fill in the customer feedback form on our delivery site.

3.1.1.2. Flaw Reporting: Follow-Up and Updates

Upon reception of a flaw report, Intrinsic ID will start investigation with priority and respond as follows:

- Acknowledge reception of the flaw report to the customer.
- Register the reported flaw in its internal database.
- Investigate the issue to find the root cause.
    - Reproduce issue/flaw in test set-up.
    - Ask for elaboration by customer if needed.
    - Pinpoint root cause.
- Add test case to regression testing (if possible).
- Implement fix for root cause.
    - Define solution and implications.
    - Discuss implications with appropriate internal parties.
    - Plan engineering effort to implement fix.
    - Provide information about the issue and its fix to affected customer(s) along with timeline for patch update.
    - Implement fix.
    - Verify issue resolution and perform release cycle.
    - Provide patch update to affected customers.
- If applicable a security guidance update will be part of the update.

Note: In some cases, extra information might be needed, e.g., to reproduce the flaw. In these cases, Intrinsic ID will contact the customer as needed.

In general, a solution will be provided by sending an updated delivery. When the design cannot be updated (e.g., after tape-out of a hardware project, or software in ROM code), a

workaround can be provided, e.g., through updated security guidance, a driver update, or a recommendation on the operating conditions.

3.1.1.3. Flaw Reporting: Secure Communication with Customer Service

If sensitive material needs to be shared, (including, but not limited to, parts or complete sets of Intrinsic ID IP) it needs to be shared securely. Security of possibly involved customer or 3rd party IP is the responsibility of the customer, Intrinsic ID will follow customer guidelines for handling these sensitive materials.

Default accepted options for sharing sensitive material are:

- PGP encryption to be done with the public PGP key of the technical contacts of either side.
- 7zip AES encryption with password communication over a second channel e.g., texting password to a known cell phone number.

Intrinsic ID is flexible in what secure mechanism is used; reasonable customer specific solutions can be used to communicate securely after internal deliberation.

3.2. Base PP Security Functional Requirements

3.2.1. Verification of Platform Identity

QuiddiKey provides the platform with a method to verify the unique component identity and version.

Conformance Rationale:

The QK IP has its own configuration and version numbers, which are respectively retrieved by reading out the QK_INFO_PRODUCT and QK_INFO_VERSION registers. It reports the product configuration and the major version, minor version and revision numbers indicating the exact version of the IP design. Next to the configuration and version information, the QK IP also has a unique project ID number which is different per customer. This is also hard coded in the HW and can be read via the register interface on the QK_INFO_ID register. The configuration and version information and project ID number of the QK IP are hard-coded into the HW logic and are immutable after tape-out. The unicity and continuity of the configuration and version information, and the unicity of the project ID number are guaranteed by the Intrinsic ID release procedures.

After production, the configuration and version information and project ID number of the QK IP can be read from their corresponding registers and compared against the release information.

3.2.2. Secure Update of Platform

QuiddiKey can (to the extent possible) be updated to a newer version in the field such that the integrity, authenticity, and confidentiality of the platform is maintained.

Conformance Rationale:

The QK IP is a HW IP component and is as such part of the platform's **immutable** trusted subsystems. For most integrating platforms, immutable subsystems are implemented in fixed hardware and hence unchangeable post-production. As a result, secure update of this part of the platform is considered out of scope for such platforms. For platforms which do support post-production hardware modifications (e.g., FPGAs), the secure update mechanisms of the composing platform shall be used for configuring updated versions of the QK IP.

3.2.3. Physical Attacker Resistance

QuiddiKey detects or prevents attacks by an attacker with physical access before the attacker compromises any of the other functional requirements, ensuring that the other functional requirements are not compromised.

Conformance Rationale:

In the certified configuration of the QK IP ('Normal operating mode'), access to the Lab Test Mode, Production Test Mode and Loopback Scan Mode are not possible. In Normal mode, it is not possible to access any (possibly sensitive) internal data contents of the QK IP through its physically accessible interfaces. The QK IP can be hard set to Normal operating mode by fusing or hardwiring the pins for mode selection. The internal signal interfaces of these mode selection pins are also implemented with enhanced fault-injection resistance.

The QK IP has several logical and algorithmic countermeasures embedded into its HW implementation to significantly enhance the side-channel and fault-injection resistance of its data paths and control flow. See [IID-QK300-1-0-SD] for a detailed overview.

3.3. SFRs for PSA-RoT Component

3.3.1. Verification of Platform Instance Identity

QuiddiKey provides a method to verify the specific instance identity of the implemented QuiddiKey component. The Security Guidance [IID-QK300-1-0-SG] outlines the recommended functional flow for implementing this method on the platform. Since the QK component is implemented in HW and hence physically bound to the platform, the QK component instance identity can also serve as a platform instance identity for the composing system.

Conformance Rationale:

QuiddiKey has the ability to reliably generate a large number of device-unique keys derived from the unique SRAM PUF implementation. One or more of these keys can be dedicated for use as a unique instance identifier.

3.3.2. Cryptographic Operation

QuiddiKey provides the composing system with a method for wrapping/unwrapping[3] cryptographic keys and assets. The Security Guidance [IID-QK300-1-0-SG] outlines the recommended functional flow for implementing this method on the platform. The application note *QuiddiKey Use Case: Embedded Key Vault* [IID-AN102] gives further details on how to use this cryptographic operation of QuiddiKey to support a cryptographic key storage solution.

Conformance Rationale:

QuiddiKey implements a NIST-compliant key wrapping/unwrapping scheme (authenticated en/decryption) which uses an SRAM PUF-derived key as the main key wrapping key. The composing system has access to this functionality through the QuiddiKey **wrap** and **unwrap** operations. In combination with a (non-secured) non-volatile memory for storing wrapped keys (so called key codes), this can straightforwardly be turned into a cryptographic key store by the composing system.

The key wrapping scheme consists of an encrypt-then-MAC construction using AES-256 in CTR mode for encryption and HMAC-SHA256 as MAC function. Used key wrapping keys depend on the used SRAM PUF/physical device and activation code, and hence wrapped keys can only be unwrapped on the same device they were wrapped on. QuiddiKey can wrap keys or other secure assets of various lengths ranging from 64 bits up to 4096 bits, including the standard symmetric key lengths of 128 bits, 192 bits, and 256 bits. The security strength of the QuiddiKey wrap/unwrap scheme is 256 bit.

It is verified in simulation testing that the expected wrapped keys are returned by QuiddiKey and can consequently be unwrapped to their original key value. The AES CTR-mode and HMAC-SHA256 functionality used in the QuiddiKey wrap/unwrap scheme can be validated as part of the CAVP certification (also see Section 1.5.1).

Table 6 summarizes the main cryptographic algorithms used in this SFR. The main intended use of this operation is to provide secure storage for the composing system.

[3] In the scope of this document, wrapping and unwrapping are to be considered as synonymous with respectively authenticated encryption and decryption, based on a wrapping key which is derived from the PUF-based root key.

Algorithm	Operations	Specification	Key Length (bit)
AES-CTR	Key encryption/decryption part of the wrap/unwrap operation.	NIST SP800-38A	256
HMAC-SHA256	Key (code) authentication/verification part of the wrap/unwrap operation.	FIPS 198-1	256
KDF in Counter Mode (based on HMAC-SHA256)	Key derivation part of the wrap/unwrap operation, generating distinct subkeys for use with AES-CTR and HMAC.	NIST SP800-108	256(key derivation key)

Table 6: Cryptographic Operation

3.3.3. Cryptographic Random Number Generation

QuiddiKey provides the composing system with a method for generating cryptographic random numbers. The Security Guidance [IID-QK300-1-0-SG] outlines the recommended functional flow for implementing this method on the platform.

Conformance Rationale:

QuiddiKey implements a NIST-compliant random bit generator which is accessible to the composing system for the generation of random bits and numbers. The random bit generator implemented by QuiddiKey consists of an entropy source (NIST SP800-90B compliant) and two chained DRBGs (NIST SP800-90A certified). After a power cycle, DRBG 1 is instantiated with entropy from the entropy source based on the SRAM PUF noise. After a power cycle and after every reset, during the initialization procedure of QuiddiKey, DRBG 2 is instantiated with a random seed generated by DRBG 1. At any point DRBG 2 can also be reseeded with a random seed generated by DRBG 1. The random output generated by DRBG 2 is available to the platform as a source of random bits, on the SW-accessible APB interface or on the HW-only SO interface.

It is verified in simulation testing that the DRBG state gets updated with expected values and that expected random data is generated. The QuiddiKey DRBG functionality can be validated as part of a CAVP certification (also see Section 1.5.1). The operation and compliance of the SRAM PUF noise-based entropy source is motivated in detail in the Entropy Source Validation Guide [IID-ESVG-NIST90B].

3.3.4. Cryptographic Key Generation

QuiddiKey provides the composing system with a method for generating cryptographic keys for use in cryptographic operations. The Security Guidance [IID-QK300-1-0-SG] outlines the recommended functional flow for implementing this method on the platform. The application note *Multiple Key Generation with QuiddiKey* [IID-AN100] gives further details on how to use QuiddiKey for cryptographic key generation.

Conformance Rationale

QuiddiKey implements a NIST-compliant key derivation method for generation of a device-unique root key derivation key from the SRAM PUF secret, and a key derivation function (KDF) for the further derivation of application-specific cryptographic keys for internal and external use. The platform can access derived keys for external use through the QuiddiKey **get key** operation. The key derivation method for generating the root key is constructed in compliance with the extract-then-expand methodology outlined in NIST SP800-56C. The key derivation function for deriving application-specific keys is compliant with the counter based KDF method from NIST SP800-108. Both are using HMAC-SHA256 as pseudorandom function primitive. The use of a cryptographic KDF ensures that all derived keys are cryptographically separated, i.e., the disclosure of any derived key does not compromise the security of any other derived key. For the externally accessible QuiddiKey **get key** operation, key differentiation is based on the used root key, which in the end traces back to the used SRAM PUF/physical device and activation code, and is also based on a user-provided key context with meta-information about the derived key. QuiddiKey generates fully random keys in various lengths ranging from 64 bits up to 4096 bits, including the standard symmetric key lengths of 128 bits, 192 bits, and 256 bits. The security strength of keys derived by QuiddiKey is 256 bit, or the length of the derived key, whichever is smaller.

QuiddiKey **does not** require secure storage of a *Hardware Unique Key* by the platform to build its secure key generation functionality on. Instead, QuiddiKey relies on the SRAM PUF as a root of trust in a self-contained manner.

It is verified in simulation testing that the expected key values are generated by QuiddiKey. The QuiddiKey KDF functionality can be validated as part of the CAVP certification (also see Section 1.5.1).

Algorithm	Specification	Key Lengths (bits)
KDF in Counter Mode (based on HMAC-SHA256)	NIST SP800-108	length range: 64—1024 bits in 64-bit increments, 2048 bits, 3072 bits, 4096 bits

Table 7: Cryptographic Key Generation

3.3.5. Cryptographic Key Store

QuiddiKey provides the composing system with a method for securely storing diverse hardware-unique symmetric keys such that not even the platform/application can compromise their authenticity and confidentiality. The Security Guidance [IID-QK300-1-0-SG] outlines the recommended functional flow for implementing this method on the platform. These keys can be used for the cryptographic operations of wrapping/unwrapping (i.e., authenticated en/decryption) and key derivation.

Conformance Rationale:

QuiddiKey implements the theoretical principle of a fuzzy commitment scheme (also see [JW99]) which is used to derive a strong cryptographic secret based on a secure but so-called

fuzzy source, i.c., an SRAM PUF. From this strong root secret, several other cryptographic keys are derived using a NIST-compliant cryptographically secure key derivation function (KDF). These keys are by their nature unique to the physical hardware they are derived on and are not stored in binary form in any non-volatile memory on the device. Instead, they are only (re)derived from the SRAM PUF when needed. In this way, there is no need for one or more root keys to be programmed from the outside world into a protected non-volatile memory on the device. Rather, the internal entropy of the silicon is used to derive and store a root key derivation key, which is never exposed outside QuiddiKey.

QuiddiKey offers the platform the functionality for using these SRAM PUF-derived keys for:

- the wrapping and unwrapping of platform- or application-level keys or assets, without the wrapping keys ever being exposed outside QuiddiKey (also see Section 3.3.2)
- the further secure derivation of platform/application-level keys, without the key derivation keys ever being exposed outside QuiddiKey (also see Section 3.3.4)

The ability to securely recreate and use cryptographic keys essentially enables QuiddiKey to operate as a cryptographic key store. An important requirement for this operation is the availability of the activation code (AC) during the **start** operation of QuiddiKey (also see the AC_AVAILABILITY objective in Section 2).

It is verified in simulation testing that the expected derived hardware-unique keys are used by QuiddiKey and can consequently be rederived when needed. The internal QuiddiKey KDF operation can be validated as part of the CAVP certification (also see Section 1.5.1).

3.4. Additional Security Functional Requirements

3.4.1. Secure Communication Support

QuiddiKey provides the application with a secure communication channel.

Conformance Rationale:

The QK IP is a HW-only fully-on-chip trusted subsystem. The security of the communication between the secure processing environment and the QK IP as trusted subsystem is ensured by the proper physical integration of the QK IP, as required per security objective TRUSTED_INTEGRATION. All communication between the secure processing environment and the QK IP occurs over internal on-chip signals which are not directly accessible nor influenceable externally.

3.4.2. Secure Communication Enforcement

QuiddiKey ensures that the application can only communicate with the trusted subsystem over a secure communication channel.

Conformance Rationale:

The QK IP is a HW-only fully-on-chip trusted subsystem. The security of the communication between the secure processing environment and the QK IP as trusted subsystem is enforced by the proper physical integration of the QK IP, as required per security objective TRUSTED_INTEGRATION. The internal on-chip signals used for secure communication

between the secure processing environment and the QK IP are the only way of accessing the QK IP. No other, potentially non-secure, communication interface exists. Further enforcement of secure communication between QK IP and dedicated security subsystems shall be controlled by proper access control measures provided by the secure processing environment as required per security objective ACCESS_CONTROL.

3.5. Optional Security Functional Requirements

No further optional SFRs for QuiddiKey are specified.

4. Mapping and Sufficiency Rationales

4.1. Assurance

The assurance activities defined in [PSA-SESIP-L3-EM] fulfil the SESIP3 activities. In particular, the required source code review, vulnerability analysis and testing of the [PSA-SESIP-L3-EM] is applicable.

Assurance Class	Assurance Families	Covered by	Rationale
ASE: Security Target evaluation	ASE_INT.1 ST Introduction	Section 1 and in the document Title	The ST reference is in the Title, the platform reference in Section 1.3, the platform overview and description in Section 1.6
	ASE_OBJ.1 Security requirements for the operational environment	Section 2	The objectives for the operational environment in Section 2 refers to the guidance documents.
	ASE_REQ.3 Listed Security requirements	Sections 3.2, 3.3, 3.4 and 3.5	All SFRs in this ST are taken from [GP-SESIP]. “Verification of Platform Identity” is included.
	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 Sections 3.2, 3.3, 3.4 and 3.5.
ADV: Development	ADV_FSP.4 Complete functional specification	Functional specification as specified in Section 1.4	The functional specification describes the complete set of TSF interfaces.

	ADV_IMP.3 Complete mapping of the implementation representation of the TSF to the SFRs	Implementation representation and mapping to SFRs as specified in Section 1.4	The implementation representation can be mapped to the SFRs defined in Sections 3.2, 3.3, 3.4 and 3.5
AGD: Guidance documents	AGD_OPE.1 Operational user guidance	Guidance documents listed in Section 1.4	The operational user guidance describes secure usage of the user accessible functions.
	AGD_PRE.1 Preparative procedures	Guidance documents listed in Section 1.4	The preparative procedures describe how the platform is brought into a secure configuration.
ALC: Life-cycle support	ALC_CMC.1 Labelling of the TOE	Section 1.3	The platform is clearly identified as stated in the ST.
	ALC_CMS.1 TOE CM Coverage	Section 1.3 and Guidance documents listed in Section 1.4	Configuration items are properly identified.
	ALC_FLR.2 Flaw reporting procedures	Section 3.1.1	The flaw reporting and remediation procedure is described.
ATE: Tests	ATE_IND.1 Independent testing: conformance	Evaluator testing carried out by laboratory	The platform evaluator will determine whether the provided evidence is suitable to meet the requirement
AVA_VAN.3	AVA_VAN.3 Focused vulnerability analysis	Vulnerability and testing carried out by the laboratory	The platform evaluator performs penetration testing, to confirm that the

			potential vulnerabilities cannot be exploited in the operational environment for the platform. Penetration testing is performed by the platform evaluator assuming an attack potential of EnhancedBasic.

Table 8: Assurance Mapping and Sufficiency Rationales

4.2. PSA Security Functions Mapping

PSA Security Function	Covered by SESIP SFR	Rationale
F.ATTESTATION	Verification of Platform Identity	Unique platform identity
	Verification of Platform Instance Identity	Instance-unique platform identity and root key
F.CRYPTO	Cryptographic Operation	Wrapping/unwrapping of sensitive data with instance-unique keys
	Cryptographic Random Number Generation	Entropy source and DRBG compliant with NIST SP800-90 methodology
	Cryptographic Key Generation	NIST SP800-108 compliant KDF for generation of instance-unique keys and random keys
	Cryptographic Key Store	Ability to regenerate keys from instance-unique root key
F.PHYSICAL	Physical Attacker Resistance	Physical inaccessibility of operation and implementation of algorithmic countermeasures
Additional security functionality	Secure Communication Support	Fully internal on-chip communication, no external access
	Secure Communication Enforcement	Exclusively fully internal on-chip communication, no alternatives

Table 9: Functionality Mapping and Sufficiency Rationales

5. References

- [PSA-RoTComponent] "IoT Security Certification for IP providers." https://www.psacertified.org/getting-certified/ip-provider/ (23/02/2023)
- [GP-SESIP] "GP_FST_070 Security Evaluation Standard for IoT Platforms (SESIP) v1.1", Jun 2021, GlobalPlatforms, https://globalplatform.org/specs-library/security-evaluation-standard-for-iot-platforms-sesip-v1-0-gp_fst_070/
- [PSA-SESIP-L3] "JSADEN011 SESIP Profile for PSA Certified™ Level 3", doc.v.1.0-REL-02, PSA, https://www.psacertified.org/app/uploads/2022/11/JSADEN011-PSA_Certified_Level_3_PP_SESIP-V1.0-REL-02.pdf
- [PSA-SESIP-L3-RoTComponent] "JSADEN018 SESIP Profile for PSA Certified™ RoT Component Level 3", doc.v.1.0-REL-02, PSA, https://www.psacertified.org/app/uploads/2022/11/JSADEN018-PSA_Certified_RoT_Component_Level_3_PP_SESIP-v1.0-REL-02.pdf
- [PSA-SESIP-L3-EM] "PSA Certified: Evaluation Methodology for PSA L3", doc.v.1.0-ALP01, PSA
- [AMD-XC7Z020] "Zynq-7000 SoC Data-Sheet", AMD/Xilinx, https://www.xilinx.com/support/documentation/data_sheets/ds187-XC7Z010-XC7Z020-Data-Sheet.pdf
- [AMD-ArtyZ7-20] "Arty Z7-20: SoC Zynq®-7000 Development Board", AMD/Xilinx, https://www.xilinx.com/products/boards-and-kits/1-pdb0q2.html
- [NIST-SP80056CR2] "NIST SP 800-56C Rev. 2 Recommendation for Key-Derivation Methods in Key-Establishment Schemes", NIST, https://csrc.nist.gov/publications/detail/sp/800-56c/rev-2/final
- [NIST-SP80090AR1] "NIST SP 800-90A Rev. 1, Recommendation for Random Number Generation Using Deterministic Random Bit Generators", NIST, https://csrc.nist.gov/publications/detail/sp/800-90a/rev-1/final
- [NIST-SP80090B] "NIST SP 800-90B Recommendation for the Entropy Sources Used for Random Bit Generation", NIST, https://csrc.nist.gov/publications/detail/sp/800-90b/final
- [IID-AN102] "QuiddiKey Use Case: Embedded Key Vault", Intrinsic ID Application Note.
- [IID-AN100] "Multiple Key Generation with QuiddiKey", Intrinsic ID Application Note.
- [NIST-SP80090A] "NIST SP 800-90A Revision 1. Recommendation for Random Number Generation Using Deterministic Random Bit Generators", NIST, https://csrc.nist.gov/publications/detail/sp/800-90a/rev-1/final
- [NIST-SP800108] "NIST Special Publication 800-108 Recommendation for Key Derivation Using Pseudorandom Functions", NIST, https://csrc.nist.gov/publications/detail/sp/800-108/rev-1/final

- [FIPS-PUB-180-4] "FIPS PUB 180-4 Secure Hash Standard (SHS)", Aug 2015, NIST, https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
- [IID-QK300-1-0-DS] "QuiddiKey 300 v1.0 Datasheet" doc.v1.3, Intrinsic ID
- [IID-QK300-1-0-IM] "QuiddiKey 300 v1.0, Integration Manual" doc.v1.1, Intrinsic ID
- [IID-QK300-1-0-DRV-RM] "QuiddiKey 300 v1.0, Driver Reference Manual", doc.v1.2, Intrinsic ID
- [IID-QK300-1-0-CCR] "QuiddiKey 300 v1.0, Code Coverage Report", doc.v1.1, Intrinsic ID
- [IID-QK300-1-0-RN] "QuiddiKey 300 v1.0, Release Notes", doc.v1.2, Intrinsic ID
- [IID-QK300-1-0-ACVP] "QuiddiKey 300 v1.0, ACVP parameters", doc.v1.1, Intrinsic ID
- [IID-QK300-1-0-SG] "QuiddiKey 300 v1.0, Security Guidance", doc.v1.3, Intrinsic ID
- [IID-QK300-1-0-SD] "QuiddiKey 300 v1.0, Security Documentation", doc.v1.3, Intrinsic ID
- [IID-TS-Guide] "TOE Test Guidance:, doc.v1.0, Intrinsic ID.
- [IID-QK300-1-0-PSA-DOCKER-SG] "QuiddiKey 300 v1.0 PSA evaluation, Docker Setup Guide", doc.v1.0, Intrinsic ID.
- [IID-QK300-1-0-PSA-TOE-CLI] "QuiddiKey 300 v1.0 PSA evaluation, TOE command line interface", doc.v1.0, Intrinsic ID.
- [IID-ESVG-NIST90B] "SRAM PUF-based Entropy Source, Validation Guide for NISTSP800-90B", v1.0, Intrinsic ID.
- [IID-QK300-1-0-ESVGA] "QuiddiKey 300 v1.0 Entropy Source Validation Annex", doc.v1.0.
- [JW99] "A Fuzzy Commitment Scheme", A. Juels and M. Wattenberg, ACM CCS 1999
- [IID-QK300-1-0-CL] "QK 300 v1.0 Configuration List Document" , doc.v1.3, Intrinsic ID.

6. Terms and abbreviations

Term	Meaning
AC	Activation Code
APB	Advanced Peripheral Bus Architecture
Application Root of Trust Service(s)	Application specific security service(s), and so not defined by PSA. Such services execute in the Secure Processing Environment are required to be in Secure Partitions.
Application Specific Software	Software that provides the functionality required of the specific device. This software runs in the Non-Secure Processing Environment, making use of the System Software, Application RoT Services and PSA-RoT Services
Connected Application	Software developed by an IoT vendor, implementing IoT end-user use case based on underlying Connected Platform. May be referred to as "Application" when there is no ambiguity.
Connected Platform	Combination of hardware and software that provides a runtime environment for a Connected Application. A Connected Platform implements security features and makes security services available to the Connected Application. May be referred to as "platform" when there is no ambiguity.
Connected product	Combination of a Connected Platform and a Connected Application that a product vendor puts on the market. May be referred as "product" when there is no ambiguity.
Critical Security Parameter	Secret information, with integrity and confidentiality requirements, used to maintain device security, such as authentication data (passwords, PIN, certificates), secret cryptographic keys, etc..
DRBG	Deterministic Random Bit Generator
Evaluation Laboratory	Laboratory or facility that performs the technical review of questionnaires submitted for Level 1 PSA certification. The list of evaluation laboratories participating to PSA Certified can be found on www.psacertified.org
HW	Hardware
ID	Identifier
Immutable Platform Root of Trust	The minimal set of hardware, firmware and data of the PSA-RoT, which is inherently trusted because it cannot be modified following manufacture. There is no software at a deeper level that can verify that it is authentic and unmodified.
Intellectual Property (IP)	An idea, a design, etc. that somebody has created and that the law prevents other people from copying.
IP	Intellectual property
KC	Key Code
KDF	Key Derivation Function
MCU	Microcontroller

NV(M)	Non-Volatile (Memory)
Platform	Used in SESIP to refer to the components which are in the scope of the evaluation.
Platform Root of Trust Service(s)	PSA defined security services for use by PSA-RoT, Application RoT Service(s) and by the NSPE. Executes in the Secure Processing Environment and may use Trusted Subsystems. This includes the services offered by the PSA Functional APIs.
PSA	Platform Security Architecture
PSA Functional APIs	PSA defined Application Programming Interfaces on which security services can be built. APIs defined so far include Crypto, Secure Storage and Attestation.
PSA Root of Trust (PSA-RoT)	The PSA defined combination of the Immutable Platform Root of Trust and the Updateable Platform Root of Trust, and considered to be the most trusted security component on the device.
PUF	Physical Unclonable Function
QK	QuiddiKey
QK IP	QuiddiKey IP: short-hand for referring to the QuiddiKey hardware module
Register-transfer level (RTL)	Register-transfer level (RTL) is a design abstraction which models a synchronous digital circuit in terms of the flow of digital signals (data) between hardware registers, and the logical operations performed on those signals.
RoT	Root-of-Trust
Security Evaluation Standard for IoT Platform (SESIP)	Security Evaluation Standard for IoT Platforms (SESIP), published by GlobalPlatform, defines a standard for trustworthy assessment of the security of the IoT platforms, such that this can be re-used in fulfilling the requirements of various commercial product domains.
Security Target (ST)	Document providing an implementation-dependent statement of security of a specific identified platform.
SI	Secure Input
SO	Secure Output
SRAM	Static random-access memory
SW	Software
Target of Evaluation (TOE)	It is the combination of the hardware and firmware components supporting a device compliant with PSA specification. The considered hardware may be a System-in-Package (SiP), a System-on-Chip (SoC) integrated on a board, or similar set-up. The hardware is in the scope of the security evaluation as it provides security features, such as immutable storage or protection of JTAG, which are essential for ensuring the security of the PSA implementation.
Updateable Platform Root of Trust	The firmware, software and data of the PSA-RoT that can be securely updated following manufacture.