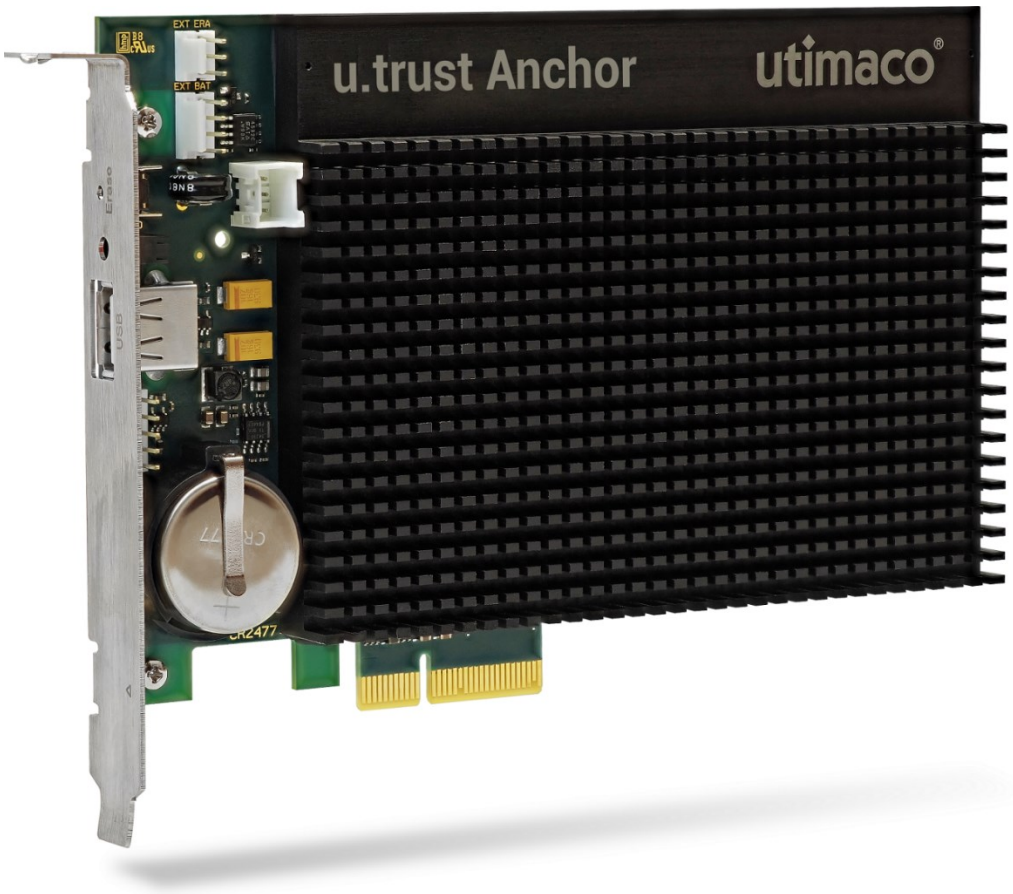

Utimaco IS GmbH

u.trust Anchor

FIPS 140-3 Non-Proprietary Security Policy

utimaco®

Imprint

Utimaco IS GmbH
Germanusstraße. 4
D-52080 Aachen, Germany

Phone
AMERICAS: +1-844-UTIMACO (+1 844-884-6226)
EMEA: +49 800-627-3081
APAC: +81 800-919-1301

Internet https://support.hsm.utimaco.com

Email support@utimaco.com

Document version 2.3.6

Date 2025-12-19

Status Released

Document No. 2020-0031

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document defines the security policy for u.trust Anchor Cryptographic Module, hereafter referred to as “the module” or “u.trust Anchor”. u.trust Anchor is a hardware security module made by Utimaco IS GmbH, hereafter referred to as Utimaco.

u.trust Anchor is suitable for use in a multi-tenant environment where multiple users run their own applications on different containers deployed on the same hardware.

The module is a multi-chip embedded cryptographic module as defined by FIPS 140-3.

1.2 Security Levels

The module meets the overall requirements of FIPS 140-3 Security Level 3.

Section	*Title*	*Security Level*
1	General	3
2	Cryptographic module specification	3
3	Cryptographic module interfaces	3
4	Roles, services, and authentication	3
5	Software/Firmware security	3
6	Operational environment	N/A
7	Physical security	3
8	Non-invasive security	N/A
9	Sensitive security parameter management	3
10	Self-tests	3
11	Life-cycle assurance	3
12	Mitigation of other attacks	3
	Overall Level	3

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

The module is an encapsulated, protected hardware security module (HSM) realized as a multi-chip embedded cryptographic module. Figure 1 shows the physical module.

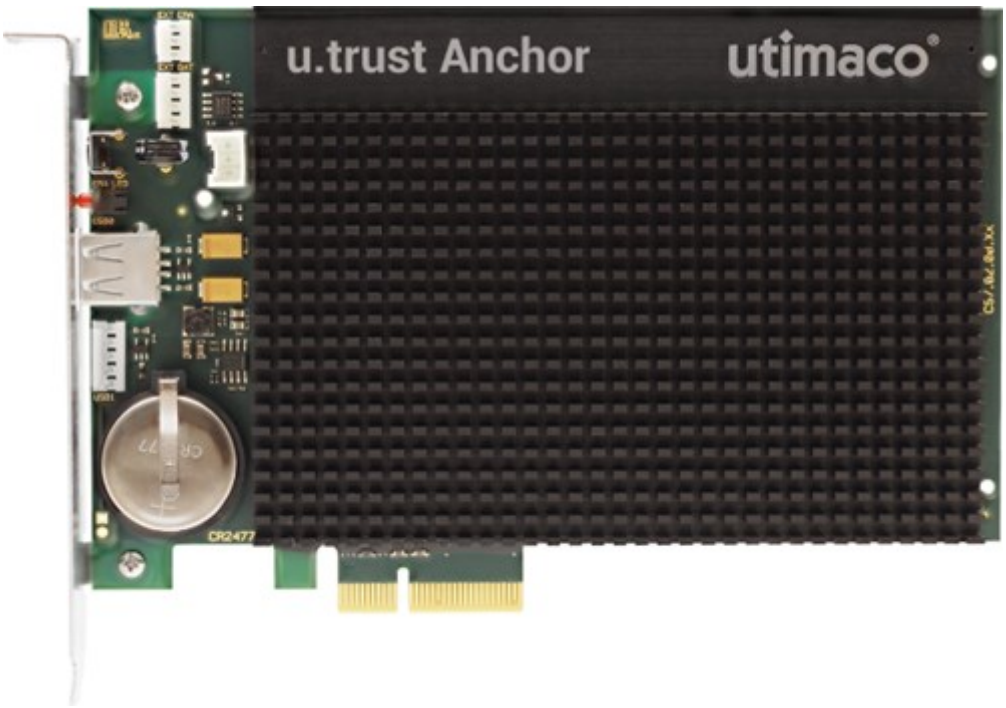

Figure 1: u.trust Anchor as a PCIe plug-in card

Purpose and Use:

The u.trust Anchor is the next generation of HSMs. It incorporates all the traditional hardware security features as well as the concept of containerized HSMs (cHSMs) and multi-tenancy.

The u.trust Anchor platform consists of the following subsystems:

- The u.trust Anchor hardware
- The u.trust Anchor platform firmware called COSMOS including Bootloader, Linux kernel, container management firmware, and Global Administration (glad) service firmware
- cHSM (containerized HSM) firmware which can be loaded into containers as provided by COSMOS

Module Type: Hardware

Module Embodiment: MultiChipEmbed

Module Characteristics:

The critical components within the module are encapsulated on a PCIe plug-in card with the boundaries described below.

Cryptographic Boundary:

The module's cryptographic boundary is defined as the outer perimeter of the heat sink on the top side and the epoxy surface on the bottom side of the module.

Figure 2, Figure 3, and Figure 4 show the side, top, and bottom views of the module. The red dashed line indicates the cryptographic boundary.

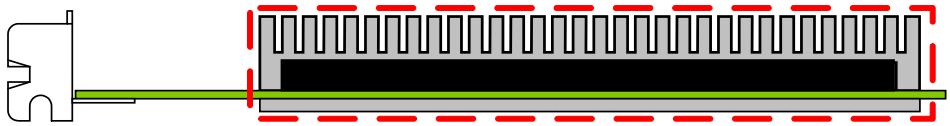

Figure 2: u.trust Anchor - side view

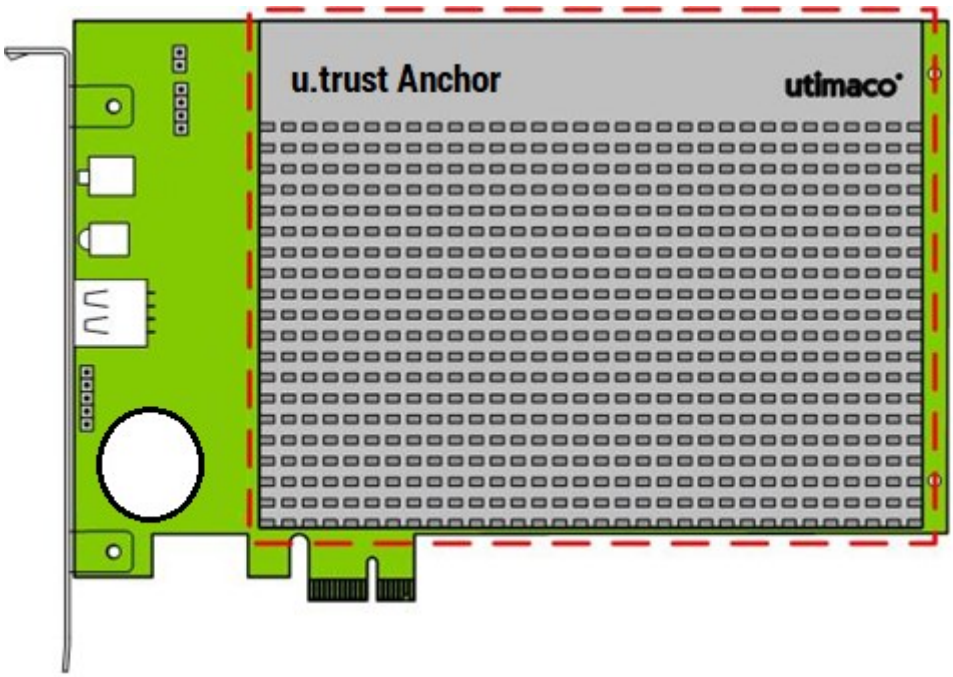

Figure 3: u.trust Anchor - top view

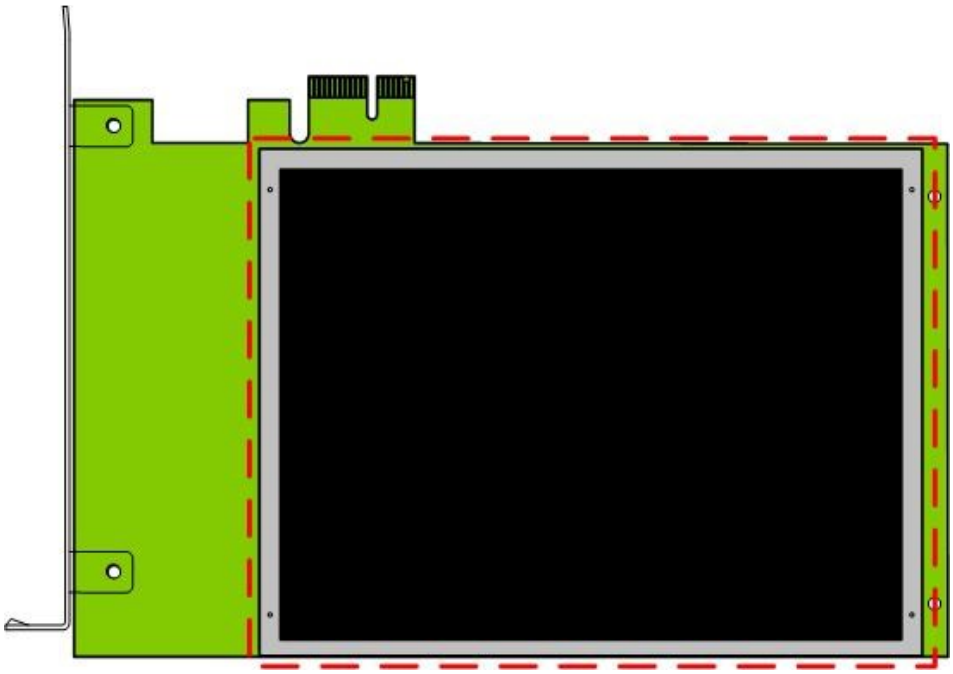

Figure 4: u.trust Anchor - bottom view

2.2 Tested and Vendor Affirmed Module Version and Identification

The module is designed to meet the requirements of FIPS 140-3 Security Level 3 (refer to Table 1). The module is available in the following configurations listed in Table 2.

Tested Module Identification – Hardware:

Model and/or Part Number	*Hardware Version*	*Firmware Version*	*Processors*	*Features*
u.trust Anchor	u.trust Anchor 7.03.00.03	Device System v6.0.1.0-c Sensory Controller v3.02.0.7, v3.02.0.8	Quad ARM Cortex-A53	Tamper Protected PCIe Card

Table 2: Tested Module Identification – Hardware

The hardware version covers variants with and without the Exar accelerator chip.

The u.trust Anchor with hardware version 7.03.00.03 is called Se[1].

CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

[1] The module could be marketed as various variants, e.g., u.trust Anchor Se100, Se2k, Se5K8, Se5K16, Se5K31, Se15K, Se40K8, Se40K16, Se40K31, and CSAR (Standard, Plus, Premium).

2.3 Excluded Components

There are no components excluded from the cryptographic boundary.

2.4 Modes of Operation

Modes List and Description:

The module supports an Approved and a Non-Approved mode of operation. Global Administrator Application (glad) always runs in the Approved mode. Each cHSM container can run either the Approved mode or the Non-Approved mode. The module does not support a maintenance interface.

Mode Name	*Description*	*Type*	*Status Indicator*
Approved	Services associated with the Global Administrator Application or a containerized HSM created with the template 'SecurityServer-FIPS'	Approved	glad_Status shows versions as listed in table 2 "Tested Module Identification" or cHSM_Status shows 'FIPS mode = ON'
Non-Approved	Services associated with a containerized HSM created with the template 'SecurityServer'	Non-Approved	cHSM_Status does not show 'FIPS mode = ON'

Table 3: Modes List and Description

Mode Change Instructions and Status:

All services associated with the Global Administrator Application always execute in the Approved mode. All services associated with a cHSM that has been created using the template ‘SecurityServer-FIPS’ always execute in the Approved mode. All services associated with a cHSM that has been created using the template ‘SecurityServer’ always execute in the Non-Approved mode. The mode of operation of services associated with the Global Administrator Application or a cHSM never changes. Once a cHSM has been created using either the ‘SecurityServer-FIPS’ or ‘SecurityServer’ template, it is not possible for the cHSM to transition between the two templates. See Section 11.1 for initialization instructions.

Degraded Mode Description:

The module does not support a degraded mode of operation.

2.5 Algorithms

The module supports the approved cryptographic algorithms shown in Table 4.

Approved Algorithms:

The module supports the following approved cryptographic algorithms.

Algorithm	*CAVP Cert*	*Properties*	*Reference*
AES-CBC	A6713	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC	A6714	Direction - Decrypt, Encrypt Key Length - 256	SP 800-38A
AES-CCM	A6713	Key Length - 128, 192, 256	SP 800-38C
AES-CMAC	A6713	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CMAC	A6714	Direction - Generation, Verification Key Length - 256	SP 800-38B
AES-ECB	A6713	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A6713	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-GCM	A6714	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.2 Key Length - 256	SP 800-38D
AES-GMAC	A6713	Direction - Decrypt, Encrypt IV Generation - Internal IV Generation Mode - 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A6713	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A6713	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A6713	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
DSA SigVer (FIPS186-4)	A6713	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-4
ECDSA KeyGen (FIPS186-5)	A6713	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Secret Generation Mode - extra bits	FIPS 186-5
ECDSA KeyGen (FIPS186-5)	A6714	Curve - P-521 Secret Generation Mode - extra bits	FIPS 186-5
ECDSA KeyGen (FIPS186-5)	A6716	Curve - P-224, P-256, P-384, P-521 Secret Generation Mode - extra bits	FIPS 186-5

Algorithm	*CAVP Cert*	*Properties*	*Reference*
ECDSA KeyGen (FIPS186-5)	A6717	Curve - P-224, P-256, P-384, P-521 Secret Generation Mode - extra bits	FIPS 186-5
ECDSA KeyVer (FIPS186-4)	A6713	Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A6716	Curve - P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A6717	Curve - P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-5)	A6713	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A6714	Curve - P-521	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A6716	Curve - P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A6717	Curve - P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A6713	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A6714	Curve - P-256, P-521 Hash Algorithm - SHA2-256 Component - No	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A6716	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A6717	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5
ECDSA SigVer (FIPS186-4)	A6713	Component - No Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4

Algorithm	*CAVP Cert*	*Properties*	*Reference*
ECDSA SigVer (FIPS186-4)	A6716	Component - No Curve - P-192, P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A6717	Component - No Curve - P-192, P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4
ECDSA SigVer (FIPS186-5)	A6713	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A6714	Curve - P-256, P-521 Hash Algorithm - SHA2-256	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A6716	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A6717	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
EDDSA KeyGen	A6713	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyGen	A6716	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyVer	A6713	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyVer	A6716	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA SigGen	A6713	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
EDDSA SigGen	A6716	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
EDDSA SigVer	A6713	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
EDDSA SigVer	A6716	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
Hash DRBG	A6714	Prediction Resistance - No Mode - SHA2-512	SP 800-90A Rev. 1

Algorithm	*CAVP Cert*	*Properties*	*Reference*
Hash DRBG	A6715	Prediction Resistance - No Mode - SHA2-512	SP 800-90A Rev. 1
HMAC-SHA-1	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA2-224	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA2-256	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA2-256	A6714	Key Length - Key Length: 256-8192 Increment 8	FIPS 198-1
HMAC-SHA2-384	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA2-512	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA3-224	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA3-256	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA3-384	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
HMAC-SHA3-512	A6713	Key Length - Key Length: 112-8192 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A6713	Domain Parameter Generation Methods - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A6714	Domain Parameter Generation Methods - P-521 Scheme - ephemeralUnified - KAS Role - responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A6716	Domain Parameter Generation Methods - P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A6717	Domain Parameter Generation Methods - P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - responder	SP 800-56A Rev. 3

Algorithm	*CAVP Cert*	*Properties*	*Reference*
KDA OneStep SP800-56Cr2	A6713	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A6714	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDF ANS 9.63 (CVL)	A6713	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF SP800-108	A6713	KDF Mode - Feedback Supported Lengths - Supported Lengths: 8-512 Increment 8	SP 800-108 Rev. 1
KDF SP800-108	A6714	KDF Mode - Feedback Supported Lengths - Supported Lengths: 8-512 Increment 8	SP 800-108 Rev. 1
KDF TLS (CVL)	A6713	TLS Version - v1.2 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KTS-IFC	A6713	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg2-crt Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
KTS-IFC	A6714	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg2-crt Scheme - KTS-OAEP-basic - KAS Role - responder Key Transport Method - Key Length - 256	SP 800-56B Rev. 2
KTS-IFC	A6716	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg2-crt Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
KTS-IFC	A6717	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg2-crt Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2

Algorithm	*CAVP Cert*	*Properties*	*Reference*
RSA KeyGen (FIPS186-5)	A6713	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - crt	FIPS 186-5
RSA KeyGen (FIPS186-5)	A6714	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - crt	FIPS 186-5
RSA KeyGen (FIPS186-5)	A6716	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - crt	FIPS 186-5
RSA KeyGen (FIPS186-5)	A6717	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - crt	FIPS 186-5
RSA SigGen (FIPS186-5)	A6713	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigGen (FIPS186-5)	A6716	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigGen (FIPS186-5)	A6717	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigVer (FIPS186-2)	A6713	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 1536, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-2)	A6716	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 1536, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-2)	A6717	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 1536, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-4)	A6713	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-4)	A6716	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-4)	A6717	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-5)	A6713	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigVer (FIPS186-5)	A6714	Modulo - 2048, 3072, 4096 Signature Type - pss	FIPS 186-5
RSA SigVer (FIPS186-5)	A6716	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5

Algorithm	*CAVP Cert*	*Properties*	*Reference*
RSA SigVer (FIPS186-5)	A6717	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
SHA-1	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-224	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A1563	Message Length - Message Length: 0-65528 Increment 8	FIPS 180-4
SHA2-256	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A6714	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A6714	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A6715	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA3-224	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-256	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-384	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-512	A6713	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHAKE-256	A6713	Output Length - Output Length: 16-65536 Increment 8	FIPS 202

Table 4: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	*Properties*	*Implementation*	*Reference*
CKG	Key Type:Symmetric and Asymmetric	N/A	NIST SP 800-133r2 and IG D.G per Section 4 example 1, Sections 5.1, 5.2, and 6.1.

Name	*Properties*	*Implementation*	*Reference*
DSA sigVer FIPS 186-4 with SHA-3	Mode:SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Length:1024/160, 2048/224, 2048/256 or 3072/256	N/A	FIPS 186-4
RSA sigVer FIPS186-4 with SHA-3	Mode:SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Length:1024	N/A	FIPS 186-4

Table 5: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

ECDSA (Key Generation and Signature Generation and Verification) with secp256k1 curve listed in Table 6 may only be used in blockchain-related applications.

Name	*Properties*	*Implementation*	*Reference*
EC Diffie-Hellman (Shared Secret Computation)	Curves:brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, brainpoolP224t1, brainpoolP256t1, brainpoolP320t1, brainpoolP384t1, brainpoolP512t1	u.trust Anchor - 32 bit yay crypto library	NIST FIPS 140-3 IG D.F Scenario 3
EC Diffie-Hellman (Key Agreement)	Curves:brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, brainpoolP224t1, brainpoolP256t1, brainpoolP320t1, brainpoolP384t1, brainpoolP512t1	u.trust Anchor - 32 bit yay crypto library	NIST FIPS 140-3 IG D.F Scenario 3
ECDSA (Signature Verification)	Curve:brainpoolP320t1	u.trust Anchor - 64 bit yay crypto library	NIST FIPS 140-3 IG C.A Resolution 1a and NIST SP 800-186
ECDSA (Key Generation, Signature Generation and Verification)	Curve:secp256k1	u.trust Anchor - 32 bit yay crypto library	NIST FIPS 140-3 IG C.A Resolution 1b
ECDSA (Key Generation, Signature Generation and Verification) yay32++ secp256k1	Curve:secp256k1	u.trust Anchor - 32 bit yay crypto library with Silex (yay32++)	NIST FIPS 140-3 IG C.A Resolution 1a

Name	*Properties*	*Implementation*	*Reference*
ECDSA (Key Generation, Signature Generation and Verification) yay32++ Brainpool	Curves:brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, brainpoolP224t1, brainpoolP256t1, brainpoolP320t1, brainpoolP384t1, brainpoolP512t1	u.trust Anchor - 32 bit yay crypto library with Silex (yay32++)	NIST FIPS 140-3 IG C.A Resolution 1a and NIST SP 800-186
ECDSA (Key Generation, Signature Generation and Verification) yay32+	Curves:brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, brainpoolP224t1, brainpoolP256t1, brainpoolP320t1, brainpoolP384t1, brainpoolP512t1	u.trust Anchor - 32 bit yay crypto library with EXAR (yay32+)	NIST FIPS 140-3 IG C.A Resolution 1a and NIST SP 800-186

Table 6: Non-Approved, Allowed Algorithms

Non-Approved, Allowed Algorithms with No Security Claimed:

The module also implements algorithms (listed in Table 7) that may be used in Approved mode of operation but is not a security function per IG 2.4.A. In accordance with IG 2.4.A example 1, these algorithms are used to obfuscate stored CSPs. These algorithms are not used for security-relevant purposes, and no security is claimed.

Name	*Caveat*	*Use and Function*
AES_CBC-CS3-256 (SP 800-38A)	Obfuscation	Obfuscation of encrypted data stored in cHSM file system. This is allowed per IG 2.4.A example 1.
AES_CTR-256 (SP 800-38A)	Obfuscation	Obfuscation of public keys and data stored in glad user backups (yay64 implementation). This is allowed per IG 2.4.A example 1.
AES_XTS-256 (SP 800-38E)	Obfuscation	Obfuscation of encrypted data stored in cHSM file system. This is allowed per IG 2.4.A example 1.
PBKDF (SP 800-132)	Derivation	Derivation of operator's password. This is allowed per IG 2.4.A example 2.

Table 7: Non-Approved, Allowed Algorithms with No Security Claimed

Non-Approved, Not Allowed Algorithms:

If a container runs the non-approved cHSM firmware 'Security Server', the container provides the non-approved, not allowed algorithms listed in Table 8 in addition to the Approved cryptographic algorithms listed in Table 4.

Name	*Use and Function*
RSA Signature (non-compliant)	Key Generation, Signature Generation/Verification, Key Wrapping (key sizes 512 - 1024); Key Generation: mechanisms other than CXI_MECH_KEYGEN_FIPS_PRIME; Sign/Verify: no padding; Sign: SHA-1, ANSI X9.31; Key Wrapping: PKCS#1 v1.5
RSA Encryption (non-compliant)	Encryption/Decryption (key sizes 512-16384)

Name	*Use and Function*
DSA Signature (non-compliant)	Key Generation, Signature Generation, DH and Domain Parameter Generation (NIST FIPS 186-4); SHA-224 (Domain Parameter Generation with SHA-224 is only possible for key length 2048/224), SHA-256, SHA-384, SHA-512
EC Cryptography (non-compliant)	Key Generation, Signature Generation/Verification, Encryption/Decryption, ECDH, Message Authentication (key sizes shorter than 224 bits, curves secp224k1, sect239k1, P-192, K-163, B-163, Sign: SHA-1, signing with deterministic k according to DCC - compliant to GBCS, not NIST FIPS 186-5)
EC Encryption with ECIES	Encryption/Decryption (all available curves)
ECDSA with K-/B-curves: K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571	Message Authentication, ECDSA Key Generation, Signature Generation, Derive Key (Shared Secret Computation)
FRP256v1 Curve	ECDSA Key Generation, Signature Generation/Verification, Key Derivation
Montgomery Curves: Curve448, Curve25519	Key Generation, ECDH
secp256k1 Curve	Message Authentication, Key Derivation
MD5, MDC-2, RIPEMD-160 (non-compliant)	Hashing
FFC Key Agreement	Diffie-Hellman key agreement based on FFC
KDF (non-compliant)	Key Derivation
DES	Encryption/Decryption, (Un-)Wrapping, Key Derivation, Split Key, Message Authentication
TDES in CBC and ECB modes	Encryption/Decryption, (Un-)Wrapping, Key Derivation, Split Key, Message Authentication via 3-key (24 bytes) and 2-key (16 bytes) constructions
TDES MAC	(Un-)Wrapping, Key Derivation, Split Key, Message Authentication via 3-key (24 bytes) and 2-key (16 bytes) constructions
TDES ANSI retail MAC	Message Authentication
AES CTR mode (non-compliant)	Encryption/Decryption, (Un-)Wrapping in yay32 implementation
AES GCM mode (non-compliant)	Encryption/Decryption, Wrapping. Non-compliant to requirements of IG C.H scenario 4.
AES GMAC with external initial IV (non-compliant)	Message Authentication
AES CBC MAC (non-compliant)	Message Authentication

Name	*Use and Function*
AES CBC with 0 IV (non-compliant)	Import DB Entry, Restore User, Encryption/Decryption, Wrapping
ECC point multiplication according to TR-03111	Shared Secret Computation on curves P-224, P-256, P-384, P-521, K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571, brainpoolP224r1/ 224t1/ 256r1/ 256t1/ 320r1/ 320t1/ 384r1/ 384t1/ 512r1/ 512t1, secp256k1, FRP256v1
Chinese Algorithms SM2, SM3, SM4 according to IETF drafts	Cryptographic functions with algorithm specifier CXI_KEY_ALGO_CUSTOM

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

The module implements the Security Functions listed in Table 9 using the Approved and Non-Approved, Allowed cryptographic algorithms.

- Cert. #A6714 is used by the glad services,
- Cert. #A6713, Cert. #A6716 and Cert. #A6717 are used by the cHSMs,
- Cert. #A6715 is used for the cHSM start-up integrity test and DRBG implementation,
- Cert. #A1563 is used for the module's start-up integrity tests.

Name	*Type*	*Description*	*Properties*	*Algorithms*
Auth_Block_Cipher	BC-Auth	Authenticated block cipher modes	Standards: NIST SP 800-38C, SP 800-38D	AES-CCM: (A6713) Key Lengths: 128, 192, 256 AES-GCM: (A6713) Directions: Encrypt, Decrypt IV Generation: Internal IV Generation Mode: 8.2.2 Key Lengths: 128, 192, 256
DRBG	DRBG	Random Bit generation and Message Digest for glad and cHSM SMOS	Standard: NIST SP 800-90A	Hash DRBG: (A6714, A6715) Prediction Resistance: No Mode: SHA2-512

Name	*Type*	*Description*	*Properties*	*Algorithms*
KAS	KAS-Full	Key Agreement Schemes and Key Derivation Functions	Standard: NIST SP 800-56A, SP 800-56C, SP 800-135 IG:IG D.F Scenario 2 path (2), split Key Confirmation:No Key Derivation: KDA (separately tested) and IG 2.4.B SP 800-135rev1 CVL Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	KAS-ECC-SSC Sp800-56Ar3: (A6713) Domain Parameter Generation Methods: P-224, P-256, P-384, P-521 KAS Role: responder Curves: brainpoolP224r1/ 224t1/ 256r1/ 256t1/ 320r1/ 320t1/ 384r1/ 384t1/ 512r1/ 512t1 KAS-ECC-SSC Sp800-56Ar3: (A6714) Domain Parameter Generation Methods: P-521 KAS Role: responder KAS-ECC-SSC Sp800-56Ar3: (A6717, A6716) Domain Parameter Generation Methods: P-224, P-256, P-384, P-521 KAS Role: responder Curves: brainpoolP224t1/ 256t1/ 320t1/ 384t1/ 512t1 KDA OneStep SP800-56Cr2: (A6713, A6714) Derived Key Length: 2048 Shared Secret Length: 224-8192 Increment 8 KDF ANS 9.63: (A6713) Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Lengths: 128, 4096 KDF TLS: (A6713) TLS Version: v1.2 Hash Algorithms: SHA2-256, SHA2-384, SHA2-512
KBKDF	KBKDF	Key-Based Key Derivation	Standards: NIST FIPS 198-1, SP 800-108	HMAC-SHA2-256: (A6713, A6714) Key Length: 256-8192 Increment 8 KDF SP800-108: (A6713, A6714) KDF Mode: Feedback Lengths: 256

Name	*Type*	*Description*	*Properties*	*Algorithms*
KeyGen	AsymKeyPair-KeyGen KAS-KeyGen CKG	Asymmetric Key Generation for cHSM and glad	Standards: NIST FIPS 186-5, SP 800-133	CKG: () Key Type: Asymmetric ECDSA KeyGen (FIPS186-5): (A6717) Curves: P-224, P-256, P-384, P-521, brainpoolP224t1/ 256t1/ 320t1/ 384t1/ 512t1 Secret Generation Mode: extra bits ECDSA KeyGen (FIPS186-5): (A6713) Curves: P-224, P-256, P-384, P-521, B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, brainpoolP224r1/ 224t1/ 256r1/ 256t1/ 320r1/ 320t1/ 384r1/ 384t1/ 512r1/ 512t1, secp256k1 Secret Generation Mode: extra bits ECDSA KeyGen (FIPS186-5): (A6714) Curve: P-521 Secret Generation Mode: extra bits ECDSA KeyGen (FIPS186-5): (A6716) Curves: P-224, P-256, P-384, P-521, brainpoolP224t1/ 256t1/ 320t1/ 384t1/ 512t1, secp256k1 Secret Generation Mode: extra bits EDDSA KeyGen: (A6713, A6716) Curves: ED-25519, ED-448 RSA KeyGen (FIPS186-5): (A6717, A6716) Key Generation Mode: probableWithProbableAux Primality Tests: 2pow100 Private Key Format: crt Key Length: 2048-8192 (even key length) RSA KeyGen (FIPS186-5): (A6713) Key Generation Mode: probableWithProbableAux Private Key Format: crt Key Length: 2048-16384 (even key length) Primality Tests: 2pow100 RSA KeyGen (FIPS186-5): (A6714) Key Generation Mode: probableWithProbableAux Primality Tests: 2pow100 Private Key Format: crt Key Length: 2048-16384 (even key length)

Name	*Type*	*Description*	*Properties*	*Algorithms*
KTS_CBC_CMAC	KTS-Wrap KTS-Unwrap	Key Transport Scheme based on AES-CBC and AES-CMAC	Standards: NIST SP 800-38A, SP 800-38B IG D.G:Approved method from IG D.G Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	AES-CBC: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-CBC: (A6714) Directions: Encrypt, Decrypt Key Length: 256 AES-CMAC: (A6713) Directions: Generation, Verification Key Lengths: 128, 192, 256 AES-CMAC: (A6714) Directions: Generation, Verification Key Length: 256
KTS_Enc	KTS-Wrap KTS-Unwrap	Key Transport scheme (with encryption, decryption, encapsulation, unencapsulation)	Standards: NIST SP 800-38D, SP 800-38F IG D.G:Approved method from IG D.G Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	AES-KW: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-KWP: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-GCM: (A6713, A6714) Directions: Encrypt, Decrypt IV Generation: Internal IV Generation Mode: 8.2.2 Key Length: 256

Name	*Type*	*Description*	*Properties*	*Algorithms*
KTS_Encap_RSA	KTS-Decap KTS-Encap	Key Transport Scheme (encapsulation, unencapsulation)	Standard: NIST SP 800-56Brev2 IG D.G:Approved method from IG D.G Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	KTS-IFC: (A6714) Moduli: 2048, 3072, 4096, 6144, 8192 Key Generation Methods: rsakpg2-crt KAS Role: responder Key Transport Method: Hash Algorithms: SHA2-256; Supports Null Associated Data Key Length: 256 Function: keyPairGen, partialVal KTS-IFC: (A6713) Moduli: 2048, 3072, 4096, 6144, 8192 Key Generation Methods: rsakpg2-crt KAS Roles: initiator, responder Key Transport Method: Hash Algorithms: SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512; Supports Null Associated Data Key Length: 1024 Function: keyPairGen, partialVal KTS-IFC: (A6717, A6716) Moduli: 2048, 3072, 4096, 6144, 8192 Key Generation Methods: rsakpg2-crt KAS Roles: initiator, responder Key Transport Method: Hash Algorithms: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512; Supports Null Associated Data Key Length: 1024 Function: keyPairGen, partialVal

Name	*Type*	*Description*	*Properties*	*Algorithms*
MAC	MAC	Message authentication generation and verification	Standards: NIST SP 800-38B, SP 800-38D, FIPS 198-1	AES-CMAC: (A6713) Directions: Generation, Verification Key Lengths: 128, 192, 256 AES-CMAC: (A6714) Directions: Generation, Verification Key Length: 256 AES-GMAC: (A6713) Directions: Encrypt, Decrypt IV Generation: Internal IV Generation Mode: 8.2.2 Key Lengths: 128, 192, 256 HMAC-SHA-1: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA2-224: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA2-256: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA2-256: (A6714) Key Length: 256-8192 Increment 8 HMAC-SHA2-384: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA2-512: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA3-224: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA3-256: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA3-384: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8 HMAC-SHA3-512: (A6713) Key Length: 112-8192 Increment 8 Additional Key Length for verification only: 80-104 Increment 8

Name	*Type*	*Description*	*Properties*	*Algorithms*
PubKeyVal	AsymKeyPair-PubKeyVal	Public Key Validation	Standard: NIST FIPS 186-5	ECDSA KeyVer (FIPS186-5): (A6713) Curves: P-224, P-256, P-384, P-521, B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, brainpool P224r1/224t1/256r1/256t1/320r1/320t1/384r1/384t1/512r1/512t1 ECDSA KeyVer (FIPS186-5): (A6714) Curve: P-521 ECDSA KeyVer (FIPS186-5): (A6716, A6717) Curves: P-224, P-256, P-384, P-521, brainpool P224t1/256t1/320t1/384t1/512t1 ECDSA KeyVer (FIPS186-4): (A6713) Curves: P-192, B-163, K-163 ECDSA KeyVer (FIPS186-4): (A6716, A6717) Curve: P-192 EDDSA KeyVer: (A6713, A6716) Curves: ED-25519, ED-448
SHS	SHA	Message Digest	Standards: NIST FIPS 202, FIPS 180-4	SHA-1: (A6713) Message Length: 0-65536 Increment 8 SHA2-224: (A6713) Message Length: 0-65536 Increment 8 SHA2-256: (A6714, A6713) Message Length: 0-65536 Increment 8 SHA2-384: (A6713) Message Length: 0-65536 Increment 8 SHA2-512: (A6714, A6715, A6713) Message Length: 0-65536 Increment 8 SHA3-224: (A6713) Message Length: 0-65536 Increment 8 SHA3-256: (A6713) Message Length: 0-65536 Increment 8 SHA3-384: (A6713) Message Length: 0-65536 Increment 8 SHA3-512: (A6713) Message Length: 0-65536 Increment 8 SHA2-256: (A1563) Message Length: 0-65528 Increment 8

Name	*Type*	*Description*	*Properties*	*Algorithms*
SigGen	DigSig-SigGen	Digital Signature Generation	Standard: NIST FIPS 186-5	ECDSA SigGen (FIPS186-5): (A6713) Curves: P-224, P-256, P-384, P-521, B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, brainpoolP224r1/ 224t1/ 256r1/ 256t1/ 320r1/ 320t1/ 384r1/ 384t1/ 512r1/ 512t1, secp256k1 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 ECDSA SigGen (FIPS186-5): (A6714) Curve: P-521 Hash Algorithm: SHA2-256 ECDSA SigGen (FIPS186-5): (A6716) Curves: P-224, P-256, P-384, P-521, brainpoolP224t1/ 256t1/ 320t1/ 384t1/ 512t1, secp256k1 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 ECDSA SigGen (FIPS186-5): (A6717) Curves: P-224, P-256, P-384, P-521, brainpoolP224t1/ 256t1/ 320t1/ 384t1/ 512t1 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 EDDSA SigGen: (A6713, A6716) Curves: ED-25519, ED-448 Hash Algorithms: SHA2-512, SHAKE256 RSA SigGen (FIPS186-5): (A6713) Signature Types: PKCS 1.5, PKCSPSS Key Length: 2048-16384, even key length only RSA SigGen (FIPS186-5): (A6716, A6717) Signature Types: PKCS 1.5, PKCSPSS Key Length: 2048-8192, even key length only SHAKE-256: (A6713) Output Length: 16-65536 Increment 8

SigVer	DigSig-SigVer	Digital Signature Verification	Standards: NIST FIPS 186-2, FIPS 186-4, FIPS 186-5	DSA SigVer (FIPS186-4): (A6713) (L, N): (1024, 160), (2048, 224), (2048, 256), (3072, 256) Hash Algorithms: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 RSA SigVer (FIPS186-2): (A6713, A6716, A6717) Signature Types: PKCS 1.5, PKCSPSS, ANSI X9.31 Key Length: 1024-4096 (even key length only) RSA SigVer (FIPS186-4): (A6713) Signature Types: PKCS 1.5, PKCSPSS, ANSI X9.31 Key Lengths: 1024-2048 for PKCS 1.5 and PKCSCSS (even key length only), 1024-16384 for ANSI X9.31 (even key length only) RSA SigVer (FIPS186-4): (A6716, A6717) Signature Types: PKCS 1.5, PKCSPSS, ANSI X9.31 Key Lengths: 1024-2048 for PKCS 1.5 and PKCSCSS (even key lengths only), 1024-8192 for ANSI X9.31 (even key lengths only) ECDSA SigVer (FIPS186-4): (A6713) Curves: B-163, K-163, P-192 Hash Algorithm: SHA-1 ECDSA SigVer (FIPS186-4): (A6716, A6717) Curve: P-192 Hash Algorithm: SHA-1 ECDSA SigVer (FIPS186-5): (A6713) Curves: P-224, P-256, P-384, P-521, B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, brainpoolP224r1/ 224t1/ 256r1/ 256t1/ 320r1/ 320t1/ 384r1/ 384t1/ 512r1/ 512t1, secp256k1 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 ECDSA SigVer (FIPS186-5): (A6714) Curves: P-256, P-521, brainpoolP320t1 Hash Algorithm: SHA2-256 ECDSA SigVer (FIPS186-5): (A6716) Curves: P-224, P-256, P-384, P-521 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512, brainpoolP224r1/ 224t1/ 256t1/ 320t1/ 384t1/ 512t1, secp256k1 ECDSA SigVer (FIPS186-5): (A6717) Curves: P-224, P-256, P-384, P-521, brainpoolP224r1/ 224t1/ 256t1/ 320t1/ 384t1/ 512t1 Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512 EDDSA SigVer: (A6713, A6716)

Name	Type	Description	Properties	Algorithms
				Curves: ED-25519, ED-448 Hash Algorithms: SHA2-512, SHAKE256 RSA SigVer (FIPS186-5): (A6713) Signature Types: PKCS 1.5, PKCSPSS Key Length: 2048-16384 (even key length only) RSA SigVer (FIPS186-5): (A6714) Signature Types: PKCSPSS Key Length: 2048-16384 (even key length only) RSA SigVer (FIPS186-5): (A6716, A6717) Signature Types: PKCS 1.5, PKCSPSS Key Length: 2048-8192 (even key length only) SHAKE-256: (A6713) Output Length: 16-65536 Increment 8
SKG	CKG	Symmetric Key Generation	Standard: NIST SP 800-133	CKG: () Key Type: Symmetric Hash DRBG: (A6714, A6715)
Unauth_Block_Cipher	BC-UnAuth	Unauthenticated block cipher modes	Standard: NIST SP 800-38A	AES-CBC: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-ECB: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-OFB: (A6713) Directions: Encrypt, Decrypt Key Lengths: 128, 192, 256 AES-CBC: (A6714) Directions: Encrypt, Decrypt Key Length: 256

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

The module's AES-GCM implementation conforms to IG C.H scenario 2. The module uses the approved DRBG (Cert. #A6714 and Cert. #A6715) to generate the IV with a length of 96-bits. The entropy source producing the DRBG seed is located inside the module's cryptographic boundary.

The module complies with IG D.M. All of the keys generated through KDF SP 800-108 (Cert. #A6713 and Cert. #A6714) are derived from keys that have been generated using an approved method or CKG per NIST SP 800-133r2 in the approved mode. The encoding of the data fields (i.e., Label, Context, and Length L) is compliant to Section 6.4 of NIST SP 800-133r2. No asymmetric keys are generated through KBKDF.

The module's use of elliptic curves conforms to IG C.A. Table 9 lists the approved curves while Tables 6 and 8 list the non-approved curves. The curve secp256k1 listed in Table 9 is only used for blockchain-related applications. EDDSA-related algorithms (Cert. #A6713, Cert. #A6717 and Cert. #A6716) are used only with Edwards25519 and Edwards448 curves.

The module's TLS 1.2 KDF can only be performed in the context of the TLS protocol.

Algorithms designated as "Legacy" can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M. These legacy algorithms are SHA-1 for DSA and ECDSA Signature Verification, HMAC Verification, and RSA Signature Verification.

Compliance to NIST SP 800-56Ar3 Assurances

For KAS ECC, the module satisfies IG D.F Scenario 2 path (2). The key derivation functions comply with NIST SP 800-56Cr2 (i.e., KDA OneStep KDF Cert. #A6713 and Cert. #A6714), NIST SP 800-108 (i.e., Feedback mode KDF Cert. #A6713 and Cert. #A6714), and KDF ANS 9.63 (Cert. #A6713). Furthermore, the module obtains the appropriate assurances as required in Section 5.6.2 of NIST SP 800-56Ar3. For KAS-ECC, the module uses C(2e,0s), thus no static key pairs are used as a part of the KAS schemes per NIST SP 800-56Ar3. Full public key validations are implemented (NIST SP 800-56Ar3 Section 5.6.2.3.3). No key confirmation is implemented.

Compliance to NIST SP 800-56Br2 Assurances

For KTS RSA, the module satisfies IG D.G. The RSA keys for KTS-IFC (Cert. #A6713, Cert. #A6714, Cert. #A6716 and Cert. #A6717) all have lengths of at least 2048 bits, which are acceptable per NIST SP 800-131Ar2. The modulus sizes, the method for key generation, and the roles (initiator, i.e., the module performs encapsulation, or responder, i.e., the module performs unencapsulation) are specified in Table 9. The RSA key generation algorithms are CAVP validated (Cert. #A6713, Cert. #A6714, Cert. #A6716 and Cert. #A6717). No key confirmation is implemented.

The module implements the KTS-OAEP key transport scheme and obtains the appropriate assurances as required in Sections 5 and 6 of NIST SP 800-56Br2.The module performs partial public-key validation. No key confirmation is implemented.

2.8 RBG and Entropy

The module incorporates a NIST SP 800-90A Hash DRBG (Cert. #A6714 and Cert. #A6715) that is seeded from the module's NIST SP 800-90B validated entropy source. The unmodified output of the DRBG is used for generating cryptographic key material. The DRBG is instantiated anew every power-on. The threshold for reseeding is monitored to not exceed 2^{48} per SP 800-90A.

A public user document for the entropy source is available at https://csrc.nist.gov/projects/cryptographic-module-validation-program/entropy-validations/certificate/108. This physical entropy source is used to seed and reseed the approved deterministic bit generators (DRBGs) for glad as well as for cHSMs.

Cert Number	*Vendor Name*
E108	Utimaco

Table 10: Entropy Certificates

Name	*Type*	*Operational Environment*	*Sample Size*	*Entropy per Sample*	*Conditioning Component*
Utimaco Entropy Source	Physical	u.trust Anchor	512	417.28	LFSR

Table 11: Entropy Sources

2.9 Key Generation

The module supports the generation of cryptographic keys and components with a security strength of at least 256 bits.

2.10 Key Establishment

The module supports key establishment using KAS with a supported security strength of 112-256 bits depending on the selection of key sizes and hash sizes utilized. The module supports AES-based key transport with a supported security strength 128-256 bits depending on the selection of key sizes and hash sizes utilized. The module supports RSA-based key transport with a supported security strength of 112-256 bits depending on the selection of key sizes and hash sizes utilized.

2.11 Industry Protocols

No industry protocols are used in this module.

3 Cryptographic Module Interfaces

For the communication with a host, the PCIe board offers a PCIe interface and a serial log interface. Figure 1 shows the module on its carrier board with PCIe interface.

Together with Utimaco's appropriate host application software, the cHSMs also provide cryptographic standard interfaces such as PKCS#11, JCE, OpenSSL, CSP/CNG and EKM.

A Secure Messaging concept uses key agreement followed by encryption and MAC to protect communication to and from the cryptographic module. The communication that is protected covers the Global Administrator command interface and the cHSM command interface in both the Approved mode and in the Non-Approved modes of operation.

3.1 Ports and Interfaces

The module incorporates physical ports and logical interfaces defined in Table 12.

Physical Port	*Logical Interface(s)*	*Data That Passes*
Battery Measuring Inputs	Data Input	Measurement Data
External communication port (PCIe)	Data Input Data Output Control Input Status Output	SSPs, Operator data, Command, Return codes, Status, Diagnostic information.
External Erase button	Control Input	Control signal to zeroize all security relevant information inside the module
LED	Status Output	Indicator that the External Erase button is pressed
Power	Power	N/A
Serial output	Status Output	Status output like boot sequence output during startup.
USB interfaces	Status Output	Status output like boot sequence output during startup.
SMBUS interface	Control Input Status Output	Status requests and output for device manager such as temperature, versioning, and PCIe state

Table 12: Ports and Interfaces

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module supports authentication methods for the Cryptographic Officer (CO) and User roles. These roles have separate identity-based authentication methods as indicated in Table 13. The module supports three authentication methods: Password, RSA Signature, and ECDSA Signature. For each authentication attempt, the operator must present their unique identifier. The methods are in conformance with NIST SP800-140E and SP 800-63B (refer to Sections 5.1.1 and 5.1.6). The operators must update the relevant default authentication SSPs upon authentication to the module for the first time.

Method Name	*Description*	*Security Mechanism*	*Strength Each Attempt*	*Strength per Minute*
Password	The operator password is a minimum of 8 characters chosen from 94 printable ASCII characters.	N/A	The probability that a random attempt will succeed, or a false acceptance will occur, is 1/(94^8), which is less than 1/1,000,000.	Due to a correctional delay of 120 milliseconds for every non-successful authentication on a cHSM, there is a maximum limit of 60 * 1000 / 120 = 500 non-successful authentications per minute. This can be stated as allowing only 500 non-successful authentication attempts per minute based on a rate of 120 ms per attempt. Therefore, the probability of successfully authenticating to the module within one minute is (less than) 500 * 1/(94^8), which is less than 1/100,000.
ECDSA Signature	The operator sends an ECDSA signed command containing its username to authenticate to the cryptographic module. (key size >= 224)	SigVer	The probability that a random attempt will succeed, or a false acceptance will occur, is less than or equal to approximately 1/(2^112) (according to NIST SP 800-57-Part1 Table 2) which is less than 1/1,000,000.	Assuming a duration of at least 1 microsecond for each signature verification, there is a maximum limit of 60 * 1,000,000 authentication attempts per minute. Therefore, the probability of successfully authenticating to the module within one minute is less than 60 * 1,000,000 * [1/(2^112)] which is less than 1/100,000.

Method Name	*Description*	*Security Mechanism*	*Strength Each Attempt*	*Strength per Minute*
RSA Signature	The operator sends an RSA signed command containing its username to authenticate to the cryptographic module. (key size >= 2048)	SigVer	The probability that a random attempt will succeed, or a false acceptance will occur, is less than or equal to approximately 1/(2^112) (according to NIST SP 800-57-Part1 Table 2), which is less than 1/1,000,000.	Assuming a duration of at least 1 microsecond for each signature verification, there is a maximum limit of 60 * 1,000,000 authentication attempts per minute. Therefore, the probability of successfully authenticating to the module within one minute is less than 60 * 1,000,000 * [1/(2^112)] which is less than 1/100,000.

Table 13: Authentication Methods

4.2 Roles

The module supports a Cryptographic Officer (CO) role, a User role, and an Unauthenticated role. The Unauthenticated role has access to non-security-relevant services only.

Name	*Type*	*Operator Type*	*Authentication Methods*
cHSM User	Identity	User	Password RSA Signature ECDSA Signature
Global Administrator (glad)	Identity	Crypto Officer (CO)	RSA Signature ECDSA Signature
Unauthenticated	Role	N/A	None

Table 14: Roles

Global Administrator operators use the gladm tool to access the Global Administrator Application glad. cHSM administrators use the csadm tool to access their cHSM. The operators must update the default authentication keys when they authenticate to the module for the first time. See Section 11.1 for details.

The module supports multiple concurrent operators separated by unique sessions. Operator authentication does not persist beyond power-cycling the module. The selection of roles is implicit.

4.3 Approved Services

The module supports the following approved services listed in Table 15. Note that all services invoked by an authenticated operator (i.e., one that takes on the CO role or the User role) are protected by the Secure Messaging concept described in Section 3.

In the description, inputs, and outputs columns in Table 15, the term “object” may be a cryptographic key, a storage object, or a configuration object within a cHSM. The term “backup blob” contains one or more objects.

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Access_Secrets	Internal service for accessing cHSM keys and User's secrets and keys, i.e., ALK, CAK, MBK, PSW_AUTH, U_S, U_KM, U_Legacy, and U_PRIV.	cHSM_Status output	None	None	KBKDF KTS_Enc KTS_CBC_CMAC	cHSM User - CMK: G,E - MBK: E - MBKDK: G,E - ALK: E - CAK: E - PSW_AUTH: E - U_S: E - U_KM: E - U_Legacy: E - U_PRIV: E
cHSM_Allocate_Key	Cuts keying material, deleting original key from the database. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key handle, key templates	Key handles or backup blobs	KTS_CBC_CMAC	cHSM User - MBKDK: E - U_KM: E,G,R - U_S: G,R
cHSM_Compute_Hash	Compute SHA-1/2/3 or HMAC of a given value. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Data or secret blob, mechanism parameters, (optional key)	Hash or HMAC	MAC SHS	cHSM User - U_KM: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Config	1. Delete parts of audit log 2. Return all search keys of given database 3. Outputs information about all master backup keys stored in the cHSM 4. Import a customer certificate for the CAK 5. Toggle Administration-Only mode (blocks all cryptographic services) 6. Set configuration parameters 7. Set maximum number of allowed consecutive failed authentication attempts 8. Set startup mode	cHSM_Status output	1. parameter 2. database name 3. N/A 4. certificate over CAK+ 5. new mode 6. module ID and config data 7. maximum number of consecutive failed authentication attempts 8. new startup mode	1. N/A 2. search keys of given database 3. information (key type, key size, key check value, etc.) about all Master Backup Keys that are stored inside the cHSM 5. N/A 6. N/A 7. N/A 8. N/A	None	cHSM User - CAK+: W
cHSM_Crypt_Data	Encrypts or Decrypts data with AES key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key handle or backup blob, data, mechanism parameter	Encrypted or decrypted data	Auth_Block_Cipher Unauth_Block_Cipher DRBG KTS_CBC_CMAC	cHSM User - U_S: W,E - S_DRBG: G,E - MBKDK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Delete_Key	Deletes an assigned object from the module	cHSM_Status output	Object handle or property list	N/A	None	cHSM User - U_S: Z - U_KM: Z - U_Legacy: Z - U_PRIV: Z - U_PUB: Z - U_DSA_PUB: Z
cHSM_Derive_Key	Derive cHSM user's secrets and keys and store in cHSM or a Backup Blob. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Base key, key template and mechanism parameter	Key handle or Backup Blob	PubKeyVal KAS KBKDF KTS_CBC_CMAC	cHSM User - Shared Secret: G,E,Z - U_S: E,R,W,G - U_KM: E,R,W,G - U_PRIV: E - U_PUB: E - MBKDK: E
cHSM_End_Session	End secure messaging session by zeroizing session key	cHSM_Status output	Session ID	N/A	None	cHSM User - SMEK: Z - SMMK: Z Unauthenticated - SMEK: Z - SMMK: Z
cHSM_Generate_Audit_Log_Key	Generate and store audit log key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key type	ALK+	KeyGen SigGen	cHSM User - ALK+: G,R - ALK: G,E
cHSM_Generate_Master_Backup_Key	Generates and outputs master backup key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key attributes	Encrypted Master Backup Key	DRBG SKG KTS_CBC_CMAC	cHSM User - MBK: G,R - SMEK: E - SMMK: E - S_DRBG: G,E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Generate_Random_Number	Generates a random number using DRBG	cHSM_Status output	Flags, Mechanism parameters	Random bytes	DRBG	cHSM User - S_DRBG: G,E
cHSM_Generate_User_Keys	Generate a key or key pair. Store key(s) in object(s) or backup blob. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key template, Mechanism parameters	Key handle or Backup Blob	DRBG KeyGen SKG KTS_CBC_CMAC	cHSM User - S_DRBG: G,E - U_S: G,R - U_PRIV: G,R - U_PUB: G,R - U_KM: G,R - MBKDK: E
cHSM_Get_Audit_Log_Key	Retrieve the public audit log signature key.	cHSM_Status output	N/A	ALK+	SigGen KTS_Enc	cHSM User - CMK: E - ALK+: R - ALK: E Unauthenticated - CMK: E - ALK+: R - ALK: E
cHSM_Get_Key_Property	Get properties of an object. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Object handle or property list	Property list, user public keys	None	cHSM User - U_PUB: R - U_DSA_PUB: R
cHSM_Get_Signed_Audit_Log	Return audit file signed by audit log signature key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Index	Signed Audit Log file	SigGen	cHSM User - ALK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Import_Master_Backup_Key	Import master backup key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Encrypted MBK, Slot Number	N/A	KTS_CBC_CMAC	cHSM User - MBK: W - SMEK: E - SMMK: E
cHSM_Init_Key_Group	Delete all local objects belonging to a key group	cHSM_Status output	Key Group	Number of deleted objects	None	cHSM User - U_S: Z - U_KM: Z - U_Legacy: Z - U_PRIV: Z - U_PUB: Z - U_DSA_PUB: Z
cHSM_Manage_Files	Load or remove files	cHSM_Status output	New file, new location or file description pattern	N/A	SigVer	cHSM User - AMSK+: W,E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Login	Set up secure messaging and authenticate cHSM operator. Generate new secure messaging session keys. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Authentication data, SMRK+	Session ID, SMLK+, certificate chain	DRBG SigGen SigVer KAS PubKeyVal KeyGen KBKDF Unauth_Block_Cipher MAC KTS_CBC_CMAC	Unauthenticated - S_DRBG: G,E - SMRK+: W,E - SMLK: G,E - SMLK+: G,R - DAK+: R - CAK+: R - CAAK+: E - PSW_AUTH: E - CAK: E - SMKDK: G,E - Shared Secret: G,E,Z - SMEK: G,E - SMMK: G,E
cHSM_Manage_Database_Backups	Manage import/export of Backup Blobs which contain an Object for backup purposes. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Backup Blob and key template/Object handle or database name	Object handle or backup blob/Backup blob	KTS_CBC_CMAC Unauth_Block_Cipher MAC	cHSM User - MBKDK: E - U_S: R,W - U_KM: R,W - U_Legacy: R,W - U_PRIV: R,W - U_PUB: R,W - U_DSA_PUB: R,W - ALK: R,W - ALK+: R,W

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Manage_Object	Import or copy an assigned key or storage object. The object is stored either in the cHSM or an exported Backup Blob. Call internal service cHSM_Access_Secrets.	cHSM_Status output	key handle or backup blob, key template	key handle or backup blob	KTS_CBC_CMAC PubKeyVal	cHSM User - U_S: R,W - U_KM: R,W - U_PRIV: R,W - U_PUB: R,W - MBKDK: E
cHSM_Manage_User_Backups	1. Import or 2. export all user account data for a given cHSM operator for backup purposes. Call internal service cHSM_Access_Secrets.	cHSM_Status output	1. Backup data containing CAAK+ or PSW_AUTH 2. Username	1. None 2. Backup data containing CAAK+ or PSW_AUTH	KTS_CBC_CMAC Unauth_Block_Cipher MAC	cHSM User - CAAK+: W,R - PSW_AUTH: W,R - MBKDK: E
cHSM_Manage_Users	Add or delete a cHSM user or change operator's key or password (resets failed authentication attempts counter). Call internal service cHSM_Access_Secrets.	cHSM_Status output	username, user attributes, CAAK+, PSW_AUTH, permissions	short username	None	cHSM User - CAAK+: W - PSW_AUTH: W
cHSM_Migrate_Key	Migrate keys from one release to another, i.e., change an internal property of Edwards keys. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key handle or backup blob	Key handle or backup blob	KTS_CBC_CMAC	cHSM User - MBKDK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Open_Key	Opens an assigned object and returns a reference or backup blob containing the object. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Object Attributes	Reference or Backup Blob	KTS_CBC_CMAC	cHSM User - MBKDK: E - U_S: R - U_KM: R - U_Legacy: R - U_PRIV: R - U_PUB: R - U_DSA_PUB: R
cHSM_Set_Key_Property	Set one or more properties for a key or storage object or local or global configuration object. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Object handle, Property list	Object handle or backup blob	KTS_CBC_CMAC	cHSM User - MBKDK: E
cHSM_Sign_Data	Generates signature or calculates MAC for given data with an assigned signing key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key handle or backup blob, data, mechanism parameter	Signature or MAC or HMAC	MAC SigGen DRBG KTS_CBC_CMAC	cHSM User - U_S: E - U_KM: E - U_PRIV: E - S_DRBG: G,E - MBKDK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Status	1. Echo 2. Get any of the following: audit log, authentication state, boot log, CXI Information, Configuration Parameters, Maximum Consecutive Failure Counter, RAM usage information, startup mode, current status of the cHSM including approved mode indicator, current internal time of cHSM clock, and user information 3. Lists any of the following: active modules, files stored in cHSM, all current users, and information about users for a given key group, properties of user keys accessible by the user	cHSM_Status output	1. data to be echoed 2. Any of the following: log file number, module ID, or username. 3. Any of the following lists: type of files, key group or property list for filtering	1. data 2. Any of the information requested 3. Any of the lists requested including user and permission list, list of logged in users	None	cHSM User - CTIME: R Unauthenticated - CTIME: R
cHSM_Time	Set internal time of cHSM clock	cHSM_Status output	CTIME or relative time	N/A	None	cHSM User - CTIME: W,G

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_Verify_Signature	Verify signature or MAC or HMAC. Call internal service cHSM_Access_Secrets.	cHSM_Status output	Key handle or backup blob, mechanism parameter, signature or MAC or HMAC, data or reference hash value	Verification result, error in case of a wrong signature	MAC SigVer KTS_CBC_CMAC	cHSM User - U_PUB: E - U_DSA_PUB: E - U_S: E - U_KM: E - U_Legacy: E - MBKDK: E
cHSM_Wrap_Unwrap_User_Keys	Export/wrap key. Import/unwrap key. Call internal service cHSM_Access_Secrets.	cHSM_Status output	key (optional wrapping key and mechanism parameter), key blob or key template	key blob, key handle or blob	Auth_Block_Cipher Unauth_Block_Cipher DRBG KTS_Enc KTS_Encap_RSA KTS_CBC_CMAC PubKeyVal	cHSM User - U_S: R,W,E - U_KM: R,W - U_Legacy: R,W - U_PRIV: R,W,E - U_PUB: R,W,E - U_DSA_PUB: R,W - MBKDK: E - S_DRBG: G,E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
cHSM_restart_and_self_tests	Restart cHSM and initiate all cHSM self-tests	cHSM_Status output	N/A	N/A	DRBG KBKDF KTS_Enc SigGen	cHSM User - DRBG_Entropy_Input: G,E - DRBG_Seed: G,E - S_DRBG: G,E - DMK: E - DEK: G,E - CBK: E,Z - CMK: Z,G - SDMK: E - SDEK: G,E - DAK: E - CAK: E - CTIME: G Unauthenticated - DRBG_Entropy_Input: G,E - DRBG_Seed: G,E - S_DRBG: G,E - DMK: E - DEK: G,E - CBK: E,Z - CMK: Z,G - SDMK: E - SDEK: G,E - DAK: E - CAK: E - CTIME: G

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_Access_Secrets	Internal service for accessing glad-accessible secrets and keys, i.e., CBK, VBS, OBS, OBSEK, GAK, and DAK.	glad_Status output	None	None	KBKDF KTS_Enc	Global Administrator (glad) - DMK: E - DEK: G,E - SDMK: E - SDEK: E - CBK: E - OBS: E - OBSEK: E - GAK: E - VBS: E - DAK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_Config	1. Get system audit log 2. Truncate audit log 3. Fetch boot log of a cHSM 4. List operator secrets 5. Get/configure cHSM resource quota 6. Fetch and delete system log 7. Get license information 8. Get system metrics 9. Get/configure system quota 10. list cHSM templates 11. set quorum requirements 12. update system status	glad_Status output	1. - 2. hash of most recent audit log entry to be deleted 3. cHSM slot 4. - 5. cHSM slot/ cHSM slot and quota 6. - 7. - 8. - 9. -/new system quota 10. - 11. quorum requirements 12. -	1. Glad system audit log 2. - 3. cHSM boot log 4. fingerprints of all operator secrets on device 5. cHSM resource quota/- 6. system log 7. license file 8. system metrics 9. system quota/- 10. list of available cHSM templates 11. - 12. update status	None	Global Administrator (glad)

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_End_Session	Terminate current secure messaging session	glad_Status output	Session ID	N/A	None	Global Administrator (glad) - SMEK: Z - SMMK: Z Unauthenticated - SMEK: Z - SMMK: Z
glad_Login_Session	Set up secure messaging and authenticate multiple glad users. Call internal service glad_Access_Secrets.	glad_Status output	Names of users, SMRK+, Signatures	Authentication mechanisms, SMLK+, Session ID, certificates	DRBG KAS PubKeyVal KeyGen KBKDF SigGen SigVer Unauth_Block_Cipher MAC KTS_CBC_CMAC	Unauthenticated - S_DRBG: G,E - GAAK+: E - GAK: E,G - GAK+: G,R - DAK: E - DAK+: R - SMRK+: W,E - SMLK: G,E - SMLK+: G,R - SMKDK: G,E - Shared Secret: G,E,Z - SMEK: G,E - SMMK: G,E
glad_Manage_User_Backup	Create or restore an encrypted backup of a glad user. Call internal service glad_Access_Secrets.	glad_Status output	If restoring: backup blob of a glad user	If creating: backup blob of a glad user	KBKDF Unauth_Block_Cipher MAC	Global Administrator (glad) - OBS: E - USMK: G,E - GAAK+: R,W - SMEK: E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_Manage_Users	Add, delete, change credentials of global admin	glad_Status output	username and credentials (GAAK+) for adding or changing a user; username for deleting a user	N/A	SHS	Global Administrator (glad) - GAAK+: W

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_Manage_cHSM_snapshots	1. Restore a cHSM from a snapshot and clone to specified cHSM slot(s). 2. Create an encrypted cHSM snapshot. Call internal services glad_Access_Secrets in both cases.	glad_Status output	1. cHSM snapshot and cHSM slot(s) 2. N/A	1. N/A 2. cHSM snapshot	KBKDF KTS_Enc KTS_CBC_CMAC SigGen MAC DRBG	Global Administrator (glad) - PSW_AUTH: W,R - DEK: E - CBK: W,R,E,Z - CMK: G,E - MBK: W,R - CAAK+: W,R - CAK: W,R,E - CAK+: W,R - ALK: W,R - ALK+: W,R - VBS: E - OBS: E - CBKEK: G,E - SEK: G,E - SMK: G,E - DAK: E - U_S: W,R - U_KM: W,R - U_Legacy: W,R - U_PRIV: W,R - U_PUB: W,R - U_DSA_PUB: W,R - S_DRBG: G,E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_Status	1. List cHSM slots or glad users 2. Get any of the following: session information, system quorum requirements, system time, system information and approved mode indicator, or system metrics.	glad_Status output	1. N/A 2. N/A	1. Information about cHSM slots or list of all glad users 2. Any of the following: session users and permissions, required quorum for each command, DTIME, status and version information and approved mode indicator, or system metrics.	None	Global Administrator (glad) - DTIME: R Unauthenticated - DTIME: R

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_System_Reset_Alarm	Reset system alarm state	glad_Status output	N/A	N/A	KBKDF DRBG SKG KeyGen KTS_Enc	Global Administrator (glad) - DMK: G - DAK: G - DAK+: G - SDMK: G,E - SDEK: G,E
glad_Time	1. Toggle ntp time adjustment 2. Set ntp configuration 3. Set device system time 4. Get ntp configuration	glad_Status output	1. flag for activation 2. flag for activation and maximum values for time adjustment 3. DTIME 4. N/A	1. N/A 2. N/A 3. N/A 4. ntp configuration file	None	Global Administrator (glad) - DTIME: W,G - CTIME: G

glad_Zeroize	Clear all system data. For more details about the zeroization methods, see "9.3 SSP Zeroization Methods."	glad_Status output	N/A	N/A	None	Global Administrator (glad) - SDMK: Z - SDEK: Z - DMK: Z - DEK: Z - VBS: Z - DAK: Z - OBS: Z - OBSEK: Z - CBK: Z - CMK: Z - CAK: Z - CBKEK: Z - SEK: Z - SMK: Z - USMK: Z - MBK: Z - MBKDK: Z - GAK: Z - PSW_AUTH: Z - ALK: Z - U_S: Z - U_KM: Z - U_Legacy: Z - U_PRIV: Z - SMLK: Z - SMKDK: Z - SMEK: Z - SMMK: Z - S_DRBG: Z - DRBG_Entropy_In

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
						put: Z - DRBG_Seed: Z
glad_chsm_create	Create new cHSM from a template. Call internal service glad_Access_Secrets.	glad_Status output	cHSM slot, CIAK+, template	Certificates over CIAK+, DAK+, GAK+	DRBG KeyGen SKG KBKDF KTS_Enc SigGen	Global Administrator (glad) - CBK: G,E,Z - CAK: G,E - MBK: G - CMK: G,E - GAK: E - DAK: E - DMK: E - DEK: G,E - GAK+: R - DAK+: R - CAK+: R - CIAK+: W,R - S_DRBG: G,E
glad_chsm_delete	Remove a cHSM from the module and remove its associated data. The cHSM must be halted before it can be removed.	glad_Status output	cHSM slot	N/A	None	Global Administrator (glad) - PSW_AUTH: Z - U_S: Z - U_KM: Z - U_Legacy: Z - U_PRIV: Z - CAK: Z - CMK: Z - ALK: Z - MBK: Z - MBKDK: Z

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_device_restart_and_self_tests	Restart the device and initiate the glad self-tests. If glad_system_update was performed beforehand, glad_device_restart_and_self_tests activates the new image.	glad_Status output	N/A	N/A	DRBG KBKDF KTS_Enc MAC SHS SigGen SigVer	Global Administrator (glad) - DRBG_Entropy_Input: G,E - DRBG_Seed: G,E - S_DRBG: G,E - DMK: E - DEK: G,E - CBK: E - SDMK: E - SDEK: G,E - DAK: E - GAK+: E - CAK+: E - ISK+: G,E Unauthenticated - DRBG_Entropy_Input: G,E - DRBG_Seed: G,E - S_DRBG: G,E - DMK: E - DEK: G,E - CBK: E - SDMK: E - SDEK: G,E - DAK: E - GAK+: E - CAK+: E - ISK+: G,E

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_key_get_csr	Get certificate signing request for the device authentication key. Call internal service glad_Access_Secrets.	glad_Status output	N/A	CSR for the DAK	SigGen	Global Administrator (glad) - DAK: E - DAK+: R
glad_key_get_wrapping_key	Generate and store the RSA OAEP wrapping key. Call internal service glad_Access_Secrets.	glad_Status output	Key size	Certificate over OBSEK+	KeyGen SigGen	Global Administrator (glad) - OBSEK: G - OBSEK+: G,R - DAK: E
glad_key_import_cert	Import an operator DAK certificate	glad_Status output	Operator DAK certificate	N/A	None	Global Administrator (glad) - DAK+: W
glad_manage_operator_secret	Delete operator secret or import a new wrapped operator secret and mark it as active. Call internal service glad_Access_Secrets.	glad_Status output	Fingerprint of OBS, OBS wrapped with OBSEK	Fingerprint of OBS	KTS_Encap_RSA	Global Administrator (glad) - OBS: W - OBSEK: E,Z

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_system_restart_and_self_tests	Restart the device and initiate the glad self-tests. If glad_system_update was performed beforehand, glad_system_restart activates the new image.	glad_Status output	N/A	N/A	DRBG KBKDF KTS_Enc MAC SHS SigGen SigVer	Global Administrator (glad) - DRBG_Entropy_Input: G,E - DRBG_Seed: G,E - S_DRBG: G,E - DMK: E - DEK: G,E - CBK: E - SDMK: E - SDEK: G,E - DAK: E - GAK+: E - CAK+: E - ISK+: G,E
glad_system_set_time_delay	Set delta time delay	glad_Status output	Flag for setting time (backwards or forwards), time delta	N/A	None	Global Administrator (glad) - DTIME: G - CTIME: G Unauthenticated - DTIME: G - CTIME: G

Name	*Description*	*Indicator*	*Inputs*	*Outputs*	*Security Functions*	*SSP Access*
glad_system_update	Update the device firmware including operational bootloader, glad firmware, and cHSM templates. Existing cHSMs will be deleted.	glad_Status output	System image binary	N/A	SigVer KTS_CBC_CMAC	Global Administrator (glad) - GIAK+: W - ISK+: W,E - SMEK: E - SMMK: E - PSW_AUTH: Z - U_S: Z - U_KM: Z - U_Legacy: Z - U_PRIV: Z - CAK: Z - CMK: Z - ALK: Z - MBK: Z - MBKDK: Z

Table 15: Approved Services

4.4 Non-Approved Services

In the Non-Approved mode, the module supports all services from the Approved mode with the additional services and modifications listed in Table 16.

Name	*Description*	*Algorithms*	*Role*
cHSM: Agree Secret	Calculate a shared secret from two ECDH or ECDSA keys as described in TR-03111.	ECC point multiplication according to TR-03111	cHSM User
cHSM: Allocate Key using non-approved, not allowed key types or mechanisms	cHSM_Allocate_Key service with the following key types or sizes is allowed only in non-approved mode: symmetric key of size < 112.	DES TDES in CBC and ECB modes	cHSM User
cHSM: any authenticated services without Secure Messaging	In non-approved mode, it is possible to execute an authenticated command without Secure Messaging.		cHSM User
cHSM: any services involving authentication using non-approved curves for ECDSA	In non-approved mode, all available curves can be used during cHSM User authentication.	EC Cryptography (non-compliant) ECDSA with K-/B-curves: K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571 FRP256v1 Curve secp256k1 Curve	cHSM User
cHSM: Compute Hash using non-approved, not allowed key types or mechanisms	cHSM_Compute_Hash service with the following key types or sizes is allowed only in non-approved mode: DES, RAW with size < 112	MD5, MDC-2, RIPEMD-160 (non-compliant) DES TDES in CBC and ECB modes TDES MAC	cHSM User

Name	*Description*	*Algorithms*	*Role*
cHSM: Crypt Data using non-approved, not allowed key types or mechanisms	cHSM_Crypt_Data service with the following key types or sizes is allowed only in non-approved mode: DES, TDES in CBC and ECB modes, AES CTR, AES CBC with 0 IV, GCM with initial IV generated externally, RSA with key length < 2048, ECDSA with key length < 224, ECIES encryption	RSA Encryption (non-compliant) EC Cryptography (non-compliant) EC Encryption with ECIES DES TDES in CBC and ECB modes AES CTR mode (non-compliant) AES GCM mode (non-compliant) AES CBC with 0 IV (non-compliant)	cHSM User
cHSM: Cryptographic functions with flag CXI_MECH_VDM_FLAG or algorithm specifier CXI_KEY_ALGO_CUSTOM	When a cryptographic function is called with the flag CXI_MECH_VDM_FLAG or the algorithm specifier CXI_KEY_ALGO_CUSTOM then the respective functions for the Chinese algorithms are used.	Chinese Algorithms SM2, SM3, SM4 according to IETF drafts	cHSM User

Name	*Description*	*Algorithms*	*Role*
cHSM: Derive Key using non-approved, not allowed key types or mechanisms	cHSM_Derive_Key service with the following key types or mechanisms is allowed only in non-approved mode: symmetric key of size < 112, ECDSA with non-approved, not allowed curves, DSA/DH/DH_PKCS, several non-approved key derivation mechanisms, which are blocked in approved mode	DSA Signature (non-compliant) EC Cryptography (non-compliant) ECDSA with K-/B-curves: K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571 FRP256v1 Curve Montgomery Curves: Curve448, Curve25519 secp256k1 Curve FFC Key Agreement KDF (non-compliant) DES TDES in CBC and ECB modes	cHSM User
cHSM: Generate DSA domain parameters	Generate DSA Param, DSA Param PQ, DSA Param G needed to create a DSA key.	DSA Signature (non-compliant)	cHSM User

Name	*Description*	*Algorithms*	*Role*
cHSM: Generate Key or Key Pair using non-approved, not allowed key types or mechanisms	cHSM_Generate_User_keys service with the following types: symmetric keys of size < 112, RSA with key length < 2048, key generation mechanism other than CXI_MECH_KEYGEN_FIPS_PRIME, ECDSA with non-approved, not allowed curves, DSA/DH/DH_PKCS.	RSA Signature (non-compliant) DSA Signature (non-compliant) EC Cryptography (non-compliant) ECDSA with K-/B-curves: K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571 FRP256v1 Curve Montgomery Curves: Curve448, Curve25519 DES TDES in CBC and ECB modes	cHSM User
cHSM: Import DB Entry	Import an encrypted database entry created by the function Export DB Entry, which is now deprecated.	AES CBC with 0 IV (non-compliant)	cHSM User
cHSM: List Registered Functions	Return a list with the module IDs of the firmware modules with external interfaces.		cHSM User, Unauthenticated
cHSM: Restore User with a backup blob with 0 IV	In non-approved mode, it is possible to restore user data taken from a backup data blob that was output by the Backup User function in previous versions. These backup data were encrypted using AES CBC encryption with 0 IV.	AES CBC with 0 IV (non-compliant)	cHSM User

Name	*Description*	*Algorithms*	*Role*
cHSM: Sign using non-approved, not allowed key types or mechanisms	cHSM_Sign_Data service with the following key types or sizes is allowed only in non-approved mode: DES, AES CBC MAC, GMAC initial IV generated externally, RAW with size < 112, RSA with key size < 2048; hash-algos: SHA-1, MD5, RIPEMD-160; ANSI X9.31, no padding, ECDSA with non-allowed curves (see column algorithms), DSA/DH/DH_PKCS	RSA Signature (non-compliant) DSA Signature (non-compliant) EC Cryptography (non-compliant) ECDSA with K-/B-curves: K-233, K-283, K-409, K-571, B-233, B-283, B-409, B-571 FRP256v1 Curve MD5, MDC-2, RIPEMD-160 (non-compliant) DES TDES MAC TDES ANSI retail MAC AES GMAC with external initial IV (non-compliant) AES CBC MAC (non-compliant)	cHSM User
cHSM: unauthenticated Get Session Key	In non-approved mode, a session key can be obtained (via a function called Get_Session_Key) without authentication.cHSM_Get_Session_Key and cHSM_End_Session services can be performed without authentication.		Unauthenticated

Name	*Description*	*Algorithms*	*Role*
cHSM: Unwrap, Import Key using non-approved, not allowed key types or mechanisms for the unwrapping	The functions Unwrap, Import Key with the following key types or sizes are allowed only in non-approved mode: DES, AES CTR, RSA with key size < 2048	RSA Encryption (non-compliant) DES TDES in CBC and ECB modes AES CTR mode (non-compliant)	cHSM User
cHSM: Verify using non-approved, not allowed key types or mechanisms	cHSM_Verify_Signature service with the following key types or sizes is allowed only in non-approved mode: DES, AES CBC MAC, RSA with key size < 1024; hash-algos: MD5, RIPEMD-160; no padding, ECDSA with non-approved, not allowed curves (see column algorithms), DSA/DH/DH_PKCS	RSA Signature (non-compliant) FRP256v1 Curve MD5, MDC-2, RIPEMD-160 (non-compliant) DES TDES ANSI retail MAC AES CBC MAC (non-compliant)	cHSM User
cHSM: Wrap, Export Key using non-approved, not allowed key types or mechanisms for the wrapping	The functions Wrap, Export Key with the following key types or sizes are allowed only in non-approved mode: DES, AES CTR, CBC 0 IV, GCM initial IV generated externally, RSA with key size < 2048, PKCS1 v1.5	RSA Encryption (non-compliant) DES TDES in CBC and ECB modes AES CTR mode (non-compliant) AES GCM mode (non-compliant) AES CBC with 0 IV (non-compliant)	cHSM User

Table 16: Non-Approved Services

4.5 External Software/Firmware Loaded

Firmware can only be loaded to the module using the service glad_system_update, which allows loading of an operational image if and only if

- the operator successfully authenticates as a Global Administrator (i.e., Crypto Officer),
- and the image being loaded is accompanied by a valid ECDSA signature (i.e., passes ECDSA signature verification algorithm Cert. #A6714) corresponding to the Image Signing Key ISK+.

5 Software/Firmware Security

5.1 Integrity Techniques

The module checks the integrity of the firmware components during pre-operational tests (during power-on) as well as automatically each 24 hours as conditional (periodic) tests. Descriptions of these tests are given in Sections 10.1 and 10.2. The algorithms for integrity testing are CRC32, SHA3-384 (used as an EDC), SHA2-256, SHA2-512, and HMAC-SHA2-256. The HMAC key for HMAC-SHA2-256 computation is a 256-bit fixed value stored in the flash memory.

5.2 Initiate on Demand

All self-tests may be invoked on demand by power-cycling the module.

Additionally, the glad services “glad_system_restart_and_self_tests" and “glad_device_restart_and_self_tests” restart the module and therewith trigger all self-tests including firmware integrity tests. “glad_system_restart_and_self_tests" requires authentication. “glad_device_restart_and_self_tests” does not require authentication.

The cHSM service “cHSM_restart_and_self_tests” triggers cHSM self-tests including firmware integrity tests. It does not require authentication.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Limited

How Requirements are Satisfied:

The module is defined as a multi-chip embedded cryptographic module whose operational environment is classified as limited. The FIPS 140-3 Area 6 Operational Environment requirements are not applicable because the cryptographic module does not contain a modifiable operational environment. The module has the ability to load its firmware from Flash memory. Only the firmware versions identified on the module's validation certificate are included in the scope of this validation. Any other firmware loaded onto the module is not validated and would require a separate FIPS 140-3 validation.

7 Physical Security

7.1 Mechanisms and Actions Required

The module is a multi-chip embedded cryptographic module encapsulated in a hard, opaque, tamper-evident coating. It entails a list of physical security mechanisms, which perform their functions autonomously and under all circumstances.

On the top side of the module a (hollow) metal heat sink is directly mounted on the printed circuit board on three edges, and the space between the PCB and the heat sink is completely filled with potting material (epoxy resin). On the bottom side of the PCB, a metal frame is stuck directly onto the printed circuit board, and the space inside the metal frame is completely filled with potting material. Epoxy hardness testing was performed over the module's storage, distribution and operating temperature range from -10˚C to +60˚C.

The heat sink and potting material together define the top and bottom sides of the module and deliver a hard, opaque coating. All the cryptographic module's hardware components (which are all mounted on the PCB) are entirely covered by this coating.

The module with its tamper-evident enclosure (the heat sink and the potting material and the sensor patch wires) implements the following physical security mechanisms:

- The cryptographic module's hardware components are covered by hard, opaque potting material or the heat sink which show evidence of tampering on the enclosure when a physical attack is attempted.

  The potting material is hard and opaque enough to prevent direct observation and easy penetration to the depth of the underlying hardware components. It is highly probable that anyone attempting to penetrate to the depth of the circuitry will break off large pieces of potting material, disrupt one of the tamper wires and activate the tamper response, or at least tear off important hardware components of the module, causing serious damage to the module.

- Active tamper response and zeroization circuitry.

  Two sensor patch wires (covered by potting material) are observed by the sensory controller. If any of these are disrupted (i.e. in the case of physical attack), the sensory controller activates the tamper response.

  Temperature sensors activate a tamper response if the module is outside of the defined temperature range of –18°C to 81°C (-0.4°F to 177.8°F).

  Voltage sensors monitor the power supply of the module and activate a tamper response if the power input is outside of the defined range (including low or removed battery).

  Tamper response and zeroization circuitry is active while module is in standby mode (powered down).

Zeroization is performed within less than 7 milliseconds after tamper detection (temperature or voltage outside of defined range or disrupted sensor patch wires).

- The module regularly inverts all bits of the plaintext master keys (DMK and SDMK) in the persistent storage referred to as Key RAM to avoid "burn in" of information into SRAM cells.

To ensure security of the cryptographic module, the module must be periodically inspected for evidence of tampering. The recommended inspection schedule depends on the customer's application area. This may vary between inspecting the module once a week and once a year.

Mechanism	*Inspection Frequency*	*Inspection Guidance*
Tamper-evident enclosure	Inspection for evidence of tampering on receipt and periodically depending on the application area and operational environment. This may vary between inspecting the module once a week and once a year	Optical inspection
Tamper response and zeroization circuitry	N/A	No inspection needed: In case of detected tamper, u.trust Anchor erases all cHSMs which immediately stops all cHSM services.

Table 17: Mechanisms and Actions Required

7.2 EFP/EFT Information

As part of the module's general alarm mechanism, measures are taken to protect against environmental failures. Alarm triggering temperature and voltage ranges and protection capabilities are described in Table 18.

Temp/Voltage Type	*Temperature or Voltage*	*EFP or EFT*	*Result*
LowTemperature	< -18C	EFP	Zeroisation
HighTemperature	> 81C	EFP	Zeroisation
LowVoltage	< 2.1V	EFP	Zeroisation
HighVoltage	> 13.9V	EFP	Zeroisation

Table 18: EFP/EFT Information

7.3 Hardness Testing Temperature Ranges

The module has been tested at the operational, storage, and distribution temperatures listed in Table 19. The module's hardness is assured within these ranges.

Temperature Type	*Temperature*
LowTemperature	-10C
HighTemperature	60C

Table 19: Hardness Testing Temperatures

8 Non-Invasive Security

The module does not provide protection against non-invasive security methods.

9 Sensitive Security Parameters Management

9.1 Storage Areas

SSPs are stored in areas defined in Table 20.

Storage Area Name	*Description*	*Persistence Type*
Cache	Part of the DDR4 RAM used as cache for encrypted or non-confidential data and instruction code during runtime.	Dynamic
Flash	Provided by the eMMC. Persistently stores firmware and encrypted or non-confidential data.	Static
Key RAM	Internal RAM located in the sensory controller. Persistently stores unencrypted system keys. Zeroised in case of an alarm.	Static
Secure RAM	Sensory protected part of the DDR4 RAM. Stores unencrypted keys during runtime. Zeroised in case of an alarm.	Dynamic

Table 20: Storage Areas

9.2 SSP Input-Output Methods

Specific SSPs may be entered and output from the module via various methods, which include plaintext or protected by either a secure channel established through Secure Messaging or an Approved KTS. Table 21 lists all SSP input-output methods.

Name	*From*	*To*	*Format Type*	*Distribution Type*	*Entry Type*	*SFI or Algorithm*
CE	External	Cache	Plaintext	Automated	Electronic	KAS
CO	Cache	External	Plaintext	Automated	Electronic	KAS
FO	Flash	External	Plaintext	Automated	Electronic	
FSME	External	Flash	Encrypted	Automated	Electronic	KTS_CBC_CMAC
FSMO	Flash	External	Encrypted	Automated	Electronic	KTS_CBC_CMAC
KTSE	External	Flash	Encrypted	Automated	Electronic	KTS_Encap_RSA
SSME	External	Secure RAM or cache	Encrypted	Automated	Electronic	KTS_CBC_CMAC
SSMO	Secure RAM or cache	External	Encrypted	Automated	Electronic	KTS_CBC_CMAC

Table 21: SSP Input-Output Methods

9.3 SSP Zeroization Methods

Table 22 lists all of the SSP zeroization methods.

Zeroization Method	*Description*	*Rationale*	*Operator Initiation*
Alarm	Physical	Zeroizes SDMK and DMK in Key RAM. Zeroizes Secure RAM and associated caches.	Through tamper detection or by means of environmental failure protection.
Erase	Physical, Service	Zeroizes DMK in Key RAM. Zeroizes Secure RAM and associated caches.	1) By pushing the external erase button when the operator has physical access to the device or 2) by the global admin (Crypto Officer) via glad interface or 3) by pushing the external erase followed by an erasure via glad interface.
Auto	Automatic	After the CSP in Secure RAM is used and no longer needed, it is automatically zeroized.	N/A

Table 22: SSP Zeroization Methods

9.4 SSPs

The SSPs used in the module are listed in Table 23 and Table 24.

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
ALK	cHSM Audit Log Signature Key: Signs audit log	EC P-256, RSA 3072 - 128	Private - CSP	KeyGen		SigGen
ALK+	Public cHSM Audit Log Signature Key: Audit Log Signature Verification, exportable	EC P-256, RSA 3072 - 128	Public - PSP	KeyGen		SigVer

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
AMSK+	Public Alternative Module Signature Key: Verifies integrity and authenticity of a loaded configuration file.	RSA 2048 - 112	Public - PSP	External		SigVer
CAAK+	cHSM Operator Public Authentication Key: cHSM operator authentication	RSA at least 2048, EC 224, 256, 320, 384, 512, 521 - 112-256	Public - PSP	External		SigVer
CAK	Container Authentication Key: Container Authentication for Secure Messaging	EC P-521 - 256	Private - CSP	KeyGen		SigGen
CAK+	Public Container Authentication Key: Container Authentication for Secure Messaging, exportable	EC P-521 - 256	Public - PSP	KeyGen		SigVer
CBK	Container Base Key: Derives Container Master Key	256 - 256	Symmetric - CSP	SKG		KBKDF

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
CBKEK	Container Base Key Encryption Key: Encrypts CBK in snapshot	256 - 256	Symmetric - CSP	KBKDF		KTS_Enc
CIAK+	cHSM operator Initial Admin Key: initial cHSM operator authentication	RSA at least 2048, EC P-256, P-521 - 112-256	Public - PSP	External		SigVer
CMK	Container Master Key: Encrypts secrets within cHSM container	256 - 256	Symmetric - CSP	KBKDF		KTS_Enc
CTIME	Container time: Internal time of the cHSM	144 - N/A	Public - PSP	N/A		
DAK	Device Authentication Key: Generates device individual certificates	EC P-521 - 256	Private - CSP	KeyGen		SigGen
DAK+	Public Device Authentication Key: Certificate verification, exportable	EC P-521 - 256	Public - PSP	KeyGen		SigVer

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
DEK	DMK-Derived Encryption Key: Encrypts all glad secrets apart from vendor secret and DAK	256 - 256	Symmetric - CSP	KBKDF		KTS_Enc
DMK	Device Master Key: Key derivation key for DEK	256 - 256	Symmetric - CSP	SKG		KBKDF
DRBG_Entropy_Input	DRBG Entropy Input to the DRBG	512 - 256	ENT - CSP	Internally by ESV E108		DRBG
DRBG_Seed	Seed for the DRBG, used to initialize the DRBG state	888 - 256	ENT - CSP	Internally by ESV E108		DRBG
DTIME	System time of the device: Internal time of the device (glad)	64 - N/A	Public - PSP	N/A		
GAAK+	Glad Admin Authentication Key: glad operator authentication	RSA at least 2048, EC P-256, P-521, brainpoolP320t - 112-256	Public - PSP	External		SigVer

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
GAK	Glad Authentication Key: Device Authentication for Secure Messaging	EC P-521 - 256	Private - CSP	KeyGen		SigGen
GAK+	Public glad Authentication Key: Device Authentication for Secure Messaging, exportable	EC P-521 - 256	Public - PSP	KeyGen		SigVer
GIAK+	Glad Initial Admin Authentication Key: initial glad operator authentication	EC P-256 - 128	Public - PSP	External		SigVer
ISK+	Image Signing Key: Authenticates new boot images on download	EC P-521 - 256	Public - PSP	External		SigVer
MBK	Master Backup Key: Derives keys for handling cHSM data backups	128, 192, 256 - 128, 192, 256	Symmetric - CSP	SKG		KBKDF

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
MBKDK	MBK-Derived Keys: Keys for handling cHSM data backups	256 - 128, 192, 256	Symmetric - CSP	KBKDF		KTS_CBC_CMAC Unauth_Block_Cipher MAC
OBS	Operator Base Secret: Derives backup and snapshot keys	256 - 256	Symmetric - CSP	External		KBKDF
OBSEK	Operator Base Secret Encryption Key: Encrypts Operator Base Secret on import	RSA 2048-16384 - 112-256	Private - CSP	KeyGen		KTS_Encap_RSA
OBSEK+	Public Operator Base Secret Encryption Key: Encrypts Operator Base Secret on import	RSA 2048-16384 - 112-256	Public - PSP	KeyGen		KTS_Encap_RSA
PSW_AUTH	cHSM User Password for cHSM User authentication	minimum 8 characters - N/A	Authentication - CSP	External		

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
SDEK	SDMK-Derived Encryption Key: Encrypts vendor secret and DAK	256 - 256	Symmetric - CSP	KBKDF		KTS_Enc
SDMK	Sticky Device Master Key: Key derivation key for SDEK	256 - 256	Symmetric - CSP	SKG		KBKDF
SEK	Snapshot Encryption Key: Encrypts cHSM container in snapshot	256 - 256	Symmetric - CSP	KBKDF		KTS_Enc
SMEK	Session Keys: derived from SMKDK per NIST SP 800-108 and used for Secure Messaging	256 - 256	Symmetric - CSP	KBKDF		KTS_CBC_CMAC Unauth_Block_Cipher
SMK	Snapshot MAC Key: Protects Integrity of cHSM snapshot	256 - 256	Symmetric - CSP	KBKDF		MAC

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
SMKDK	Secure Messaging Session Key Derivation Key: derived from Shared Secret via KDA OneStep NIST SP 800-56C and used to derive session keys for Secure Messaging	256 - 256	Symmetric - CSP	KAS		KBKDF
SMLK	Secure Messaging Local Diffie-Hellman Key: generated by the glad as well as each cHSM for Secure Messaging with ECDH	EC P-521 - 256	Private - CSP	KeyGen		KAS

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
SMLK+	Secure Messaging Local Public ECDH Key: generated by the module (glad or cHSM) and used for ECDH for Secure Messaging	EC P-521 - 256	Public - PSP	KeyGen		KAS
SMMK	Session Keys: derived from SMKDK per NIST SP 800-108 and used for Secure Messaging	256 - 256	Symmetric - CSP	KBKDF		KTS_CBC_CMAC MAC
SMRK+	Secure Messaging Remote Public ECDH Key: generated by the host and used for ECDH for Secure Messaging (glad or cHSM)	EC P-521 - 256	Public - PSP	External		PubKeyVal KAS
S_DRBG	DRBG Secrets V and C	888 - 256	ENT - CSP	DRBG		DRBG KeyGen SKG

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
Shared Secret	ECDH key establishment Shared Secret per NIST SP 800-56Ar3	EC 224, 256, 320, 384, 512, 521 - 112-256	Shared Secret - CSP		KAS	KAS
USMK	User Snapshot MAC key: Protects integrity of user backup	256 - 256	Symmetric - CSP	KBKDF		MAC
U_DSA_PUB	Public DSA User Key for signature verification	1024/160, 2048/224, 2048/256 or 3072/256 - 80, 112, 128	Public - PSP	External		SigVer
U_KM	User Key Material	>=112 - >=112	Symmetric - CSP	KAS SKG	KAS	SKG MAC KAS
U_Legacy	HMAC key for legacy used (HMAC verification only)	80-111 - 80-111	Symmetric - CSP	External		MAC
U_PRIV	User Asymmetric Private Key for signature generation, key wrapping, and key agreement	RSA at least 2048 for SigGen and KTS_Encap_RSA, EC 224, 256, 320, 384, 512, 521 for SigGen and KAS, ed25519 and ed448 for SigGen - 112-256	Private - CSP	KeyGen		SigGen KTS_Encap_RSA KAS

Name	*Description*	*Size - Strength*	*Type - Category*	*Generated By*	*Established By*	*Used By*
U_PUB	User Asymmetric Public Key for signature verification, key wrapping, and key agreement	RSA at least 1024 for SigVer, RSA at least 2048 for KTS_Encap_RSA, EC 163-571 for SigVer, KAS and PubKeyVal, ed25519 and ed448 for SigVer - 80-256	Public - PSP	KeyGen		SigVer KTS_Encap_RSA KAS PubKeyVal
U_S	AES User Key: Key Encryption, Data Encryption or MAC	128, 192, 256 - 128, 192, 256	Symmetric - CSP	KAS SKG		Auth_Block_Cipher Unauth_Block_Cipher MAC KTS_Enc KAS
VBS	Vendor Base Secret: Derives snapshot keys	256 - 256	Symmetric - CSP	External		KBKDF

Table 23: SSP Table 1

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
ALK	FSME FSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	ALK+:Paired With CMK:Encrypted By
ALK+	FO FSME FSMO	Flash:Plaintext Cache:Plaintext		N/A	ALK:Paired With
AMSK+	FSME FSMO	Flash:Plaintext Cache:Plaintext		N/A	

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
CAAK+	FSME FSMO	Flash:Plaintext Cache:Plaintext		N/A	
CAK	FSME FSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CAK+:Paired With CMK:Encrypted By
CAK+	FSME FSMO FO	Flash:Plaintext Cache:Plaintext		N/A	CAK:Paired With DAK:Signed By
CBK	FSME FSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DEK:Encrypted By CMK:Derives CBKEK:Encrypted By
CBKEK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CBK:Encrypts VBS:Derived From OBS:Derived From
CIAK+	FSME FSMO	Flash:Plaintext Cache:Plaintext		N/A	
CMK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CBK:Derived From
CTIME	FO FSME FSMO	Flash:Plaintext		N/A	
DAK		Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SDEK:Encrypted By DAK+:Paired With GAK+:Signs CAK+:Signs
DAK+	FSMO FO	Flash:Plaintext Cache:Plaintext		N/A	DAK:Paired With GAK+:Verifies CAK+:Verifies

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
DEK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DMK:Derived From
DMK		Key RAM:Plaintext Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DEK:Derives
DRBG_Entropy_Input		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DRBG_Seed:Derives
DRBG_Seed		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DRBG_Entropy_Input:Derived From S_DRBG:Derives
DTIME	FO FSME FSMO	Flash:Plaintext		N/A	
GAAK+	FSME FSMO	Flash:Plaintext Cache:Plaintext		N/A	
GAK		Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DEK:Encrypted By GAK+:Paired With
GAK+	FSMO FO	Flash:Plaintext Cache:Plaintext		N/A	GAK:Paired With DAK:Signed By
GIAK+	FSME	Flash:Plaintext Cache:Plaintext		N/A	
ISK+	FSME	Flash:Plaintext Cache:Plaintext		N/A	

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
MBK	FSME FSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CMK:Encrypted By MBKDK:Derives
MBKDK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	MBK:Derived From
OBS	FSME KTSE	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DEK:Encrypted By OBSEK+:Encrypted By CBKEK:Derives SEK:Derives SMK:Derives USMK:Derives
OBSEK		Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DEK:Encrypted By OBSEK+:Paired With
OBSEK+		Flash:Plaintext Cache:Plaintext		N/A	OBSEK:Paired With OBS:Encrypts
PSW_AUTH	FSME FSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CMK:Encrypted By
SDEK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SDMK:Derived From
SDMK		Key RAM:Plaintext Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SDEK:Derives
SEK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	VBS:Derived From OBS:Derived From
SMEK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SMKDK:Derived From

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
SMK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CBK:Used With VBS:Derived From OBS:Derived From
SMKDK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SMEK:Derives SMMK:Derives
SMLK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SMLK+:Paired With
SMLK+	CO	Cache:Plaintext		N/A	SMLK:Paired With
SMMK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SMKDK:Derived From
SMRK+	CE	Cache:Plaintext		N/A	
S_DRBG		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	DRBG_Seed:Derived From
Shared Secret		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SMKDK:Derives
USMK		Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	OBS:Derived From
U_DSA_PUB	FSME FSMO SSME SSMO	Flash:Plaintext Cache:Plaintext		N/A	

Name	*Input - Output*	*Storage*	*Storage Duration*	*Zeroization*	*Related SSPs*
U_KM	FSME FSMO SSME SSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CMK:Encrypted By
U_Legacy	FSME FSMO SSME SSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CMK:Encrypted By
U_PRIV	FSME FSMO SSME SSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	U_PUB:Paired With CMK:Encrypted By
U_PUB	FSME FSMO SSME SSMO	Flash:Plaintext Cache:Plaintext		N/A	U_PRIV:Paired With
U_S	FSME FSMO SSME SSMO	Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	CMK:Encrypted By
VBS		Flash:Encrypted Secure RAM:Plaintext	Until no longer needed	Alarm Erase Auto	SDEK:Encrypted By CBKEK:Derives SEK:Derives SMK:Derives

Table 24: SSP Table 2

10 Self-Tests

The following pre-operational and conditional self-tests are performed by the module. All self-tests are performed regardless of whether the module is in the Approved or Non-Approved mode. The data output interface is inhibited whenever the cryptographic module is in a self-test condition.

10.1 Pre-Operational Self-Tests

The pre-operational tests listed in Table 25 are performed upon power-up and on demand, the latter by power-cycling the module. The module logs an appropriate error message and transitions to its error state if the test fails. Each approved algorithm in the pre-operational tests is conditionally self-tested prior to its invocation. SHA3-384 is also used as an EDC for the integrity test of the bootRom and BOOT.bin as a part of the pre-operational tests.

Algorithm or Test	*Test Properties*	*Test Method*	*Test Type*	*Indicator*	*Details*
cHSM firmware integrity test	SHA2-512 (A6715) or CRC32	KAT	SW/FW Integrity	cHSM_Status will show operational mode in case of success. Otherwise, the module will be in Critical Error state.	Verification of cHSM firmware modules
glad firmware integrity test	HMAC-SHA2-256 (A6714) Key length 160 bits or SHA2-256 (A6714)	KAT	SW/FW Integrity	glad_Status will show approved mode in case of success. Otherwise, an error will be logged or the module will be in Critical Error state.	Verification of glad services and SMOS
SHA2-256 (A1563)	N/A	KAT	SW/FW Integrity	glad_Status will show approved mode in case of success. Otherwise, an error will be logged or the module will be in Critical Error State.	Verification of boot image

Table 25: Pre-Operational Self-Tests

10.2 Conditional Self-Tests

The module performs the conditional self-tests listed in Table 26. Cryptographic algorithm self-tests (CAST) are performed on power-up and on-demand. All other conditional tests are triggered when the conditions specified for the tests occur.

Algorithm or Test	*Test Properties*	*Test Method*	*Test Type*	*Indicator*	*Details*	*Conditions*
AES-GCM (A6713, A6714) Encrypt	Key length 256 bits, IV 96 bits, Tag 128 bits	KAT	CAST	Implicit	Encrypt	Prior to first use
AES-GCM (A6713, A6714) Decrypt	Key length 256 bits, IV 96 bits, Tag 128 bits	KAT	CAST	Implicit	Decrypt	Prior to first use
DSA SigVer (FIPS186-4) (A6713)	Key Length 2048 bits	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use
ECDSA SigGen (FIPS 186-5) (A6713, A6714, A6716, A6717)	P-256, B-283, SHA2-256	KAT	CAST	Implicit	Digital Signature Generation KAT	Prior to first use
ECDSA SigVer (FIPS 186-4) (A6713, A6716, A6717)	P-256, B-283, SHA2-256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to fist use
ECDSA SigVer (FIPS 186-5) (A6713, A6714, A6716, A6717)	P-256, B-283, SHA2-256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use
ECDSA SigVer (FIPS186-5) (A6714)	P-521	Signature Verification	SW/FW Load	Implicit	Digital Signature Verification	When loading new firmware
EDDSA SigGen (A6713, A6716)	Ed25519, Ed448, Ed25519ctx, Ed448ctx, SHA2-512, SHAKE256	KAT	CAST	Implicit	Digital Signature Generation KAT	Prior to first use
EDDSA SigVer (A6713, A6716)	Ed25519, Ed448, Ed25519ctx, Ed448ctx, SHA2-512, SHAKE256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use

Algorithm or Test	*Test Properties*	*Test Method*	*Test Type*	*Indicator*	*Details*	*Conditions*
Hash DRBG (A6714, A6715)	Hash DRBG, SHA2-512 based	KAT	CAST	Implicit	NIST SP 800-90A Health Tests for Instantiate, Generate, Reseed	Prior to first use
HMAC-SHA2-256 (A6713, A6714)	SHA2-256	KAT	CAST	Implicit	MAC	Prior to first use
KAS-ECC-SSC SP800-56Ar3 (A6713, A6714, A6716, A6717)	P-256, B-283	KAT	CAST	Implicit	Shared Secret Computation KAT	Prior to first use
KDA OneStep SP800-56Cr2 (A6713, A6714)	SHA2-256, SHA3-256	KAT	CAST	Implicit	Key Derivation	Prior to first use
KDF ANS 9.63 (A6713)	X9.63	KAT	CAST	Implicit	Key Derivation	Prior to first use
KDF TLS (A6713)	TLS 1.2 KDF	KAT	CAST	Implicit	Key Derivation	Prior to first use
KDF SP800-108 (A6713, A6714)	HMAC-SHA256 feedback mode	KAT	CAST	Implicit	Key Derivation	Prior to first use
KTS-IFC (A6713, A6714, A6716, A6717) Encapsulation	Key Length 2048 bits	KAT	CAST	Implicit	Encapsulation KAT	Prior to first use
KTS-IFC (A6713, A6714, A6716, A6717) Decapsulation	Key Length 2048 bits	KAT	CAST	Implicit	Decapsulation KAT	Prior to first use

Algorithm or Test	*Test Properties*	*Test Method*	*Test Type*	*Indicator*	*Details*	*Conditions*
Public Key Validation	P-224, P-256, P-384, P-521, ED-25519, ED-448	N/A	Critical Function	Implicit	KAS-ECC Peer Public Key, ECDSA Public Key, EDDSA Public Key	Prior to first use
PWCT (A6713, A6714, A6716, A6717)	Sign/Verify	Sign/Verify	PCT	Implicit	ECDSA (FIPS 186-5), RSA (FIPS 186-5)	Upon key generation
PWCT (A6713, A6716)	Sign/Verify	Sign/Verify	PCT	Implicit	EDDSA	Upon key generation
RSA SigGen (FIPS 186-5) (A6713, A6716, A6717)	2048-bit PSS with SHA2-256	KAT	CAST	Implicit	Digital Signature Generation KAT	Prior to first use
RSA SigVer (FIPS 186-2) (A6713, A6716, A6717)	2048-bit PSS with SHA2-256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use
RSA SigVer (FIPS 186-4) (A6713, A6716, A6717)	2048-bit PSS with SHA2-256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use
RSA SigVer (FIPS 186-5) (A6713, A6714, A6716, A6717)	2048-bit PSS with SHA2-256	KAT	CAST	Implicit	Digital Signature Verification KAT	Prior to first use
SHA2-256 (A1563)	SHA2-256	KAT	CAST	Implicit	Hash KAT	Prior to first use
SHA2-512 (A6715)	SHA2-512	KAT	CAST	Implicit	Hash KAT	Prior to first use
SHA3-224 (A6713)	SHA3-224	KAT	CAST	Implicit	Hash KAT	Prior to first use

Algorithm or Test	*Test Properties*	*Test Method*	*Test Type*	*Indicator*	*Details*	*Conditions*
SHS (A6713, A6714)	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	KAT	CAST	Implicit	Hash KAT	Prior to first use
SP800-90B	APT, RCT	NIST SP 800-90B Health Tests	CAST	Implicit	APT, RCT	Upon use

Table 26: Conditional Self-Tests

10.3 Periodic Self-Test Information

The module performs the glad firmware integrity test and the cHSM firmware integrity test automatically without external input or control every 24 hours. SHA2-256 (Cert. #A1563) pre-operational test is exempt from the periodic test requirements per IG 10.3.E Resolution 3.c.

Algorithm or Test	*Test Method*	*Test Type*	*Period*	*Periodic Method*
cHSM firmware integrity test	KAT	SW/FW Integrity	Every 24 hours	Automatic
glad firmware integrity test	KAT	SW/FW Integrity	Every 24 hours	Automatic
SHA2-256 (A1563)	KAT	SW/FW Integrity	N/A	N/A

Table 27: Pre-Operational Periodic Information

Algorithm or Test	*Test Method*	*Test Type*	*Period*	*Periodic Method*
AES-GCM (A6713, A6714) Encrypt	KAT	CAST	N/A	N/A
AES-GCM (A6713, A6714) Decrypt	KAT	CAST	N/A	N/A
DSA SigVer (FIPS186-4) (A6713)	KAT	CAST	N/A	N/A
ECDSA SigGen (FIPS 186-5) (A6713, A6714, A6716, A6717)	KAT	CAST	N/A	N/A

Algorithm or Test	*Test Method*	*Test Type*	*Period*	*Periodic Method*
ECDSA SigVer (FIPS 186-4) (A6713, A6716, A6717)	KAT	CAST	N/A	N/A
ECDSA SigVer (FIPS 186-5) (A6713, A6714, A6716, A6717)	KAT	CAST	N/A	N/A
ECDSA SigVer (FIPS186-5) (A6714)	Signature Verification	SW/FW Load	N/A	N/A
EDDSA SigGen (A6713, A6716)	KAT	CAST	N/A	N/A
EDDSA SigVer (A6713, A6716)	KAT	CAST	N/A	N/A
Hash DRBG (A6714, A6715)	KAT	CAST	N/A	N/A
HMAC-SHA2-256 (A6713, A6714)	KAT	CAST	N/A	N/A
KAS-ECC-SSC SP800-56Ar3 (A6713, A6714, A6716, A6717)	KAT	CAST	N/A	N/A
KDA OneStep SP800-56Cr2 (A6713, A6714)	KAT	CAST	N/A	N/A
KDF ANS 9.63 (A6713)	KAT	CAST	N/A	N/A
KDF TLS (A6713)	KAT	CAST	N/A	N/A
KDF SP800-108 (A6713, A6714)	KAT	CAST	N/A	N/A
KTS-IFC (A6713, A6714, A6716, A6717) Encapsulation	KAT	CAST	N/A	N/A
KTS-IFC (A6713, A6714, A6716, A6717) Decapsulation	KAT	CAST	N/A	N/A

Algorithm or Test	*Test Method*	*Test Type*	*Period*	*Periodic Method*
Public Key Validation	N/A	Critical Function	N/A	N/A
PWCT (A6713, A6714, A6716, A6717)	Sign/Verify	PCT	N/A	N/A
PWCT (A6713, A6716)	Sign/Verify	PCT	N/A	N/A
RSA SigGen (FIPS 186-5) (A6713, A6716, A6717)	KAT	CAST	N/A	N/A
RSA SigVer (FIPS 186-2) (A6713, A6716, A6717)	KAT	CAST	N/A	N/A
RSA SigVer (FIPS 186-4) (A6713, A6716, A6717)	KAT	CAST	N/A	N/A
RSA SigVer (FIPS 186-5) (A6713, A6714, A6716, A6717)	KAT	CAST	N/A	N/A
SHA2-256 (A1563)	KAT	CAST	N/A	N/A
SHA2-512 (A6715)	KAT	CAST	N/A	N/A
SHA3-224 (A6713)	KAT	CAST	N/A	N/A
SHS (A6713, A6714)	KAT	CAST	N/A	N/A
SP800-90B	NIST SP 800-90B Health Tests	CAST	N/A	N/A

Table 28: Conditional Periodic Information

10.4 Error States

The module transitions into the error states when an error condition is encountered and provides an error status indicator as specified in Table 29. All data output and cryptographic functions are inhibited while the module is in an error state.

Name	*Description*	*Conditions*	*Recovery Method*	*Indicator*
Critical Error	The module is unable to perform any action or services or cHSM does not accept security-relevant commands or any command.	Critical error during pre-operational self-tests or error during cHSM initialization or self-test phase or integrity FW test failure.	Reset or power-cycle or restart the cHSM	Implicit. No communication possible with the device or cHSMs or cHSM_Status service answers with "error state = ON" and mode = 'Operational'
Hard Error	Alarm event or external erase or boot image update failure.	Physical alarm or external erase event triggered or failure in the secondary boot image	Invoke glad_System_Reset_Alarm or glad_system_update with succeeding glad_system_restart_and_self_tests	Implicit. glad_Status returns "Alarm" or contains "recovery"
Transient Error	Entropy source or DRBG pre-operational or conditional tests failure.	Entropy source, glad DRBG, or cHSM DRBG pre-operational or conditional tests failure.	Automatically	Implicit. TRNG returns an error or any glad or cHSM service requesting new random bits return an error.

Table 29: Error States

10.5 Operator Initiation of Self-Tests

At any time, the Global Administrator operator can force the module to perform all power-up self-tests via the following services: glad_device_restart_and_self_tests and glad_system_restart_and_self_tests. Each cHSM operator can force their cHSM to perform all cHSM power-up self-tests via the cHSM_restart_and_self_tests service.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The administration manual delivered with the product bundle describes in details procedures for installing, initializing, and starting the module.

Configuration of Approved Mode:

Global Administrator operators use the gladm tool to access the Global Administrator Application glad. cHSM administrators use the csadm tool to access their cHSM.

The operators must update the default authentication keys when they authenticate to the module for the first time. The Global Initial Admin Key is delivered as part of the u.trust Anchor product bundle that can be downloaded from the Utimaco download portal.

The device is *in FIPS 140-3 validated state* if a valid FIPS certificate exists for the device identifier versions output by the glad_Status service:

```
$ gladm Dev=<device_address> system-info
```

A Global Administrator operator can perform glad_chsm_create to create a cHSM. To create a cHSM in the Approved mode, the operator uses the FIPS template 'SecurityServer-FIPS'. To create a cHSM in the Non-Approved mode, the operator uses the non-FIPS template 'SecurityServer'.

The cHSM mode cannot be changed after creation.

Each cHSM's mode of operation is indicated by the cHSM_Status service:

```
$ csadm Dev=<device address, cHSM slot number> GetState
```

A cHSM operates *in the Approved mode* if the device is in FIPS 140-3 validated state and if the cHSM_Status service outputs 'FIPS mode = ON'.

11.2 Administrator Guidance

The administrator Guidance delivered with the product bundle provides information and guidance for Crypto Officers administrating the u.trust Anchor device and the cHSMs.

11.3 Non-Administrator Guidance

cHSM User guidance delivered with the product bundle describes the usage of cryptographic services within the cHSMs.

11.4 Design and Rules

The following security rules are enforced by the cryptographic module to ensure the FIPS 140-3 security requirements are met.

1. The module supports an Approved and a Non-Approved mode of operation.

2. The module does not allow unauthenticated operators to have any access to the module's cryptographic services.
3. Concurrent operators are separated by the unique sessions. No services from a given session can modify the services/data of another separate session.
4. The module inhibits data output during self-tests, firmware load, zeroization, and error states.
5. The module logically disconnects data output from the processes performing zeroization and key generation.
6. The module enforces identity-based authentication.
7. The module does not retain the authentication of an operator following power-off or reboot.
8. The module supports the following roles: Cryptographic Officer and User.
9. The module does not support a bypass mode or maintenance mode.
10. The module supports the following logically distinct interfaces:
    a) Data input interface
    b) Data output interface
    c) Control input interface
    d) Status output interface
    e) Power interface
11. The module protects critical security parameters from unauthorized disclosure, modification, and substitution.
12. The zeroization excludes the overwriting of an unprotected SSP with another unprotected SSP.
13. The module performs power-on, on-demand, and periodic self-testing.
14. The module logs errors whenever an error state is entered.
15. The module does not perform any cryptographic functions while in an error state.

11.5 Maintenance Requirements

The battery status shall be checked regularly using the service glad_Config and be replaced if the status is critical. Detailed instructions can be found in operation manuals delivered with the product bundle: in the PCIe manual under section "Replacing the Battery" and in the LAN manual under section "Maintenance."

11.6 End of Life

Before disposing of the module, the Crypto Officer shall perform an external erase by pushing the erase button. An alarm will be triggered that is shown by the LED flashing up to red, causing the zeroization of sensitive data. The battery shall, in addition, be

removed and disposed of according to the recycling rules and laws of the country where the module is deployed. Apart from pushing the external erase, the removal of the battery also causes the automatic deletion of sensitive data after a maximum of 30 minutes.

12 Mitigation of Other Attacks

The cryptographic module has been designed to mitigate several physical attacks, Simple and Differential Power Analysis (SPA/DPA) and timing analysis listed in Section 12.1.

12.1 Attack List

Mitigation of other attacks is described below.

Other Attacks	*Mitigation Mechanism*
Timing Analysis	Triple-DES and AES operations are executed in constant time so that it is not feasible to determine the value of an algorithm's keys by measuring the execution time of a cryptographic operation. If blinding is switched on for RSA and ECDSA on a cHSM, the input data for a single RSA and ECDSA signature generation is randomized through the use of a blinding technique so that the input parameters of the algorithm are not known by the operator. In this case, it is hard to gain knowledge about the private key based on the amount of time required by the signature generation operation. Blinding is not possible for bulk signing.
Mechanical attack	Each u.trust Anchor with hardware version 7.03.00.03 has two sensor patch wires installed in the epoxy. If one of these is disturbed (i.e., in the case of physical attack), the tamper response is activated resulting in zeroization.
Temperature and voltage	The module provides physical EFP temperature and voltage protections that are outside the scope of FIPS 140-3 physical security Level 3. A tamper response (that is, zeroization) is activated if the module is outside the defined temperature range (–18°C to 81°C) or voltage range (2.1V to 13.9V).

Table 30: Mitigation of Other Attacks