Cogito Group

DIGITAL IDENTITY AND SECURITY

Jellyfish CA

Security Target

24 April 2026

Version 11.0

Owner:	Cogito Group Pty Ltd
Contact details:	Telephone: 1800 COGITO (264486) Email: security.services@cogitogroup.net
Document status:	FINAL

Identifier

C-J-CA-ST-10.0

Revision history

Revision date	Version No.	Author	Description of changes
08/09/2023	1.0		Initial Release
14/06/2024	1.1	Viden Consulting	Updates to remediate findings in EOR1 & EOR2.
05/07/2024	1.2	Viden Consulting	Updates to remediate findings in EOR2.
09/07/2024	1.3	Viden Consulting	Updates to remediate findings in EOR2.
08/12/2024	2.0	Viden Consulting	Updated for Jellyfish v 7.0.17.4.
04/01/2024	2.1	Viden Consulting	Formatting Changes.
24/04/2025	3.0	Viden Consulting	Address FPR_TUD_EXT.1 X509 script Justify non-applicable SFRs.
19/05/2025	4.0	Viden Consulting	Adjustments for non-applicable SFRs.
25/06/2025	5.0	Viden Consulting	Adjustments for non-applicable SFRs.
15/09/2025	6.0	Viden Consulting	Minor formatting corrections.
4/11/2025	6.1	Viden Consulting	Removed OE Security Objective & feedback incorporated.
18/11/2025	7.0	Viden Consulting	Republished inclusive of feedback & updated version numbers.
22/01/2026	8.0	Viden Consulting	Update to include a NIAP Technical Decision and clarification on webserver protocol.
17/02/2026	9.0	Viden Consulting	Clarification of Package Claims.
20/04/2026	10.0	Viden Consulting	Removal of Leviathan.
24/04/2026	11.0	Viden Consulting	Build Guide version change.

Figures

Tables

1 Introduction

1.1 Description of Jellyfish CA

Jellyfish CA is an enterprise certificate authority built on a collection of microservices allowing the issuance and life cycle management of public key certificates of the type specified in the X.509 v3 standard (RFC 5280).

Jellyfish CA is part of the wider Jellyfish security suite offered by Cogito, allowing full visibility and a single pane of glass for creation of symbiotic security ecosystems.

Jellyfish CA functionalities can be accessed through web interfaces (by end users or TOE users) or APIs (by applications that integrate it). More information can be found at Cogito's website.

The rest of this document describes the Jellyfish CA Target of Evaluation (TOE) which forms the scope of a Common Criteria (CC) evaluation and the corresponding Security Target (ST).

1.2 TOE Reference

Table 1: TOE References

ST Title	Jellyfish CA Security Target
ST Reference	C-J-CA-ST-11.0
TOE Identification	Cogito Jellyfish Certificate Authority 7.0
Common Criteria Conformance	Common Criteria Version 3.1, Revision 5 (CC)
(Protection Profile) PP Conformance	NIAP Protection Profile for Certification Authorities, v2.1, 2017-12-01 [PP_CA_v2.1]

1.3 TOE Overview

The TOE is a Certificate Authority (CA). As an enterprise class Public Key Infrastructure (PKI) CA, Jellyfish CA issues and manages the life cycle of public key certificates through application of software, hardware, processes and protocols. The CA supports creation of X.509 v3 standard (RFC 5280) certificates and utilises NIST-compliant encryption algorithms. This allows the issuance of public key certificates for different purposes, including:

- System management functions (e.g., security audit, configuration management, archive)
- Key generation/storage in the Operational Environment
- Certificate generation, modification, re-key, renewal, and distribution
- Certificate revocation list (CRL) generation and distribution
- Key escrow and recovery
- Directory management of certificate related items
- Certificate token initialization/programming/management

Public Key Cryptography relies on digital certificates in order to authenticate users. Given the complex nature of the issuance and management of the digital certificate lifecycle, organisations that want to carry out these

types of operations usually require Certificate Authority applications to ensure effective registration, revocation and renewal of certificates.

The Cogito Jellyfish CA is built on mature technologies and is platform independent in the delivery of PKI services for the end user. Further, integration of Application Programming Interface (API) technologies allows other applications, including the Jellyfish Security Suite, to interface with a secure key infrastructure provider.

1.3.1 Non-TOE Hardware/Software/Firmware

Additional minimum hardware and software components are required in order to operate the system. These components are considered to be excluded from the TOE boundary:

- A server/virtual machine already provisioned running Ubuntu Linux (22.04) with the minimum recommended specification for the server/virtual machine as follows:
    - Cores: 2
    - RAM: 4GB
    - HDD: 30GB
- The following source files need to be copied to the Virtual Machine:
    - Consul Hashicorp Community 1.21
    - Node.js 22.17
    - PostgreSQL 15.13
    - SoftHSM 2.6

1.4 TOE Description

1.4.1 TOE Physical Scope

As illustrated by Figure 1 below, the TOE includes the following:

- Jellyfish CA codebase
- Consul API
- Postgresql database

Excluded from the TOE is the:

- system hardware
- operating system
- Hardware Security Module (HSM).

All items are delivered in an archived file from a Cogito Jellyfish web server.

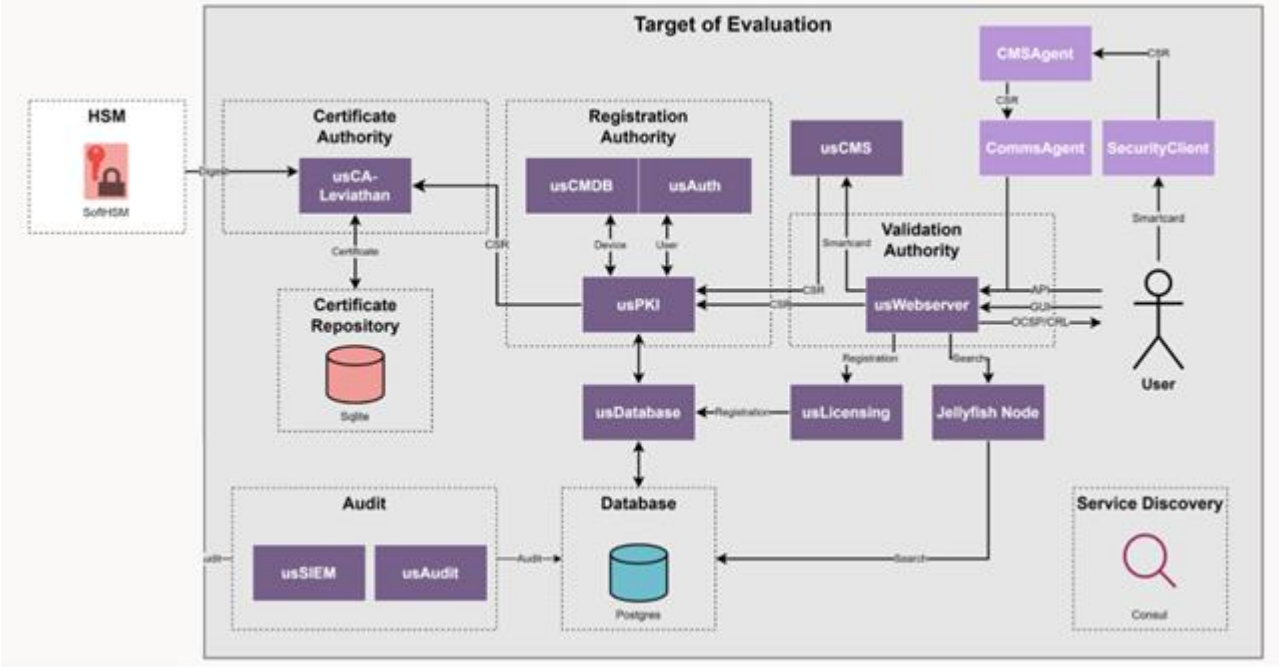


Figure 1: TOE Boundary

1.4.2 TOE Components

1.4.2.1 Jellyfish CA Application

The Jellyfish CA Application is a series of services that allow Graphical User Interface (GUI) and API access for:

- generation and parsing of certificate requests
- creation of digital certificates and private keys
- search functionality
- generation and viewing of audit events
- interfacing with the Operational Environment HSM to provide key generation, storage and management services
- Online Certificate Status Protocol (OCSP) and Certificate Revocation List (CRL) services for certificate revocation
- user authentication and authorisation.

1.4.2.2 React FrontEnd

The React FrontEnd provides a GUI for the performance of administrative functions on the application.

1.4.2.3 Webserver

The webserver provides implementation of Hypertext Transfer Protocol over Transport Layer Security (HTTPS) for the provision of API and GUI interactions for the system user. Further, the webserver utilises standard Go libraries to invoke TLS and HTTPS functionality for the system.

1.4.2.4 Consul Hashicorp Community

Consul is a middleware service mesh that allows for the discovery, secure communication through Mutual Transport Layer Security (mTLS) and service access for microservices running on Jellyfish CA.

1.4.2.5 Database

The database on Jellyfish CA is used to:

- store audit events associated with the system
- store system objects including certificate requests, certificates, certificate metadata and CRLs
- store user credentials and roles
- store profiles used for certificate generation
- store configuration as relating to CA functionality.

1.4.3 TOE Environment

The Operational Environment in which Jellyfish CA operates in, is described below.

1.4.3.1 Operating System

Jellyfish CA, as a microservices operated platform, operates independent of hardware and operating system. Therefore, Jellyfish CA is expected to maintain equivalent security functionality regardless of the employed operating system. Ubuntu Linux has been selected for the TOE, but the underlying technologies are vendor neutral and if the Operational Environment objectives are met, offer the same level of security.

1.4.3.2 Hardware Security Module

To ensure the efficacy of cryptographic operations and allow integration into existing environments, Jellyfish CA interfaces with HSMs to provide cryptographic generation and management services that support the generation and storage of keys.

1.4.3.3 Evaluated Configuration

In order to ensure the consistency of the TOE when installed by end users, the following configuration has been used in the evaluation.

Table 2: Evaluated Configuration

Part	Description/Identifier
Jellyfish CA	7.0
Jellyfish CogVA Build Guide	Build_Guide-VA_CogVA_v1.5.pdf
Operating System	Ubuntu 22.04”.pdf
Hardware Security Module	SoftHSM 2
Relational Database	Postgresql 15
Functional Specification	EFV-T002 Functional Specification v1.2.xlsx
Build Procedures	Build_Guide-Jellyfish_Linux_All-In-One_Common Criteria_v1.51.2.pdf

Deployment Guide	Jellyfish_Deployment_v7.0.1.0-v7.0.17.4.docx
Administrative Procedures	Jellyfish Client Admin Guide_v7.6.121.pdf
Configuration List	Jellyfish_Configuration_List_v1.18.pdf

1.4.4 TOE Logical Scope

The TOE performs the following logical functions.

1.4.4.1 Generation and parsing of certificate requests

Certificate requests are a fundamental service which ensure the consistent and authenticated creation of certificates by a PKI. Jellyfish CA ensures that certificate requests are generated and parsed in conformance with RFC 2511.

1.4.4.2 Creation of digital certificates and private keys

Jellyfish CA ensures that certificates are created in accordance with RFC 5280. Additionally, Jellyfish CA invokes the creation of certificate private keys by HSMs in the Operational Environment. Proper creation of certificates is important to ensure the:

- integrity of the certificates is preserved
- cryptographic algorithms are robust and entropy sources are used in generating private keys
- certificates are signed by a chain of intermediate CAs leading to a trusted root CA.

1.4.4.3 Search functionality

Implementation of a broad search functionality is important to allow the effective management of all CA functions provided by the system.

Search on Jellyfish CA supports security functionality by allowing the ability to:

- find certificates by subject name or other parameters
- search, refine and report on audit events collected by the system
- find and manage system users.

1.4.4.4 Generation and viewing of audit events

Effective auditing allows the early detection and mitigation of adverse security events on the system. Jellyfish CA integrates audit storage and search within the TOE to ensure the requirements of the PP.

1.4.4.5 Interfacing with Operating Environment HSM to provide key generation, storage and management services

Robust generation, storage and management of cryptographic material is required to ensure the confidentiality and integrity of the PKI infrastructure and operations. Through interfaces to the Operational Environment, Jellyfish CA provides users with the ability to utilise existing HSMs, allowing for a return on investment and modularity as new HSMs become available for the system.

The following services are provided:

- key generation
- key recovery

- token management
- electronic signatures.

1.4.4.6 OCSP and CRL services for certificate revocation

Implementation of a CRL is critical to ensure that the revocation of certificates can be performed in such a way that ensures the integrity of the overarching infrastructure. CRLs, by their nature, are static and prone to relying on end-users to regularly interrogate the service to maintain currency of the CRL. OCSP is a protocol that allows the real-time query of revoked certificates by applications to ensure the authenticity of certificates created by the CA. Jellyfish CA implements CRL and OCSP in line with RFC 5280 and RFC 5019 respectively.

1.4.4.7 User authentication and authorisation

Aggregation of privileges for a certificate authority can lead to security issues including unauthorised certificate generation, revocation or deletion. To ensure effective separation of duties in certificate authority operations, effective user authentication and authorisation is required. Jellyfish CA implements user authentication and authorisation for users in the GUI and the API.

1.4.5 TOE Roles

Administrative roles are fully configurable, but a defined set of roles via JSON files is provided with the TOE. The following roles are defined for the CC configuration of Jellyfish CA and correspond to the roles defined in the Protection Profile:

Table 3: TOE Roles for Jellyfish CA configuration

Protection Profile Role	Common Criteria Configuration Role
Administrator	Administrator.json
Auditor	Auditor.json
CA Operations Staff	CA Operator.json
RA Staff	RA Operator.json
Registration Officer	Registration Officer.json

2 Conformance Claims

CC Conformance Claims

This ST claims conformance to Parts 2 (extended) and 3 (conformant) of Common Criteria Version 3.1, Release 5.

2.1 PP Claim

This ST claims **exact conformance** to the NIAP-CCEVS Certification Authorities (PP_CA_v2.1). Spelling in this document has been changed to Australian English as a grammatical change where appropriate, but this does not represent a material change to the conformance claim. Further, refinement of SFRs has occurred to meet stricter selection requirements as articulated in NSA CsfC Selections for Certificate Authorities.

2.2 Applicable Technical Decisions

The following NIAP Technical Decisions are applicable to the PP_CA_V2.1 and have been considered in development of this Security Target:

- 0932 – Clarification when CTR_DRBG is Selected for FCS_RBG_EXT.1.2 in PP_CA_V2.1
- 0946 – Adding FIPS 186-5 in PP_CA_V2.1
- 0866 – Removal of Obsolete Parts of TLSS.1.1 Test 4
- 0845 – Addition of File-Based Protocols to FIA_CMCS_EXT.1.3
- 0796 – Removal of SHA-1 from Various Selections
- 0782 – Terminology Change in CAPP: Extended to Functional Package
- 0599 – Corrections to SAR Section in CAPP
- 0522 – Updates to Certificate Revocation (FIA_X509_EXT.1)
- 0500 – Cryptographic selections and updates for CAPP
- 0415 – Trusted Update Test 4 Conditional
- 0375 – FMT_MOF.1(4) selection
- 0353 – Guidance for Certificate Profiles
- 0348 – FCS_TLSS_EXT.2.4 for TLS 1.2 or higher
- 0328 – Split Knowledge Procedures distinction
- 0294 – Correction of TLS SFRs in CA PP v2.1
- 0287 – FAU_STG.4 Testing
- 0286 – Audit Events for FPT_RCV.1
- 0278 – Clarification of Role for Managing Manual Certificate Requests
- 0276 – X.509 Code Signing on TOE Updates

Conformance Rationale

All assumptions, threats, policies, objectives and security requirements defined for PPs for CAs have been reproduced in this ST. No additional assumptions, threats, policies, objectives or security requirements have been used.

Table 4 lists all Security Functional Requirements (SFRs) that have or have not been included. The following key is used:

- M- Mandatory
- O – Optional
- S – Selectable.

Table 4: Conformance Rationale

Security Functional Requirement	M	O	S	Incl.	Rationale
FAU_ADP_EXT.1 Audit Dependencies	X			Y	Mandatory
FAU_GCR_EXT.1 Generation of Certificate Repository	X			Y	Mandatory
FAU_GEN.1 Audit Data Generation	X			Y	Mandatory
FAU_GEN.2 User Identity Association	X			Y	Mandatory
FAU_SAR.1 Audit Review			X	Y	
FAU_SAR.3 Selectable Audit Review			X	Y	
FAU_SCR_EXT.1 Certificate Repository Review			X	Y	
FAU_SEL.1 Selective Audit			X	N	All audit logs are stored on the TOE
FAU_STG.1(1) Protected Audit Trail Storage			X	Y	
FAU_STG.1(2) Protected Audit Trail Storage (Archive Data)			X	Y	
FAU_STG.4 Prevention of Audit Data Loss	X			Y	Mandatory
FAU_STG_EXT.1 External Audit Trail Storage			X	Y	
FAU_STG_EXT.2 Audit Data Retention			X	Y	
FCO_NRO_EXT.2 Certificate-Based Proof of Origin	X			Y	Mandatory
FCO_NRR_EXT.2 Certificate-Based Proof of Receipt			X	Y	
FCS_CDP_EXT.1 Cryptographic Dependencies	X			Y	Mandatory
FCS_CKM.1 Cryptographic Key Generation			X	Y	

Security Functional Requirement	M	O	S	Incl.	Rationale
FCS_CKM.2 Cryptographic Key Establishment			X	Y	
FCS_CKM_EXT.1(1) Symmetric Key Generation for DEKs			X	N	System does not produce DEKs
FCS_CKM_EXT.1(2) Key Generation Key Encryption Keys			X	Y	
FCS_CKM_EXT.1(3) Key Generation for Key Encryption Keys (TOE Key Archival)			X	Y	
FCS_CKM_EXT.1(4) Generation of Key Shares			X	Y	
FCS_CKM_EXT.4 Cryptographic Key Destruction			X	Y	
FCS_CKM_EXT.5 Public Key Integrity			X	Y	
FCS_CKM_EXT.6 TOE Key Archival			X	Y	
FCS_CKM_EXT.7 Key Generation for KEKs			X	Y	
FCS_CKM_EXT.8 Key Hierarchy Entropy			X	Y	
FCS_COP.1(1) Cryptographic Operation (AES Encryption/Decryption)			X	Y	
FCS_COP.1(2) Cryptographic Operation (Cryptographic Signature)			X	Y	
FCS_COP.1(3) Cryptographic Operation (Cryptographic Hashing)			X	Y	
FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication)			X	Y	
FCS_COP.1(5) Cryptographic Operation (Password-Based Key Derivation Function)		X		N	System does not create password-based keys
FCS_HTTPS_EXT.1 HTTPS Protocol			X	Y	
FCS_IPSEC_EXT.1 IPSec Protocol			X	N	Jellyfish does not support IPSec
FCS_KSH_EXT.1 Key Sharing		X		N	Keys are not exportable for sharing from TOE
FCS_RBG_EXT.1 Cryptographic Random Bit Generation			X	Y	
FCS_STG_EXT.1 Cryptographic Key Storage	X			Y	Mandatory
FCS_TLSC_EXT.2 TLS Client Protocol			X	N	Client not part of the TOE

Security Functional Requirement	M	O	S	Incl.	Rationale
FCS_TLSS_EXT.1 TLS Server Protocol			X	Y	
FDP_CER_EXT.1 Certificate Profiles	X			Y	Mandatory
FDP_CER_EXT.2 Certificate Request Matching	X			Y	Mandatory
FDP_CER_EXT.3 Certificate Issuance Approval	X			Y	Mandatory
FDP_CER_EXT.4 Non-X.509v3 Certificate Generation			X	N	Jellyfish only uses x509v3 Certificates
FDP_CRL_EXT.1 Certificate Revocation List Validation			X	Y	
FDP_CSI_EXT.1 Certificate Status Information	X			Y	Mandatory
FDP_ITT.1 Basic Internal Transfer Protection			X	Y	
FDP_OCSPG_EXT.1 OCSP Basic Response Generation	X			Y	Mandatory
FDP_RIP.1 Subset Residual Information Protection	X			Y	Mandatory
FDP_SDP_EXT.1 User Sensitive Data Protection		X		N	Jellyfish does not store sensitive user data
FDP_STG_EXT.1 Public Key Protection		X		Y	
FIA_AFL.1 Authentication Failure Handling			X	Y	
FIA_CMCC_EXT.1 Certificate Management over CMS (CMC) Client			X	N	Jellyfish does not contain a CMC client
FIA_CMCS_EXT.1 Certificate Management over CMS (CMC) Server			X	Y	
FIA_ENR_EXT.1 Certificate enrolment		X		N	No external interfaces available for CA certificate generation
FIA_ESTC_EXT.1 Enrolment over Secure Transport (EST) Client			X	N	Jellyfish does not utilise EST as a protocol
FIA_ESTC_EXT.2 EST Client use of TLS-unique value		X		N	Jellyfish does not utilise EST as a protocol
FIA_ESTS_EXT.1 Enrolment over Secure Transport (EST) Server			X	N	Jellyfish does not utilise EST as a protocol
FIA_ESTS_EXT.2 Enrolment over Secure Transport (EST) Server		X		N	Jellyfish does not utilise EST as a protocol

Security Functional Requirement	M	O	S	Incl.	Rationale
FIA_PMG_EXT.1 Password Management			X	Y	
FIA_PSK_EXT.1 Pre-Shared Key Composition			X	N	
FIA_UAU.7 Protected Authentication Feedback			X	Y	
FIA_UAU_EXT.1 Authentication Mechanism	X			Y	Mandatory
FIA_UIA_EXT.1 User Identification and Authentication	X			Y	Mandatory
FIA_X509_EXT.1 Certificate Validation	X			Y	Mandatory
FIA_X509_EXT.2 Certificate-Based Authentication	X			Y	Mandatory
FIA_X509_EXT.3 X509 Certificate Request		X		N	External Certificate enrolment with PKCS#10 not included in scope
FMT_MOF.1(1) Management of Security Functions Behaviour (Administrator Functions)	X			Y	Mandatory
FMT_MOF.1(2) Management of Security Functions Behaviour (CA/RA Functions)	X			Y	Mandatory
FMT_MOF.1(3) Management of Security Functions Behaviour (CA Operations Functions)	X			Y	Mandatory
FMT_MOF.1(4) Management of Security Functions Behaviour (Admin/Officer Functions)	X			Y	Mandatory
FMT_MOF.1(5) Management of Security Functions Behaviour (Auditor Functions)	X			Y	Mandatory
FMT_MTD.1 Management of TSF Data	X			Y	Mandatory
FMT_SMF.1 Specification of Management Functions	X			Y	Mandatory
FMT_SMR.2 Restrictions on Security Roles	X			Y	Mandatory
FPT_APW_EXT.1 Protection of Privileged User Passwords			X	Y	
FPT_FLS.1 Failure with Preservation of Secure State	X			Y	Mandatory
FPT_ITT.1 Basic Internal TSF Data Transfer Protection			X	Y	

Security Functional Requirement	M	O	S	Incl.	Rationale
FPT_KST_EXT.1 No Plaintext Key Export	X			Y	Mandatory
FPT_KST_EXT.2 TSF Key Protection	X			Y	Mandatory
FPT_NPE_EXT.1 NPE Constraints		X		Y	
FPT_RCV.1 Manual Trusted Recovery	X			Y	Mandatory
FPT_SKP_EXT.1 Protection of Keys	X			Y	Mandatory
FPT_SKY_EXT.1 Split Knowledge Procedures		X		N	Split knowledge not implemented
FPT_SKY_EXT.2 Key Share Access			X	N	Key share access not implemented
FPT_STM.1 Reliable Time Stamps	X			Y	Mandatory
FPT_TST_EXT.1 TOE Integrity Test		X		N	The TOE does not implement integrity testing.
FPT_TST_EXT.2 Integrity Test		X		N	The TOE does not implement integrity testing.
FPT_TUD_EXT.1 Trusted Update	X			Y	Mandatory
FTA_SSL.3 TSF-Initiated Termination		X		Y	
FTA_SSL.4 User-Initiated Termination	X			Y	Mandatory
FTA_SSL_EXT.1 TSF-Initiated Session Locking		X		N	Connecting to local session is not possible as Jellyfish server is on VM
FTA_TAB.1 Default TOE Access Banners	X			Y	Mandatory
FTP_ITC.1 Inter-TSF Trusted Channel			X	Y	
FTP_TRP.1 Trusted Path	X			Y	Mandatory

3 Security Problem Description

The security problem is described in terms of the threats that the TOE is expected to address, assumptions about its Operational Environment, and any organisational security policies that the TOE is expected to enforce.

3.1 Threats

Table 5: Threat Descriptions

Threat	Description
T.PRIVILEGED_USER_ERROR	A privileged user or non-person entity (NPE) improperly exercises or adversely affects the TOE, resulting in unauthorised services, ineffective security mechanisms, or unintended circumvention of security mechanisms.
T.TSF_FAILURE	Security mechanisms of the TOE may fail, leading to a compromise of the TOE Security Function (TSF).
T.UNAUTHENTICATED_TRANSACTIONS	Relying parties within an information system depend on the TOE to accurately bind subjects to their credentials for use in authenticating and providing privacy for transactions. Without the proper binding provided by the TOE, relying parties cannot ensure adequate access controls on sensitive information, ensure transactional integrity, ensure proper accountability, and/or enforce non- repudiation.
T.UNAUTHORISED_ACCESS	A malicious user, process, or external IT entity intentionally circumvents TOE security mechanisms.
T.UNAUTHORISED_UPDATE	A malicious party attempts to supply the end user with an update to the product that may compromise the security features of the TOE.
T.UNDETECTED_ACTIONS	Remote users or external IT entities may take actions that adversely affect the security of the TOE.
T.USER_DATA_REUSE	A malicious user, process, or external IT entity may gain access to user data that is not cleared when resources are reallocated.
T.WEAK_CRYPTO	A weak hash or signature scheme may be compromised by an attacker and used to apply integrity checks to malicious content so that it appears legitimate.

3.2 Assumptions

Assumptions are articulated in line with those as stated in PP_CA_V2.1 in Table 6 below.

Table 6: Assumption Descriptions

Assumptions	Description
A.NO_GENERAL_PURPOSE	It is assumed that there are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
A.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment.
A.TRUSTED_ADMIN	TOE Administrators are assumed to follow and apply all administrator guidance in a trusted manner.

3.3 Organisational Security Policies

Organisational Security Policies are articulated in line with those as stated in PP_CA_V2.1 in Table 7 below.

Table 7: Organisational Security Policies

Organisational Security Policies	Description
P.ACCESS_BANNER	The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.

4 Security Objectives

In some cases, an objective is addressed only by requirements that are either selection-based or optional. In these cases, if none of those requirements are included in the ST, the ST author does not include that objective in the ST.

Security Objectives for the TOE

Security Objectives for the TOE are articulated in line with those as stated in PP_CA_V2.1 in Table 8 below.

Table 8: Description of Security Objectives for the TOE

Security Objective	Description
O.AUDIT_LOSS_RESPONSE	The TOE will respond to possible loss of audit records when audit trail storage is full or nearly full by restricting auditable events.
O.AUDIT_PROTECTION	The TOE will protect audit records against unauthorised access, modification, or deletion to ensure accountability of user actions.
O.CERTIFICATES	The TSF must ensure that certificates, certificate revocation lists, and certificate status information are valid.
O.CONFIGURATION_MANAGEMENT	The TOE will conduct configuration management to assure identification of system connectivity (software, hardware, and firmware), and components (software, hardware, and firmware), auditing of configuration data, and controlling changes to configuration items.
O.DISPLAY_BANNER	The TOE will display an advisory warning regarding use of the TOE.
O.INTEGRITY_PROTECTION	The TOE will provide appropriate integrity protection for TSF data and software, and any user data stored by the TOE.
O.NON_REPUDIATION	The TOE will prevent a subscriber from avoiding accountability for sending a message by providing evidence that the subscriber sent the message, and control communications from unknown source.
O.PROTECTED_COMMUNICATIONS	The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorised IT entities. The TOE will protect data assets when they are being transmitted to and from the TOE, including through intervening untrusted components.

O.RECOVERY	The TOE will have the capability to store and recover to a previous state at the direction of the administrator (e.g. provide support for archival and recovery capabilities).
O.RESIDUAL_INFORMATION_CLEARING	The TOE will ensure that any data contained in a protected resource is not available when the resource is reallocated.
O.SESSION_LOCK	The TOE will provide mechanisms that mitigate the risk of unattended sessions being hijacked.
O.SYSTEM_MONITORING	The TOE will provide the capability to generate audit data. The TOE will record in audit records: date and time of action and the entity responsible for the action.
O.TOE_ADMINISTRATION	The TOE will provide mechanisms to ensure that only privileged users are able to log in and configure the TOE, and provide protections for logged-in users. The TOE will ensure that administrative responsibilities are separated across different roles to mitigate the impact of improper administrative activities or unauthorised administrative access.
O.TSF_SELF_TEST	The TOE will provide integrity protection to detect modifications to firmware, software, and archived data.
O.VERIFIABLE_UPDATES	The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the administrator to be unaltered and from a trusted source.

4.1 Security Objectives for the Operational Environment

Security Objectives for the Operational Environment are articulated in line with those as stated in PP_CA_V2.1 in Table 9 below.

Table 9: Security Objectives for the Operational Environment

Security Objectives for the Operational Environment	Description
OE.CRYPTOGRAPHY	The Operational Environment provides cryptographic services that can be invoked by the TSF in order to perform security functionality.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.KEY_ARCHIVAL	The Operational Environment provides the ability to use split knowledge procedures to enforce two-party control to export keys necessary to resume CA functionality if the TSF should fail.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.PUBLIC_KEY_PROTECTION	The Operational Environment provides protection for specified public keys associated with CA functions.
OE.SESSION_PROTECTION_REMOTE	The Operational Environment provides the ability to lock or terminate remote administrative sessions.
OE.TOE_ADMINISTRATION	The Operational Environment provides specified management capabilities required for the overall operation of a Certificate Authority, and the ability to restrict access to a subset of the capabilities as specified in the ST.
OE.TRUSTED_ADMIN	The administrator of the TOE is not careless, wilfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.
OE.TRUSTED_PLATFORM	The operating system on which the TOE has been installed is securely configured, regularly patched, and not subject to unauthorised access.

4.2 Security Objectives Rationale

Table 10 illustrates the correspondence between the threats, assumptions, and organisational security policies described in the security problem definition and the TOE/environmental objectives that are satisfied in order to ensure that the threats are sufficiently mitigated by the TSF and the Operational Environment.

Table 10: Security Requirements Rationale

SPD Element	Objective	Requirements
A.NO_GENERAL_PURPOSE It is assumed that there are no general- purpose computing capabilities (e.g. compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.	**OE.NO_GENERAL_PURPOSE** There are no general-purpose computing capabilities (e.g. compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.	N/A
A.PHYSICAL Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment.	**OE.PHYSICAL** Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.	N/A
A.TRUSTED_ADMIN TOE Administrators are assumed to follow and apply all administrator guidance in a trusted manner.	**OE.TRUSTED_ADMIN** The administrator of the TOE is not careless, wilfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.	N/A
T.PRIVILEGED_USER_ERROR A privileged user or non-person entity (NPE) improperly exercises or adversely affects the TOE, resulting in unauthorised services, ineffective security mechanisms, or unintended circumvention of security mechanisms.	**O.AUDIT_LOSS_RESPONSE** The TOE will respond to possible loss of audit records when audit trail storage is full or nearly full by restricting auditable events.	FAU_ADP_EXT.1, FAU_STG.4
	O.AUDIT_PROTECTION The TOE will protect audit records against unauthorised access, modification, or deletion to ensure accountability of user actions.	FAU_ADP_EXT.1, FAU_STG.1(1), FAU_STG.1(2), FAU_STG_EXT.2
	O.TOE_ADMINISTRATION The TOE will provide mechanisms to ensure that only privileged users are able to log in and configure the TOE, and provide protections for logged-in users. The TOE will ensure that administrative responsibilities are separated across different roles to mitigate the impact of improper administrative activities or unauthorised administrative access.	FIA_AFL.1, FIA_PMG_EXT.1, FIA_UAU.7, FIA_UAU_EXT.1, FIA_UIA_EXT.1, FMT_MOF.1(1), FMT_MOF.1(2), FMT_MOF.1(3), FMT_MOF.1(4), FMT_MOF.1(5), FMT_MTD.1, FMT_SMF.1, FMT_SMR.2, FPT_APW_EXT.1, FTA_SSL.3, FTA_SSL.4

T.TSF_FAILURE Security mechanisms of the TOE may fail, leading to a compromise of the TSF.	**O.TSF_SELF_TEST** The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly. The TOE will provide integrity protection to detect modifications to firmware, software, and archived data.	FPT_TST_EXT.1, FPT_TST_EXT.2
T.UNAUTHENTICATED_TRANSACTIONS Relying parties within an information system depend on the TOE to accurately bind subjects to their credentials for use in authenticating and providing privacy for transactions. Without the proper binding provided by the TOE, relying parties cannot ensure adequate access controls on sensitive information, ensure transactional integrity, ensure proper accountability, and/or enforce non-repudiation.	**O.CERTIFICATES** The TSF must ensure that certificates, certificate revocation lists, and certificate status information are valid.	FDP_CER_EXT.1, FDP_CER_EXT.2, FDP_CER_EXT.3, FDP_CRL_EXT.1, FDP_CSI_EXT.1, FDP_OCSPG_EXT.1, FDP_STG_EXT.1, FIA_X509_EXT.1, FIA_X509_EXT.2, FPT_NPE_EXT.1
	O.CONFIGURATION_MANAGEMENT The TOE will conduct configuration management to assure identification of system connectivity (software, hardware, and firmware), and components (software, hardware, and firmware), auditing of configuration data, and controlling changes to configuration items.	FDP_CER_EXT.1, FDP_CER_EXT.4, FDP_CRL_EXT.1, FDP_OCSPG_EXT.1, FMT_MOF.1(1), FMT_MOF.1(2), FMT_MOF.1(3), FMT_MOF.1(4), FMT_MOF.1(5), FMT_MTD.1, FPT_NPE_EXT.1
	O.INTEGRITY_PROTECTION The TOE will provide appropriate integrity protection for TSF data and software, and any user data stored by the TOE.	FCS_CDP_EXT.1, FCS_CKM_EXT.5, FDP_ITT.1, FPT_ITT.1
	O.NON_REPUDIATION The TOE will prevent a subscriber from avoiding accountability for sending a message by providing evidence that the subscriber sent the message, and control communications from unknown source.	FCO_NRO_EXT.2, FCO_NRR_EXT.2
T.UNAUTHORISED_ACCESS A malicious user, process, or external IT entity intentionally circumvents TOE security mechanisms.	**O.PROTECTED_COMMUNICATIONS** The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorised IT entities. The TOE will protect data assets when they are being transmitted to and from the TOE, including through intervening untrusted components.	FCS_CDP_EXT.1, FCS_HTTPS_EXT.1, FCS_RBG_EXT.1, FCS_STG_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.1, FCS_TLSS_EXT.2 (selection-based), FDP_ITT.1, FPT_ITT.1,

		FPT_KST_EXT.1, FPT_KST_EXT.2, FPT_SKP_EXT.1, FPT_SKY_EXT.1, FTP_ITC.1, FTP_TRP.1
	O.TOE_ADMINISTRATION The TOE will provide mechanisms to ensure that only privileged users are able to log in and configure the TOE, and provide protections for logged-in users. The TOE will ensure that administrative responsibilities are separated across different roles in order to mitigate the impact of improper administrative activities or unauthorised administrative access.	FIA_AFL.1 (selection-based), FIA_PMG_EXT.1 (selection-based), FIA_UAU.7 (selection-based), FIA_UAU_EXT.1, FIA_UIA_EXT.1, FMT_MOF.1(1), FMT_MOF.1(2), FMT_MOF.1(3), FMT_MOF.1(4), FMT_MOF.1(5), FMT_MTD.1, FMT_SMF.1, FMT_SMR.2, FPT_APW_EXT.1 (selection- based), FTA_SSL.3 (optional), FTA_SSL.4
	OE.CRYPTOGRAPHY The Operational Environment provides cryptographic services that can be invoked by the TSF in order to perform security functionality.	FCS_CKM.1, FCS_CKM.2, FCS_CKM_EXT.1(1), FCS_CKM_EXT.1(2), FCS_CKM_EXT.1(3), FCS_CKM_EXT.1(4), FCS_CKM_EXT.4, FCS_CKM_EXT.7, FCS_CKM_EXT.8, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_COP.1(5)
T.UNAUTHORISED_UPDATE A malicious party attempts to supply the end user with an update to the product that may compromise the security features of the TOE.	**O.VERIFIABLE_UPDATES** The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the administrator to be unaltered and from a trusted source.	FCS_CDP_EXT.1, FCS_COP.1(2) FIA_X509_EXT.2, FPT_TUD_EXT.1
T.UNDETECTED_ACTIONS	**O.AUDIT_LOSS_RESPONSE** The TOE will respond to possible loss of audit records when audit trail storage	FAU_ADP_EXT.1, FAU_STG.4

Remote users or external IT entities may take actions that adversely affect the security of the TOE.	is full or nearly full by restricting auditable events.	
	O.AUDIT_PROTECTION The TOE will protect audit records against unauthorised access, modification, or deletion to ensure accountability of user actions.	FAU_ADP_EXT.1, FAU_STG.1(1) (selection- based), FAU_STG.1(2) (selection-based), FAU_STG_EXT.2 (selection-based)
	O.SYSTEM_MONITORING The TOE will provide the capability to generate audit data and send those data to an external IT entity. The TOE will record in audit records: date and time of action and the entity responsible for the action.	FAU_ADP_EXT.1, FAU_GEN.1, FAU_GEN.2, FAU_SAR.1 (selection-based), FAU_SAR.3 (selection-based), FAU_GCR_EXT.1, FAU_SCR_EXT.1 (selection- based), FAU_SEL.1 (selection-based), FAU_STG_EXT.1 (selection-based), FIA_UIA_EXT.1, FPT_STM.1
T.USER_DATA_REUSE A malicious user, process, or external IT entity may gain access to user data that is not cleared when resources are reallocated.	**O.RESIDUAL_INFORMATION_CLEARING** The TOE will ensure that any data contained in a protected resource is not available when the resource is reallocated.	FDP_RIP.1
T.WEAK_CRYPTO A weak hash or signature scheme may be compromised by an attacker and used to apply integrity checks to malicious content so that it appears legitimate.	**O.PROTECTED_COMMUNICATIONS** The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorised IT entities. The TOE will protect data assets when they are being transmitted to and from the TOE, including through intervening untrusted components.	FCS_CDP_EXT.1, FCS_CKM.1 (selection- based), FCS_CKM.2 (selection-based), FCS_CKM_EXT.1(1) (selection-based), FCS_CKM_EXT.1(2) (selection-based), FCS_CKM_EXT.1(3) (selection-based), FCS_CKM_EXT.1(4) (selection-based), FCS_CKM_EXT.4 (selection-based), FCS_CKM_EXT.7 (selection-based), FCS_CKM_EXT.8

		(selection-based), FCS_COP.1(1) (selection-based), FCS_COP.1(2) (selection-based), FCS_COP.1(3) (selection-based), FCS_COP.1(4) (selection-based), FCS_COP.1(5) (optional), FCS_HTTPS_EXT.1 (selection-based), FCS_IPSEC_EXT.1 (selection-based), FCS_RBG_EXT.1 (selection-based), FCS_STG_EXT.1, FCS_TLSS_EXT.1 (selection-based), FCS_TLSS_EXT.2 (selection-based), FDP_ITT.1 (selection-based), FIA_PSK_EXT.1 (selection-based), FPT_ITT.1 (selection- based), FPT_KST_EXT.1, FPT_KST_EXT.2, FPT_SKP_EXT.1, FPT_SKY_EXT.1 (optional), FPT_SKY_EXT.2 (selection-based), FTP_ITC.1 (selection-based), FTP_TRP.1
	O.VERIFIABLE_UPDATES The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the administrator to be unaltered and from a trusted source.	FCS_CDP_EXT.1, FCS_COP.1(2), IFIA_X509_EXT.2, FPT_TUD_EXT.1
	OE.CRYPTOGRAPHY The Operational Environment provides cryptographic services that can be invoked by the TSF in order to perform security functionality.	FCS_CKM.1, FCS_CKM.2, FCS_CKM_EXT.1(1), FCS_CKM_EXT.1(2), FCS_CKM_EXT.1(3), FCS_CKM_EXT.1(

		4), FCS_CKM_EXT.4, FCS_CKM_EXT.7, FCS_CKM_EXT.8, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_COP.1(5)
	OE.KEY_ARCHIVAL The Operational Environment provides the ability to use split knowledge procedures to enforce two-party control to export keys necessary to resume CA functionality if the TSF should fail.	
P.ACCESS_BANNER The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.	**O.DISPLAY_BANNER** The TOE will display an advisory warning regarding use of the TOE.	FTA_TAB.1

5 Extended Components Definition

Extended Security Functional Requirements

Extended security functional components are identified below and are further articulated in PP_CA_v2.1

- FAU_ADP_EXT.1 Audit Dependencies
- FAU_GCR_EXT.1 Generation of Certificate Repository
- FAU_SCR_EXT.1 Certificate Repository Review
- FAU_STG_EXT.1 External Audit Trail Storage
- FAU_STG_EXT.2 Audit Data Retention
- FCO_NRO_EXT.2 Certificate-Based Proof of Origin
- FCO_NRR_EXT.2 Certificate-Based Proof of Receipt
- FCS_CDP_EXT.1 Cryptographic Dependencies
- FCS_CKM_EXT.1 (1) Cryptographic Asymmetric Key Generation
- FCS_CKM_EXT.1 (2) Key Generation Key Encryption Keys
- FCS_CKM_EXT.1 (3) Key Generation for Key Encryption Keys (TOE Key Archival)
- FCS_CKM_EXT.1 (4) Generation of Key Shares
- FCS_CKM_EXT.4 Cryptographic Key Destruction
- FCS_CKM_EXT.5 Public Key Integrity
- FCS_CKM_EXT.6 TOE Key Archival
- FCS_CKM_EXT.7 Key Generation for KEKs
- FCS_CKM_EXT.8 Key Hierarchy Entropy
- FCS_HTTPS_EXT.1 HTTPS Protocol
- FCS_KSH_EXT.1 Key Sharing
- FCS_RBG_EXT.1 Cryptographic Random Bit Generation
- FCS_STG_EXT.1 Cryptographic Key Storage
- FCS_TLSS_EXT.2 TLS Server Protocol with Mutual Authentication
- FDP_CER_EXT.1 Certificate Profiles
- FDP_CER_EXT.2 Certificate Request Matching
- FDP_CER_EXT.3 Certificate Issuance Approval
- FDP_CRL_EXT.1 Certificate Revocation List Validation
- FDP_CSI_EXT.1 Certificate Status Information
- FDP_OCSPG_EXT.1 OCSP Basic Response Generation
- FDP_SDP_EXT.1 User Sensitive Data Protection

- FDP_STG_EXT.1 Public Key Protection
- FIA_PMG_EXT.1 Password Management
- FIA_X509_EXT.1 Certificate Validation
- FIA_X509_EXT.2 Certificate-Based Authentication
- FIA_UAU_EXT.1 Authentication Mechanism
- FIA_UIA_EXT.1 User Identification and Authentication
- FPT_APW_EXT.1 Protection of Privileged User Passwords
- FPT_KST_EXT.1 No Plaintext Key Export
- FPT_KST_EXT.2 TSF Key Protection
- FPT_NPE_EXT.1 NPE Constraints
- FPT_SKP_EXT.1 Protection of Keys
- FPT_SKY_EXT.1 Split Knowledge Procedures
- FPT_TUD_EXT.1 Trusted Update
- FTA_SSL_EXT.1 TSF-Initiated Session Locking

5.1 Extended Security Assurance Requirements

No extended security assurance requirements are defined for this security target.

6 Security Requirements

The SFRs included in this section are derived from Part 2 of the CC, with additional extended functional components.

Part 2 of the CC defines operations on SFRs: assignments, selections, iterations, and refinements. This document uses the following font conventions to identify the operations defined by the CC:

- **Refinement** (denoted by **bold text**): is used to add details to a requirement, and thus further restricts a requirement.
- *Selection* (denoted by *italicised text*): is used to select one or more options provided by the CC in stating a requirement.
- *[Assignment]* (denoted by *italicised text* in square brackets): is used to assign a specific value to an unspecified parameter, such as the length of a password.
- (Iteration): is identified with a number inside parentheses (e.g. “(1)”).
- Extended SFRs: are identified by having a label “EXT” after the SFR name.

Spelling in this document has been changed to Australian English as a grammatical change where appropriate, but this does not represent an operation on controls.

TOE Security Functional Requirements

The SFRs included in this section are derived from Part 2 of the CC, with additional extended functional components.

The following table lists the SFRs that are defined in this section as well as any auditable events associated with their enforcement.

Table 11: Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents	Retention Normal/Extended	Responsible TSF or OE Component
FAU_ADP_EXT.1	None.	None.	N/A	
FAU_GCR_EXT.1	None.	None.	N/A	
FAU_GEN.1	None.	None.	N/A	
FAU_GEN.2	None.	None.	N/A	
FAU_SEL.1	All modifications to the audit configuration that occur while the audit collection functions are operating.	None.	Normal.	TSF.
FDP_RIP.1	None.	None.	N/A	
FIA_X509_EXT.1	Failed certificate validations.	None.	Normal	TSF
FIA_X509_EXT.2	Failed authentications.	None.	Normal	TSF
FIA_UAU_EXT.1	All uses of the authentication mechanism used for access to TOE related functions.	Origin of the attempt (e.g., IP address).	Normal	TSF

Requirement	Auditable Events	Additional Audit Record Contents	Retention Normal/Extended	Responsible TSF or OE Component
FIA_UIA_EXT.1	All use of the identification and authentication mechanism used for TOE related roles.	Provided user identity. Origin of the attempt (e.g., IP address).	Normal	TSF
FMT_MOF.1(1)	None.	None.	N/A	
FMT_MOF.1(2)	None.	None.	N/A	
FMT_MOF.1(3)	None.	None.	N/A	
FMT_MOF.1(4)	None.	None.	N/A	
FMT_MOF.1(5)	None.	None.	N/A	
FMT_MTD.1	None.	None.	N/A	
FMT_SMF.1	None.	None.	N/A	
FMT_SMR.2	Modifications to the group of users that are part of a role.	Modifications to the group of users that are part of a role.	Extended.	TSF
FPT_FLS.1	Invocation of failures under this requirement.	Indication that the TSF has failed with the type of failure that occurred.	Normal.	TSF
FPT_KST_EXT.1	None.	None.	N/A	
FPT_KST_EXT.2	All unauthorised attempts to use TOE secret and private keys.	Identifier of user or process that attempted access.	Normal.	OE
FPT_RCV.1	The fact that a failure or service discontinuity occurred; resumption of the regular operation.	The type of failure or service discontinuity.	Extended	TSF
FPT_SKP_EXT.1	None.	None.	N/A	
FPT_STM.1	Changes to the time.	The old and new values for the time.	Normal.	OE
FPT_TUD_EXT.1	Initiation of update.	Version number.	Extended.	TSF
FTA_SSL.4	The termination of an interactive session.	None.	Normal.	TSF
FTA_TAB.1	None.	None.	N/A	
FTP_TRP.1	Initiation of the trusted channel. Termination of the trusted channel. Failures of the trusted path functions.	Identification of the claimed user identity.	Normal.	TSF

Requirement	Auditable Events	Additional Audit Record Contents	Retention Normal/Extended	Responsible TSF or OE Component
FDP_STG_EXT.1	Changes to the trusted public keys and certificates relevant to TOE functions, including additions and deletions.	The public key and all context information associated with the key.	Normal.	TSF
FDP_CER_EXT.1	Certificate Generation.	Success: certificate object identifier.	Extended.	TSF
FDP_CER_EXT.2	Linking of certificate to certificate request.	Success: certificate object identifier, link to certificate request object identifier. Failure: Reason for failure, link to Certificate request object identifier.	Extended.	TSF
FDP_CER_EXT.3	Linking of certificate to certificate request	Success: certificate object identifier, link to certificate request object identifier. Failure: Reason for failure, link to Certificate request object identifier	Normal.	TSF
FPT_NPE_EXT.1	All changes to NPE rule sets and NPE associations.	The changes made to the NPE rule sets and associations.	Extended.	TSF
FTA_SSL.3	The termination of a remote session by the session termination mechanism.	None.	Normal.	TSF
FAU_SCR_EXT.1	None.	None.	N/A	
FAU_SAR.1	None.	None.	N/A	
FAU_SAR.3	None.	None.	N/A	
FAU_STG.1(1)	None.	None.	N/A	
FAU_STG.1(2)	None.	None.	N/A	
FAU_STG_EXT.1	None.	None.	N/A	
FAU_STG_EXT.2	None.	None.	N/A	
FCO_NRO_EXT.2				
FCO_NRR_EXT.2	None.	None.	N/A	
FCS_CKM.1	All occurrences of non-ephemeral and ephemeral key generation for TOE related functions.	Success: public key generated.	Normal	OE

Requirement	Auditable Events	Additional Audit Record Contents	Retention Normal/Extended	Responsible TSF or OE Component
FCS_CKM.2	All occurrences of non- ephemeral and ephemeral key establishment for TOE related functions.	Success: key established.	Normal.	OE
FCS_CKM_EXT.1(2)	None.	None.	N/A	
FCS_CKM_EXT.1(3)	None	None.	N/A	
FCS_CKM_EXT.1(4)	None.	None.	N/A	
FCS_CKM_EXT.4	Failure of the key destruction process for TOE related keys.	Identity of object or entity being cleared.	Normal.	OE
FCS_CKM_EXT.5	Detection of integrity violation for stored TSF data.	None.	Normal.	OE
FCS_CKM_EXT.6	All key archival actions.	None.	Extended	OE
FCS_CKM_EXT.7	None.	None.	N/A	
FCS_CKM_EXT.8	None.	None.	N/A	
FCS_COP.1(1)	None.	None.	N/A	
FCS_COP.1(2)	All occurrences of signature generation using a CA signing key.	Name/identifier of object being signed. Identifier of key used for signing.	Extended.	TSF
	Failure in signature generation	None.	Normal.	TSF
FCS_COP.1(3)	None.	None.	N/A	
FCS_COP.1(4)	None.	None.	N/A	
FCS_HTTPS_EXT.1	Failure to establish a HTTPS session. Establishment/ Termination of a HTTPS	Reason for failure. Non-TOE endpoint of connection (IP address) for both successes and failures	Normal.	TSF
FCS_RBG_EXT.1	None.	None.	N/A	
FCS_TLSS_EXT.1	Failure to establish a TLS session. Establishment/Termination of a TLS session.	Reason for failure. None.	Normal.	TSF
FDP_CRL_EXT.1	Failure to generate CRL.	None.	Normal.	TSF
FDP_ITT.1	None.	None.	N/A	
FDP_OCSPG_EXT.1	Failure to generate certificate status information.	None.	Extended.	TSF

Requirement	Auditable Events	Additional Audit Record Contents	Retention Normal/Extended	Responsible TSF or OE Component
FIA_AFL.1	The reaching of the threshold for the unsuccessful authentication attempts. The action taken. The re- enablement of disabled non-administrative accounts.	None.	Normal.	TSF
FIA_CMCS_EXT.1	CMC requests (generated or received) containing certificate requests or revocation requests. CMC responses issued.	Identifiers for all entities authenticating the request, including the entity providing client authentication for the CMC transport (if any). The submitted request. Any signed response.	Extended	
FIA_CMCC_EXT.1	CMC requests (generated or received) containing certificate requests or revocation requests. CMC responses issued.	Identifiers for all entities authenticating the request, including the entity providing client authentication for the CMC transport (if any). The submitted request. Any signed response.	Extended	
FIA_PMG_EXT.1	None.	None.	N/A	
FIA_UAU.7	None.	None.	N/A	
FPT_APW_EXT.1	None.	None.	N/A	
FPT_ITT.1	None.	None.	N/A	
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.	Normal.	TSF

6.1.1 Security Audit (FAU)

6.1.1.1 FAU_ADP_EXT.1 Audit Dependencies

FAU_ADP_EXT.1.1	The TSF shall implement audit functionality and *no additional audit functionality* in order to perform audit operations on the following audit data: [*As identified in Table 11.*]

6.1.1.2 FAU_GCR_EXT.1 Generation of Certificate Repository

FAU_GCR_EXT.1.1	The TSF shall *store* certificates and *CRLs* issued by the TSF.

6.1.1.3 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1	The TSF shall generate **and *no other actions*** an audit record of the following auditable events: a) start-up of the **TSF** audit functions, b) all auditable events for the [*not specified*] level of audit, and c) [*all administrative actions invoked through the TSF interface,* d) [*specifically defined auditable events listed in Table 11*]].
FAU_GEN.1.2	The TSF shall ***include*** within each audit record at least the following information: a) date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event, and b) for each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [*information specified in column three of Table 11*].

6.1.1.4 FAU_GEN.2 User Identity Association

FAU_GEN.2.1	For audit events resulting from actions of identified users, the TSF shall be able to ***associate*** each auditable event with the identity of the user that caused the event.

6.1.1.5 FAU_SAR.1 Audit Review

FAU_SAR.1.1	The TSF shall provide [*Auditors*] with the capability to read all information from the audit records.
FAU_SAR.1.2	The TSF shall provide the audit records in a manner suitable for the **Auditor** to interpret the information.

6.1.1.6 FAU_SAR.3 Selectable Audit Review

FAU_SAR.3.1	The TSF shall provide the ability to apply [searches] of audit data based on *all certificate fields, user identity store and CMDB identifier* associated with the event.

6.1.1.7 FAU_STG.1(1) Protected Audit Trail Storage

FAU_STG.1.1(1)	The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.
FAU_STG.1.2(1)	The TSF shall be able to [*prevent*] unauthorised modifications to the stored audit records in the audit trail.

6.1.1.8 FAU_STG.1(2) Protected Audit Trail Storage (Archive Data)

FAU_STG.1.1(2)	The TSF shall protect the stored audit records **with extended retention requirements** in the audit trail from deletion prior to their retention period by an auditor.
FAU_STG.1.2(2)	The TSF shall be able to [prevent] modifications to the stored audit records **with extended retention requirements** in the audit trail.

6.1.1.9 FAU_STG.4 Prevention of Audit Data Loss

FAU_STG.4.1	The TSF shall prevent audited events, except those taken by the Auditor and manual archival of audit logs if the audit trail cannot be written to.

6.1.1.10 FAU_SCR_EXT.1 Certificate Repository Review

FAU_SCR_EXT.1.1	The TSF shall provide the capability to search for certificates containing specified values of the following certificate fields: • *subject name,* • *individual components of subject alternative name,* • *subject ID,* • *issuer ID,* • *algorithm ID,* • *public key,* • *key usage,* • *extended key usage,* • *serial number,* • *[certificate status,* • *expiry date,* • *any other certificate fields],* returning all matching certificates and user and device identity store information.

6.1.1.11 FAU_STG_EXT.2 Audit Data Retention

6.1.1.12

FAU_STG_EXT.2.1	The TSF shall apply the following rules for retention of audit data: • Audit records required to have extended retention shall be retained at least until an auditor configured extension beyond the validity of all certificates impacted by the event. • [*An auditor configured retention period for rolling to-disk log,* • *deletion after forwarding to a SIEM in the Operational Environment*].

6.1.2 Communications (FCO)

6.1.2.1 FCO_NRO_EXT.2 Certificate-Based Proof of Origin

FCO_NRO_EXT.2.1	The TSF shall provide proof of origin for certificates it issues in accordance with the digital signature requirements using a mechanism in accordance with RFC 5280 and FCS_COP.1(2).

FCO_NRO_EXT.2.1	The TSF shall provide proof of origin for certificate status information it issues in accordance with the digital signature requirements in *CRLs (RFC 5280)* and FCS_COP.1(2)
FCO_NRO_EXT.2.3	The TSF shall require and verify proof of origin for certificate requests it receives via *CMC using mechanisms in accordance with FIA_CMCS_EXT.1*
FCO_NRO_EXT.2.4	The TSF shall require and verify proof of origin for public keys contained in certificate requests it receives via *proof-of-possession mechanisms in CMC using mechanisms in accordance with FIA_CMCS_EXT.1*
FCO_NRO_EXT.2.5	The TSF shall require and verify proof of origin for revocation requests it receives via ***CMC using mechanisms in accordance with FIA_CMCS_EXT.1,], [*assignment*: support manual processes for revocation requests and responses*]**

6.1.2.2 FCO_NRR_EXT.2 Certificate-Based Proof of Receipt

FCO_NRR_EXT.2.1	The TSF shall provide proof of receipt for CMC by providing signed responses using mechanisms in accordance with FIA_CMCS_EXT.1.

6.1.3 Cryptographic Support (FCS)

In this section iteration has been applied to the cryptographic support requirements as relate to the following trusted security functional interfaces; (a)HSM and (b)Golang webserver and Consul.

6.1.3.1 FCS_CDP_EXT.1 Cryptographic Dependencies

FCS_CDP_EXT.1.1	The TSF shall *invoke interfaces provided by the Operational Environment* in order to perform *all* cryptographic operations.

6.1.3.2 FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1 (a)	The TSF shall ***invoke interfaces provided by the Operational Environment to generate* asymmetric** cryptographic keys in accordance with the specified key generation algorithm: • ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3,*** • ***ECC schemes using "NIST curves" P-256 and P-384 that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4,*** *and specified cryptographic key sizes [equivalent to or greater than a symmetric key strength of 128 bits].*
FCS_CKM.1.1 (b)	The TSF shall ***invoke interfaces provided by the Operational Environment to generate* asymmetric** cryptographic keys in accordance with the specified key generation algorithm: • ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3,*** • ***ECC schemes using "NIST curves" P-256 and P-384 that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4,*** *and specified cryptographic key sizes [equivalent to or greater than a symmetric key strength of 128 bits].*
FCS_CKM.1.1 (c)	The TSF shall ***invoke interfaces provided by the Operational Environment to generate* asymmetric** cryptographic keys in accordance with the specified key generation algorithm: • ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3,*** • ***ECC schemes using "NIST curves" P-256 and P-384 that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4,*** • ***and specified cryptographic key sizes [equivalent to or greater than a symmetric key strength of 128 bits].***

6.1.3.3 FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1	The TSF shall ***invoke interfaces provided by the Operational Environment to perform*** **key establishment** in accordance with a specified cryptographic key **establishment** algorithm • ***RSA-based key establishment schemes that meet the following: NIST Special Publication 800-56B Revision 1, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorisation Cryptography",*** • ***elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for PairWise Key Establishment Schemes Using Discrete Logarithm Cryptography",*** • ***key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526.***

6.1.3.4 FCS_CKM_EXT.1(1) Symmetric Key Generation for DEKs

Jellyfish CA utilises a Registration Authority for management of keys, hence this SFR us not selected.

6.1.3.5 FCS_CKM_EXT.1(2) Key Generation Encryption Keys

FCS_CKM_EXT.1.1(2)	The TSF shall be able to *invoke interfaces in the Operational Environment to generate asymmetric KEKs of [greater than or equal to 256 bit]* security strength in accordance with FCS_CKM.1, *128-bit and 256-bit* symmetric KEKs using • *a key generation capability of the Operational Environment.*

6.1.3.6 FCS_CKM_EXT.1(3) Key Generation for Key Encryption Keys (TOE Key Archival)

FCS_CKM_EXT.1.1(3)	The TSF shall be able to *invoke interfaces provided by the Operational Environment to generate asymmetric KEKs of [security strength greater than or equal to 112 bits]* security strength, symmetric KEKs of size *128-bit and 256-bit* using • *an Operational Environment-provided mechanism that combines Key Shares and produces a KEK,* for the archival and recovery of TOE keys from two or more shares according to a key sharing mechanism.

6.1.3.7 FCS_CKM_EXT.1(4) Generation of Key Shares

FCS_CKM_EXT.1.1(4)	The TSF shall be able to *invoke interfaces provided by the* Operational Environment *to generate* key shares of strength *greater than or equal to the security strength of the KEK defined in FCS_CKM_EXT.1(3)* for the key sharing mechanism indicated in FCS_CKM_EXT.1(3).

6.1.3.8 FCS_CKM_EXT.4 Cryptographic Key Destruction

FCS_CKM_EXT.4.1	The TSF shall *invoke interfaces provided by the Operational Environment to destroy* all cryptographic keys and critical security parameters which are not permanently protected from export by hardware when no longer required, in accordance with the specified cryptographic key destruction method • *for non-volatile memory that consists of the invocation of an interface provided by the underlying platform that*

	○ *instructs the underlying platform to destroy the abstraction that represents the key*.
FCS_CKM_EXT.4.2	The TSF shall *invoke interfaces provided by the Operational Environment to destroy* all plaintext keying material cryptographic security parameters when no longer needed.

6.1.3.9 FCS_CKM_EXT.5 Public Key Integrity

FCS_CKM_EXT.5.1	The TSF shall *protect* public keys used to meet CA requirements against undetected modification through the use of *digital signatures (in accordance with FCS_COP.1(2)).*
FCS_CKM_EXT.5.2	The *digital signature* used to protect a public key shall be verified upon each access to the key.

6.1.3.10 FCS_CKM_EXT.6 TOE Key Archival

FCS_CKM_EXT.6.1	The TSF shall *invoke interfaces in the Operational Environment to provide* a mechanism to protect TOE secret and private keys required for continuity of operations and *user private keys*.
FCS_CKM_EXT.6.2	The TSF shall *invoke interfaces in the Operational Environment to be able to* export the protected keys (in FCS_CKM_EXT.6.1) for the purpose of archival in encrypted form.
FCS_CKM_EXT.6.3	The TSF shall *invoke interfaces in the Operational Environment to be able to* import protected keys (in FCS_CKM_EXT.6.1) for the purpose of continued operations after failure.
FCS_CKM_EXT.6.4	The TSF shall *invoke interfaces in the Operational Environment to encrypt* the keys specified in FCS_CKM_EXT.6.1 in accordance with *FCS_COP.1(1) and FCS_CKM.1* using the KEK generated in accordance with FCS_CKM_EXT.1(3).
FCS_CKM_EXT.6.5	The TSF shall *invoke interfaces in the Operational Environment to decrypt* the keys specified in FCS_CKM_EXT.6.1 in accordance with *FCS_COP.1(1) and FCS_CKM.1* using the KEK generated in accordance with FCS_CKM_EXT.1(3).

6.1.3.11 FCS_CKM_EXT.7 Key Generation for KEKs

FCS_CKM_EXT.7.1	The Operational Environment shall support a hardware protected REK generated in accordance with FCS_CKM_EXT.1.1(2).
FCS_CKM_EXT.7.2	A REK shall not be able to be read from or exported from the hardware.
FCS_CKM_EXT.7.3	The TSF shall be able only to request encryption/decryption by the key and shall not be able to read, import, or export a REK.
FCS_CKM_EXT.7.4	A REK shall be generated *by a RBG in accordance with FCS_RBG_EXT.1.*

6.1.3.12 FCS_CKM_EXT.8 Key Hierarchy Entropy

FCS_CKM_EXT.8.1	The TSF shall provide a traceable hierarchy of keys (DEKs or KEKs) formed from combinations of by encrypting one key with another to a REK generated in accordance with FCS_RBG_EXT.1 using a hardware-based mechanism.
FCS_CKM_EXT.8.2	Key entropy for KEKs shall be preserved according to the sensitivity of the DEK, KEK, or key it encrypts.

FCS_CKM_EXT.8.3	Key entropy for DEKs shall be *256* bits in accordance with the sensitivity of the data encrypted.

6.1.3.13 FCS_COP.1(1) Cryptographic Operation (AES Encryption/Decryption)

FCS_COP.1.1(1)	The TSF shall ***invoke interfaces in the Operational Environment to perform*** [encryption and decryption] in accordance with a specified cryptographic algorithm: • ***AES-CBC (as defined in NIST SP 800-38A) mode,*** • ***AES-GCM (as defined in NIST SP 800-38D) mode,*** and cryptographic key size of ***256-bit****.*

6.1.3.14 FCS_COP.1(2) Cryptographic Operation (Cryptographic Signature)

FCS_COP.1.1(2)	The TSF shall ***invoke interfaces in the Operational Environment to perform*** cryptographic signature services in accordance with the following specified cryptographic algorithms • ***RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 3072 bits or greater that meets FIPS-PUB 186-4, "Digital Signature Standard",*** • ***elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater that meets FIPS PUB 186-4, "Digital Signature Standard" with "NIST curves" P-256, P-384 and P-521 (as defined in FIPS PUB 186-4, "Digital Signature Standard").***

6.1.3.15 FCS_COP.1(3) Cryptographic Operation (Cryptographic Hashing)

FCS_COP.1.1(3)	The TSF shall ***invoke interfaces in the Operational Environment to perform*** *[cryptographic hashing services]* in accordance with a specified cryptographic algorithm***, SHA-256, SHA-384, SHA-512*** **and message digest sizes*****, 256, 384, 512*** **bits** that meet the following: *[FIPS Pub 180-4, "Secure Hash Standard"].*

6.1.3.16 FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication)

FCS_COP.1.1(4)	The TSF shall ***invoke interfaces in the Operational Environment to perform*** *[keyed hash message authentication]* in accordance with a specified cryptographic algorithm **HMAC-*****SHA-256, SHA-384, SHA-512***, **key size *****256 bits***, **and message digest sizes *****256, 384, 512*** **bits** that meet the following: [*FIPS Pub 198-1,"The Keyed Hash Message Authentication Code"; FIPS Pub 180-4, "Secure Hash Standard"].*

6.1.3.17 FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1	The TSF shall implement the HTTPS protocol that complies with RFC 2818.
FCS_HTTPS_EXT.1.2	The TSF shall implement HTTPS using TLS.

6.1.3.18 FCS_IPSEC_EXT.1 IPSec Protocol

The TOE does not implement IPSec.

6.1.3.19 FCS_KSH_EXT.1 Key Sharing

Key sharing is not supported by this TOE.

6.1.3.20 FCS_RBG_EXT.1 Cryptographic Random Bit Generation

FCS_RBG_EXT.1.1	The TSF shall *invoke interfaces in the* Operational Environment *to perform* all deterministic random bit generation (RBG) services in accordance with NIST Special Publication 800-90A using *CTR_DRBG (AES).*
FCS_RBG_EXT.1.2	The deterministic RBG shall be seeded by an entropy source that accumulates entropy from *an Operational Environment-based noise source* with a minimum of *256 bits* of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and authorisation factors that it will generate.

6.1.3.21 FCS_STG_EXT.1 Cryptographic Key Storage

FCS_STG_EXT.1.1	Persistent private and secret keys shall be stored within the *Operational Environment* • *in a hardware cryptographic module.*

6.1.3.22 FCS_TLSC_EXT.2 TLS Client Protocol

The TOE does not implement TLS Client Protocol.

6.1.3.23 FCS_TLSS_EXT.1 TLS Server Protocol

FCS_TLSS_EXT.1.1	The TSF shall implement *TLS 1.2 (RFC 5246)* supporting the following ciphersuites: • *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289* • *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289* • *and no other ciphersuite.*
FCS_TLSS_EXT.1.2	The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and *TLS 1.1.*
FCS_TLSS_EXT.1.3	The TSF shall perform RSA key establishment with key size 2048 bits, 3072 bits, 4096 bits; generate EC Diffie-Hellman parameters over NIST curves secp256r1, secp384r1, secp521r1 and no other curves; generate Diffie-Hellman parameters of size 2048 bits and 3072 bits, no other size.
FCS_TLSS_EXT.1.4	The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: *secp256r1, secp384r1, secp521r1* and no other curves.

6.1.3.24 FCS_CKM_EXT.5 Public Key Integrity

FCS_CKM_EXT.5.1	The TSF shall *protect* public keys used to meet CA requirements against undetected modification through the use of *digital signatures (in accordance with FCS_COP.1(2)).*
FCS_CKM_EXT.5.2	The *digital signature* used to protect a public key shall be verified upon each access to the key.

6.1.4 User Data Protection (FDP)

6.1.4.1 FDP_CER_EXT.1 Certificate Profiles

FDP_CER_EXT.1.1	The TSF shall implement a certificate profile function and shall ensure that issued certificates are consistent with configured profiles.
FDP_CER_EXT.1.2	The TSF shall generate certificates using profiles that comply with requirements for certificates as specified in IETF RFC 5280, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", while ensuring that the following conditions are met: a) The version field shall contain the integer 2. b) The issuerUniqueID or subjectUniqueID fields are not populated. c) The serialNumber shall be unique with respect to the issuing Certification Authority. d) The validity field shall specify a notBefore value that does not precede the current time and a notAfter value that does not precede the value specified in notBefore. e) The issuer field is not empty. f) The signature field and the algorithm in the subjectPublicKeyInfo field shall contain the OID for a signature algorithm specified in FCS_COP.1(2). g) The following extensions are supported: a. subjectKeyIdentifier, b. authorityKeyIdentifier, c. basicConstraints, d. keyUsage, e. extendedKeyUsage, and f. certificatePolicy. h) A subject field containing a null Name (e.g., a sequence of zero relative distinguished names) is accompanied by a populated critical subjectAltName extension. i) The subjectKeyIdentifier extension is populated with a value unique for each public key contained in a certificate issued by the TSF. j) The authorityKeyIdentifier extension in any certificate issued by the TOE must be populated and must be the same as the subjectKeyIdentifier extension contained in the issuer's signing certificate. k) Populated keyUsage and extendedKeyUsage fields in the same certificate contain consistent values.
FDP_CER_EXT.1.3	The TSF shall be able to generate at least 20 bits of random for use in issued certificates to be included in *serialNumber* fields, where the random values are generated in accordance with FCS_RBG_EXT.1.

6.1.4.2 FDP_CER_EXT.2 Certificate Request Matching

FDP_CER_EXT.2.1	The TSF shall establish a linkage from certificate requests to issued certificates.

6.1.4.3 FDP_CER_EXT.3 Certificate Issuance Approval

FDP_CER_EXT.3.1	The TSF shall support the approval of certificates by *RA, CA Operations Staff, rules* issued according to a configured certificate profile.

6.1.4.4 FDP_CER_EXT.4 Non-X.509v3 Certificate Generation

The TOE only accepts X.509v3 Certificates.

6.1.4.5 FDP_CRL_EXT.1 Certificate Revocation List Validation

FDP_CRL_EXT.1.1	A TSF that issues CRLs shall verify that all mandatory fields in any CRL issued contain values in accordance with ITU-T Recommendation X.509. At a minimum, the following items shall be validated: a) If the version field is present, then it shall contain a 1. b) If the CRL contains any critical extensions, then the version field shall be present and contain the integer 1. c) If the issuer field contains a null Name (e.g., a sequence of zero relative distinguished names), then the CRL shall contain a critical issuerAltName extension. d) The signature and signatureAlgorithm fields shall contain the OID for a digital signature algorithm in accordance with FCS_COP.1(2). e) The thisUpdate field shall indicate the issue date of the CRL. f) The time specified in the nextUpdate field (if populated) shall not precede the time specified in the thisUpdate field.

6.1.4.6 FDP_CSI_EXT.1 Certificate Status Information

FDP_CSI_EXT.1.1	The TSF shall provide certificate status information whose format complies with *OCSP standards RFC 6960, ITU-T Recommendation X.509v2 CRL.*
FDP_CSI_EXT.1.2	The TSF shall support the approval of changes to the status of a certificate by *RA, CA operations staff, rules*.

6.1.4.7 FDP_ITT.1 Basic Internal Transfer Protection

FDP_ITT.1.1	The TSF shall prevent the [*disclosure, modification*] of user data when it is transmitted between physically separated parts of the TOE **through the use of *TLS/HTTPS.***

6.1.4.8 FDP_OSCPG_EXT.1 OSCP Basic Response Generation

FDP_OCSPG_EXT.1.1	The TSF shall ensure that all mandatory fields in the OCSP response contain values in accordance with the standards specified in FDP_CSI_EXT.1. At a minimum, the following items shall be enforced: a) The version field shall indicate a current version. b) The signatureAlgorithm field shall contain the object identifier (OID) for a digital signature algorithm in accordance with FCS_COP.1(2). c) The thisUpdate field shall indicate the time at which the status being indicated is known to be correct. d) The producedAt field shall indicate the time at which the OCSP responder signed the response. e) The time specified in the nextUpdate field (if populated) shall not precede the time specified in the thisUpdate field.

6.1.4.9 FDP_RIP.1 Subset Residual Information Protection

FDP_RIP.1.1	The TSF **and *Operational Environment*** shall ensure that any previous information content of a resource is made unavailable upon the ***deallocation of the resource from*** the following objects: • [*API Keys,* • *HSM Tokens,* • *support tickets,*

	• *organisations,* • *tenancies,* • *users, and* • *devices].*

6.1.4.10 FDP_SDP_EXT.1 User Sensitive Data Protection

The TOE does not store, process or communicate Personally Identifiable Information (PII).

6.1.4.11 FDP_STG_EXT.1 Public Key Protection

FDP_STG_EXT.1.1	The TSF shall use *access controlled storage* to protect the trusted public keys and certificates (trust store elements) used to validate local logon, trusted channel, and external communication to the CA.

6.1.5 Identification and Authentication (FIA)

6.1.5.1 FIA_AFL.1 Authentication Failure Handling

FIA_AFL.1.1	The TSF shall ***implement* the ability** to detect when *an administrator configurable positive integer within 3 to 20 unsuccessful authentication attempts occur related to remote login by a privileged user.*
FIA_AFL.1.2	When the defined number of unsuccessful authentication attempts has been [*met*], the TSF shall *prevent the remote privileged user from successfully authenticating until [an account unlock action] is taken by an Administrator or shall prevent the privileged user from successfully authenticating until a 24 hour period has elapsed.*

6.1.5.2 FIA_CMCS_EXT.1 Certificate Management over CMS (CMC) Server

FIA_CMCS_EXT.1.1	The TSF shall be able to accept and process CMC full requests and [*no other requests*].
FIA_CMCS_EXT.1.2	The TSF shall be able to generate CMC simple responses and [*no other*] that are consistent with the selected certificate profile and which are in accordance with RFC 5272 as updated by RFC 6402, meeting the compliance requirements for CMS server and certification authorities in accordance with RFC 5474 as updated by RFC 6402.
FIA_CMCS_EXT.1.3	The TSF shall require CMC transport over HTTPS for online CMC messages in accordance with RFC 5273 as updated by RFC 6402, where the HTTPS is established in accordance with FCS_HTTPS_EXT.1. For CMC requests containing certificate requests other than initial certificate requests authenticated using shared secrets in AuthenticatedData requests or in the Identity Proof Version 2 Control of SignedData requests, the TSF shall require HTTPS with client authentication, shall ensure the authenticating entity is the same as the entity signing the CMC request and any subject indicated in the requested certificate(s) are the same as the authenticating entity, or the authenticating entity is [no other entity].
FIA_CMCS_EXT.1.4	The TSF shall require CMC simple and full messages use cryptographic support in accordance with this profile. At a minimum the TSF shall ensure: • Signature generation and verification for SignedData are performed in accordance with FCS_COP.1(2) • Encryption for EnvelopedData is performed in accordance with FCS_COP.1(1) • PasswordRecipientInfo for EnvelopedData or AuthenticatedData is derived in accordance with FCS_COP.1(5) • hashAlgId in Identity Proof Version 2 control, keyGenAlgorithm in Pop Link • Witness Version 2 control, witnessAlgID in Encrypted POP and Decrypted POP controls, hashAlgorithm in Publish Trust Anchors control are in accordance with FCS_COP.1(3) • macAlgId in Identity Proof Version 2 control, macAlgorithm in POP Link Witness Version 2 Control, and the POPAlgID in Encrypted POP and Decrypted POP controls, are in accordance with FCS_COP.1(4) • DHPOP mechanisms shall be as specified in RFC 6955 with cryptographic support in accordance with this Protection Profile
FIA_CMCS_EXT.1.5	The TSF shall accept, process and export CMC messages under the control of local privileged user sessions for privileged users with CA Operations Staff, *no other* role.

6.1.5.3 FIA_CMCC_EXT.1 Certificate Management over CMS (CMC) Client

The TOE does not provide CMC Client functionality.

6.1.5.4 FIA_ENR_EXT.1 Certificate enrolment

FIA_ENR_EXT.1 is not implemented in this TOE.

6.1.5.5 FIA_ESTC_EXT.1 Enrolment over Secure Transport (EST) Client

The TOE does not implement EST.

6.1.5.6 FIA_ESTC_EXT.2 EST Client use of TLS-unique value

The TOE does not implement EST.

6.1.5.7 FIA_ESTS_EXT.1 Enrolment over Secure Transport (EST) Server

The TOE does not implement EST.

6.1.5.8 FIA_ESTS_EXT.2 Enrolment over Secure Transport (EST) Server

The TOE does not implement EST.

6.1.5.9 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1	The TSF shall provide the following password management capabilities for privileged passwords: • Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: *"!", "@", "#", "$", "%", "^", "&", "*", "(", ")".* • Minimum password length shall be settable by the Administrator, and support passwords of 14 characters or greater.

6.1.5.10 FIA_PSK_EXT.1 Pre-Shared Key Composition

Pre-shared key composition is not utilised by this TOE.

6.1.5.11 FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1	The TSF shall provide only ***obscured feedback and [no other information]*** to the **privileged** user while the authentication is in progress.

6.1.5.12 FIA_X509_EXT.1 Certificate Validation

FIA_X509_EXT.1.1	The TSF shall *validate* certificates in accordance with the following rules: • IETF RFC 5280 certificate validation and certificate path validation. • The certificate path must terminate with a certificate in the Trust Anchor Database. • The TSF shall validate a certificate path by ensuring the presence of them basicConstraints extension and that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met. • The TSF shall validate the revocation status of the certificate using CRL as specified in RFC 5280 and refined by RFC 8603. • The TSF shall validate the extendedKeyUsage (EKU) field according to the following rules: ○ Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-

	kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field. o Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the EKU field. o Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the EKU field. o Delegated OCSP signer's certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the EKU field. o Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.
FIA_X509_EXT.1.2	The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

6.1.5.13 FIA_X509_EXT.2 Certificate-Based Authentication

FIA_X509_EXT.2.1	The TSF shall *use* X.509v3 certificates as defined by RFC 5280 to support authentication for code signing for TOE updates, *TLS, HTTPS* and *no additional uses.*
FIA_X509_EXT.2.2	When the TSF cannot determine the current revocation status of a certificate, the TSF shall *not accept the certificate.*
FIA_X509_EXT.2.3	The TSF shall not establish a trusted communication channel if the peer certificate is deemed invalid.

6.1.5.14 FIA_UAU_EXT.1 Authentication Mechanism

FIA_UAU_EXT.1.1	The TSF shall *provide* a *password-based authentication mechanism*, [*X509 mutual authentication and API Key*] to perform privileged user authentication.

6.1.5.15 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1.1	The TSF shall allow the following actions prior to requiring a non-TOE entity to initiate the identification and authentication process: • Display the warning banner in accordance with FTA_TAB.1. • Obtain certificate status information. • [*Submit a certificate request*].
FIA_UIA_EXT.1.2	The TSF shall require each user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that user, including subscriber certificate renewal, subscriber revocation requests, privileged user access, *no other actions*.
FIA_UIA_EXT.1.3	For subscriber actions, the TSF shall verify that the DN of the certificate presented by the subscriber for authentication matches that of the certificate being affected by the subscriber's actions.

6.1.6 Security Management (FMT)

6.1.6.1 FMT_MOF.1(1) Management of Security Functions Behaviour (Administrator Functions)

FMT_MOF.1.1(1)	The ***TSF*** shall restrict the ability to **1. manage the TOE locally and remotely,** **2. configure the audit mechanism,** **3. configure and manage certificate profiles,** **4. modify revocation configuration,** **5. perform updates to the TOE,** **6. perform on-demand integrity tests;** **7. import and remove X.509v3 certificates into/from the Trust Anchor Database;** **8. import [all types of keys for the system]**, ***9. configure certificate revocation list function,*** ***10. configure OCSP function;*** ***11. disable deprecated algorithms,*** ***12. accept certificates whose validity cannot be determined;*** ***13. export PKCS#10 certificate request;*** ***14. import CA certificate;*** **15. *[manage the creation, deletion and modification of system users and tenancies*]]** to [Administrators].

6.1.6.2 FMT_MOF.1(2) Management of Security Functions Behaviour (CA/RA Functions)

FMT_MOF.1.1(2)	The ***TSF*** shall restrict the ability to **1. approve and execute the issuance of certificates,** **2. configure subscriber self-service request constraints,** ***3. configure automated certificate approval management,*** ***4. approve rulesets that govern the authorisations of AORs to manage particular certificates on behalf of an organisation,*** ***5. accept, process and export CMC messages,*** **6. *no other function*** to ***CA Operations Staff, RA Staff*.**

6.1.6.3 FMT_MOF.1(3) Management of Security Functions Behaviour (CA Operations Functions)

FMT_MOF.1.1(3)	The ***TSF*** shall restrict the ability to **1. approve certificate revocation,** ***2. perform archival and recovery,*** ***3. import a key share to support recovery of a CA signing key,*** ***4. approve rulesets that govern the authorisations of RAs to manage particular certificates on behalf of an organisation,*** ***5. export PKCS#10 certificate request,*** ***6. import CA certificate,*** **7. *no other function*** to ***CA Operations Staff*.**

6.1.6.4 FMT_MOF.1(4) Management of Security Functions Behaviour (Admin/Officer Functions)

FMT_MOF.1.1(4)	The ***TSF*** shall restrict the ability to **1. perform destruction of sensitive data when no longer needed,** ***2. participate as a second party for archival and recovery,*** ***3. import a key share to support recovery of a CA signing key,*** ***4. perform encrypted export of private or secret key or critical data*** to **Administrators, Auditor, CA Operations staff**.

6.1.6.5 FMT_MOF.1(5) Management of Security Functions Behaviour (Auditor Functions)

FMT_MOF.1.1(5)	The ***TSF*** shall restrict the ability to **1. delete entries from the audit trail,** **2. *search the audit trail,*** **3. *set or change the retention period parameter for audit records requiring extended retention,*** to [auditors].

6.1.6.6 FMT_MTD.1 Management of TSF Data

FMT_MTD.1.1	The TSF shall restrict the ability to **manage the TSF data** to [*privileged users].*

6.1.6.7 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1	The ***TSF*** shall be capable of performing the following management functions: *1. Ability to manage the TOE locally and remotely.* *2. Ability to perform updates to the TOE.* *3. Ability to perform archival and recovery.* *4. Ability to manage the audit mechanism.* *5. Ability to configure and manage certificate profiles.* *6. Ability to approve and execute the issuance of certificates.* *7. Ability to approve certificate revocation.* *8. Ability to modify revocation configuration.* *9. Ability to configure subscriber self-service request constraints.* *10. Ability to perform on-demand integrity tests.* *11. Ability to destroy sensitive user data when no longer needed.* *12. Ability to import and remove X.509v3 certificates into/from the Trust Anchor Database.* *13. Ability to configure the NPE ruleset.* *14. Ability to configure automated process used to approve the revocation of a certificate or information about the revocation of a certificate.* *15. Ability to approve rulesets that govern the authorisations of RAs or AORs to manage particular certificates on behalf of an organisation.* *16. Ability to modify the CRL configuration.* 17. *Ability to modify the OCSP configuration.* 18. *Ability to configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1.* 19. *Ability to configure the cryptographic functionality.* 20. *Ability to import private keys.* 21. *Ability to export TOE private keys (not for archival).* 22. *Ability to disable deprecated algorithms.* 23. *Ability to accept, process and export CMC messages.* 24. *No other capabilities.*

6.1.6.8 FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1	The TSF **and *no other component*** shall maintain the roles**:** • administrator, • auditor, • CA Operations Staff, • **RA Staff.**

FMT_SMR.2.2	The TSF **and *no other component*** shall be able to associate users with roles.
FMT_SMR.2.3	The TSF **and *no other component*** shall ensure that the conditions: • No identity is authorised to assume both an Auditor role and any of the other roles in FMT_SMR.2.1, and • No identity is authorised to assume both a CA Operations Staff role and any of the other roles in FMT_SMR.2.1, are satisfied.

6.1.7 Protection of the TSF (FPT)

6.1.7.1 FPT_APW_EXT.1 Protection of Privileged User Passwords

FPT_APW_EXT.1.1	The TSF shall store passwords in non-plaintext form.

6.1.7.2 FPT_FLS.1 Failure with Preservation of Secure State

FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: [*audit log exhaustion, loss of access to the HSM].*

6.1.7.3 FPT_ITT.1 Basic Internal TSF Data Transfer Protection

FPT_ITT.1.1	The TSF shall protect TSF data from disclosure, modification when it is transmitted between separate parts of the TOE **through the use of *TLS, TLS/HTTPS.***

6.1.7.4 FPT_KST_EXT.1 No Plaintext Key Export

FPT_KST_EXT.1.1	The TSF and *Operational Environment* shall prevent the plaintext export of: [all cryptographic keys used by the TSF]**.**

6.1.7.5 FPT_KST_EXT.2 TSF Key Protection

FPT_KST_EXT.2.1	The TSF and *Operational Environment* shall prevent unauthorised use of all TSF private and secret keys.

6.1.7.6 FPT_NPE_EXT.1 NPE Constraints

FPT_NPE_EXT.1.1	The TSF shall enforce an Administrator-configurable ruleset that specifies authorisations to submit NPE certificate requests.
FPT_NPE_EXT.1.2	The TSF shall require the CA Operations Staff to register any RA, and shall require a CA Operations Staff or authorised RA to register any AORs, and associate each AOR with an organisation or set of devices prior to that AOR making requests on behalf of an assigned organisation or devices.

6.1.7.7 FPT_RCV.1 Manual Trusted Recovery

FPT_RCV.1.1	After *[audit log storage exhaustion or communication failure with the HSM*] the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

6.1.7.8 FPT_SKP_EXT.1 Protection of Keys

FPT_SKP_EXT.1.1	The TSF shall *interface with the Operational Environment to implement* the ability to prevent reading of all pre-shared keys, private, and secret keys (e.g., KEKs, DEKs, session keys).

6.1.7.9 FPT_STM.1 Reliable Time Stamps

FPT_STM.1.1	The TSF shall ***interface with the Operational Environment to provide*** reliable time stamps.

6.1.7.10 FPT_TST_EXT.1 TOE Integrity Test

The TOE does not implement integrity testing on the TOE.

6.1.7.11 FPT_TST_EXT.2 Integrity Test

The TOE does not implement integrity testing.

6.1.7.12 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1	The TSF shall *interface with the Operational Environment to implement* the ability to check for updates and patches to the TOE.
FPT_TUD_EXT.1.2	The TSF shall *interface with the Operational Environment to implement* the ability to provide Administrators the ability to initiate updates to TOE firmware/software.
FPT_TUD_EXT.1.3	The TSF shall *interface with the Operational Environment to implement* the ability to verify firmware/software updates to the TOE using a digital signature prior to installing those updates.
FPT_TUD_EXT.1.4	The TSF shall *interface with the Operational Environment to implement* the ability to verify the digital signature whenever the software or firmware is externally loaded into the TOE and if verification fails, the TSF shall *[take no action].*

6.1.8 TOE Access (FTA)

6.1.8.1 FTA_SSL.3 TSF-Initiated Termination

FTA_SSL.3.1	The TSF shall terminate **a remote** interactive session after a *[default 15 minutes of session inactivity].*

6.1.8.2 FTA_SSL.4 User-Initiated Termination

FTA_SSL.4.1	The TSF shall ***implement the ability to*** allow **privileged** user-initiated termination of the **privileged** user's own interactive session.

6.1.8.3 FTA_TAB.1 Default TOE Access Banners

FTA_TAB.1.1	Before establishing **a privileged** user session the TSF shall display **an Administrator-configured** advisory **notice and consent** warning message regarding unauthorised use of the TOE.

6.1.9 Trusted Path/Channels (FTP)

6.1.9.1 FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1	The TSF shall **use *HTTPS and TLS* to** provide a **trusted** communication channel between itself and **authorised external network based IT entities supporting the following capabilities: *external cryptographic module, directory services, RA, [database and CRL Publication points]*** that are logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from modification or disclosure.
FTP_ITC.1.2	The TSF shall permit *the TSF and the authorised IT entities* to initiate communication via the trusted channel.
FTP_ITC.1.3	The TSF shall initiate communication via the trusted channel for *external cryptographic module, directory services, RA, database and CRL Publication points.*

6.1.9.2 FTP_TRP.1 Trusted Path

FTP_TRP.1.1	The TSF shall **use *HTTPS, TLS* to** provide **a trusted** communication path between itself and **remote subscribers and privileged users** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [modification and disclosure].
FTP_TRP.1.2	The TSF shall permit **remote subscribers** and privileged users to initiate communication via the trusted path.
FTP_TRP.1.3	The TSF shall require the use of the trusted path for *[initial subscriber and privileged user authentication and all remote administration actions].*

6.2 Security Assurance Requirements

6.2.1 Class ADV: Development

6.2.1.1 ADV_FSP.1 Basic Functional Specification

	Developer action elements:
ADV_FSP.1.1D	The developer shall provide a functional specification.
ADV_FSP.1.2D	The developer shall provide a tracing from the functional specification to the SFRs.
	Content and presentation elements:
ADV_FSP.1.1C	The functional specification shall describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI.
ADV_FSP.1.2C	The functional specification shall identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI.
ADV_FSP.1.3C	The functional specification shall provide rationale for the implicit categorisation of interfaces as SFR-non-interfering.
ADV_FSP.1.4C	The tracing shall demonstrate that the SFRs trace to TSFIs in the functional specification.
	Evaluator action elements:
ADV_ FSP.1.1E	The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.
ADV_ FSP.1.2E	The evaluator shall determine that the functional specification is an accurate and complete instantiation of the SFRs.

6.2.2 Class AGD: Guidance Documentation

6.2.2.1 AGD_OPE.1 Operational User Guidance

	Developer action elements:
AGD_OPE.1.1D	The developer shall provide operational user guidance.
	Content and presentation elements:
AGD_OPE.1.1C	The operational user guidance shall describe, for each privileged user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.
AGD_OPE.1.2C	The operational user guidance shall describe, for each privileged user role, how to use the available interfaces provided by the TOE in a secure manner.
AGD_OPE.1.3C	The operational user guidance shall describe, for each privileged user role, the available functions and interfaces, in particular all security parameters under the control of the privileged user, indicating secure value as appropriate.
AGD_OPE.1.4C	The operational user guidance shall, for each privileged user role, clearly present each type of security-relevant event relative to the privileged user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.
AGD_OPE.1.5C	The operational user guidance shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

AGD_OPE.1.6C	The operational user guidance shall, for each privileged user role, describe the security measures to be followed in order to fulfill the security objectives for the Operational Environment as described in the ST.
AGD_OPE.1.7C	The operational user guidance shall be clear and reasonable.
	Evaluator action elements:
AGD_OPE.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

6.2.2.2 AGD_PRE.1 Preparative Procedures

	Developer action elements:
AGD_PRE.1.1D	The developer shall provide the TOE, including its preparative procedures.
	Content and presentation elements:
AGD_PRE.1.1C	The preparative procedures shall describe all the steps necessary for secure acceptance of the delivered TOE in accordance with the developer's delivery procedures.
AGD_PRE.1.2C	The preparative procedures shall describe all the steps necessary for secure installation of the TOE and for the secure preparation of the Operational Environment in accordance with the security objectives for the Operational Environment as described in the ST.
	Evaluator action elements:
AGD_PRE.1.1E	The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.
AGD_PRE.1.2E	The evaluator *shall apply* the preparative procedures to confirm that the TOE can be prepared securely for operation.

6.2.3 Class ALC: Life-Cycle Support

6.2.3.1 ALC_CMC.1 Labelling of the TOE

	Developer action elements:
ALC_CMC.1.1D	The developer shall provide the TOE and a reference for the TOE.
	Content and presentation elements:
ALC_CMC.1.1C	The TOE shall be labelled with its unique reference.
	Evaluator action elements:
ALC_CMC.1.1E	The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

6.2.3.2 ALC_CMS.1 TOE CM Coverage

	Developer action elements:
ALC_CMS.2.1D	The developer shall provide a configuration list for the TOE.
	Content and presentation elements:
ALC_CMS.2.1C	The configuration list shall include the following: the TOE itself; and the evaluation evidence required by the SARs.
ALC_CMS.2.2C	The configuration list shall uniquely identify the configuration items.
	Evaluator action elements:

ALC_CMS.2.1E	The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.

6.2.4 Class ASE: Security Target Evaluation

Security Target evaluation is per the definition in the Common Evaluation Methodology (CEM).

6.2.5 Class ATE: Tests

6.2.5.1 ATE_IND.1 Independent Testing – Conformance

	Developer action elements:
ATE_IND.1.1D	The developer shall provide the TOE for testing.
	Content and presentation elements:
ATE_IND.1.1C	The TOE shall be suitable for testing.
	Evaluator action elements:
ATE_IND.1.1E	The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.
ATE_IND.1.2E	The evaluator *shall test* a subset of the TSF to confirm that the TSF operates as specified.

6.2.6 Class AVA: Vulnerability Analysis.

6.2.6.1 AVA_VAN.1 Vulnerability Survey

	Developer action elements:
AVA_VAN.1.1D	The developer shall provide the TOE for testing.
	Content and presentation elements:
AVA_VAN.1.1C	The TOE shall be suitable for testing.
	Evaluator action elements:
AVA_VAN.1.1E	The evaluator *shall confirm* that the information provided meets all requirements for content and presentation of evidence.
AVA_VAN.1.2E	The evaluator *shall perform* a search of public domain sources to identify potential vulnerabilities in the TOE.
AVA_VAN.1.3E	The evaluator *shall conduct* penetration testing, based on the identified potential vulnerabilities, to determine that the TOE is resistant to attacks performed by an attacker possessing Basic attack potential.

7 TOE Summary Specification

7.1 FAU: Audit

7.1.1 FAU_ADP_EXT.1 Audit Dependencies

Auditing of the TSF is performed by an audit microservice - usAudit - that forwards all audit events for the system to the central database. The auditable events are listed in Table 11: Security Functional Requirements and Auditable Events.

All audit events for the system are forwarded to the central database in the Operational Environment or to journalctl. Audit Records are configured to conform to a protobuf schema that includes timestamps and user identity. As per the Admin Guide all auditable events are retained in the central database until the Operational Environment disk drive is full, and OE stored logs are rotated as configuration in journalctl allows.

If the audit microservice fails or stops working for any reason, all activities will cease on the Jellyfish client, i.e. Jellyfish client will not issue or revoke certificates.

Jellyfish allows for audit logs within the system or using an external audit server. Audit logs are accessed based on the roles assigned within the system, i.e. administrator can access and view the audit logs.

All events outlined in Table 11 above, are logged and audited.

An example of the protobuf definition which details the type of content recorded has been provided in Annex A.

7.1.2 FAU_GCR_EXT.1 Generation of Certificate Repository

Certificates and CRLs in Jellyfish CA are stored in the local certificate repository within the central database in the OE. The repository is accessed via the usDatabase microservice using internal ports.

7.1.3 FAU_GEN.1 Audit Data Generation

Jellyfish CA implements an audit microservice - usAudit - that forwards all audit events for the system to the central database.

All auditing event types and fields are described in the FAU_GEN.1.1 SFR and Table 11 per the requirements of the Protection Profile

Jellyfish CA uses an Operational Environment NTP source in order to ensure consistent timestamping of events.

Events are stored in the database and use with the schema as identified in FAU_ADP_EXT.1

7.1.4 FAU_GEN.2 User Identity Association

As stated in paragraph 7.1.1, audit Records are configured to conform to a protobuf schema that includes timestamps and the user identity which is audited.

All users within Jellyfish CA are uniquely identifiable by email address and audit events are associated with user identification as defined in the application.

7.1.5 FAU_SAR.1 Audit Review

Audit administrators may use the implemented Jellyfish CA search functionality in the web GUI to query the audit logs in the form of a table of events.

Jellyfish CA implements a search functionality through a web GUI that allows queries for audit administrators and allows queries for audit administrators in the form of a table of events.

Audit records are returned as a human readable table of events. If required, query results can also be downloaded in various formats.

7.1.6 FAU_SAR.3 Selectable Audit Review

As identified in paragraph 7.1.1 above, auditors can use the GUI search field to review audit logs. The search interface allows the use of free text to query and filter on all objects in the log.

7.1.7 FAU_SCR_EXT.1 Certificate Repository Review

The Jellyfish CA provides a GUI search interface allows an auditor the ability to search for certificates in the certificate repository using the free text and filtering capabilities. The free text search allows the for the certificate repository to be search against all certificate fields. Related objects, such as users and devices, can also be searched through the GUI search interface.

Audit records are returned as a human readable table of events. If required, query results can also be downloaded in various formats.

7.1.8 FAU_SEL.1 Selective Audit

Select Audit is not a feature of Jellyfish CA. Audit is full-take and therefore selective audit does not occur in the system.

7.1.9 FAU_STG.1(1) Protected Audit Trail Storage

All access to the usAudit microservice is limited to authenticated system users with the Auditor role only. The ability to manipulate and edit any audit logs is not possible from the TOE and can only occur from the Operational Environment.

Further, the database is configured to allow only connections from the Operational Environment.

7.1.10 FAU_STG.1(2) Protected Audit Trail Storage (Archive Data)

Audit events can be configured to be forwarded to an external SIEM, allowing for retention and archiving independent of the TOE. Where audit logs are preserved on the TOE, a modification or deletion of the audit events can only occur through authorised access to the operating system and therefore independent of the TOE.

Audit logs marked as extended retention are retained for the life of the system within the central database and only allow the JF database user to control the tables in the database.

Further, audit logs can be configured to forward to a centralised SIEM but this functionality does not form part of the trusted security functionality of the TOE.

7.1.11 FAU_STG.4 Prevention of Audit Data Loss

On exhaustion of resources in the audit log database, the system will halt in such a way that further interaction cannot occur until the audit database is manually cleared or rolled over in the operating system.

7.1.12 FAU_STG_EXT.1 External Audit Trail Storage

All transactions to the audit database are managed by the usDatabase microservice and all audit data is stored on the database in the OE. Forwarding of audit events to external SIEMs is configurable by the administrator and protected by HTTPS to ensure confidentiality and integrity of the audit events.

7.1.13 FAU_STG_EXT.2 Audit Data Retention

All audit data with retention requirements are retained on the system until manual rollover from the Postgresql database occurs. Data without retention requirements are maintained in the OE until that system rolls over new data. In accordance with FAU_STG.4, the system will halt until the database is manually cleared by an operator.

If configured within the system build, all audit data is onforwarded to a SIEM in the Operational Environment that provides audit retention capabilities for the system.

7.2 FCO: Communication

7.2.1 FCO_NRO_EXT.2 Certificate-Based Proof of Origin

All certificates, certificate revocation lists and OCSP responses issued by CA are digitally signed using X509 compliant certificates with the CA attribute set to True and in accordance with RFC 5280 and as described in FCS_COP.1(2).

Proof of origin for CMC certificate requests and revocations is provided through a signed client info attribute in the CSR and PKCS7. Further, proof of possession is implemented using mechanisms described in FIA_CMCS_EXT.1.

Manual revocation of certificates can only be performed by authenticated users with appropriate permissions within the system in accordance with the Jellyfish Administrator Guide.

7.2.2 FCO_NRR_EXT.2 Certificate-Based Proof of Receipt

All communications are sent over TLS as described in section 7.3.20.

Session identity proof is provided by a session cookie signed with an ephemeral symmetric key, which is rotated every hour.

CMS (Cryptographic Message Syntax) messages are sent in PKCS7 format and signed using the SHA256 digest algorithm. The following CMS messages are encrypted using AES256GCM:

- CMC Full Request when submitted offline
- CMC Full Response when issued offline

Certificate signing requests (CSRs) are signed with the client's private key. These CSRs can be generated in two ways:

- By an external system when submitting a CSR to the Jellyfish portal.
- In the web browser through the Jellyfish JavaScript application when creating the CSR in the portal.

CMS certificate request messages are signed with the CSR's private key. CMC responses sent from the Jellyfish system are signed by the private key of the issuing CA.

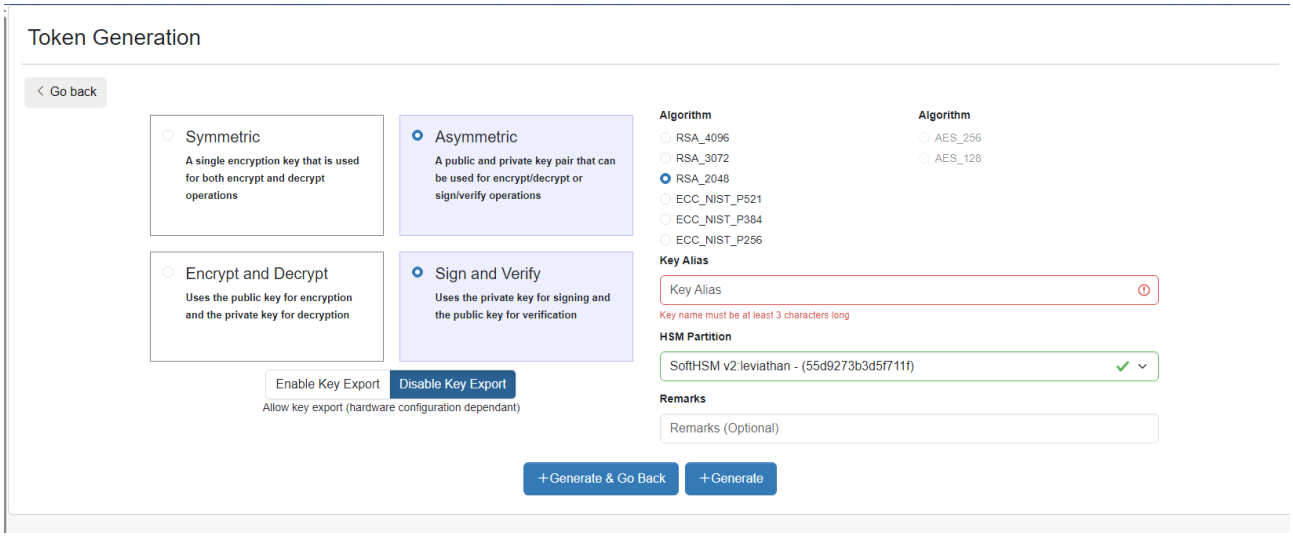
Proof of origin for CMC certificate requests is provided through a signed client info attribute in the CSR and PKCS7. Proof of origin for CMC certificate revocation requests is provided through a signed client info attribute in the PKCS7.

To ensure receipt of certificate requests and revocation requests, the system verifies the digital signatures on all received requests. If the signature is not valid, the request is rejected.

7.3 FCS: Cryptographic Support

7.3.1 General Concepts

Key generation within the TOE occurs through interaction with Hardware Security Modules (HSM) configured to interact with the usCrypto microservice. An example below is the token generation option presented to an authorised user in the application.



Cryptographic functionality implemented for HTTPs and mTLS for web connection and microservice communication respectively utilises the standard GoLang libraries complied with the boringcrypto flag in order to utilise FIPS 140-3 compliant libraries validated against CAVP, subsequently issued certificate A1657: https://csrc.nist.gov/projects/Cryptographic-Algorithm-Validation-Program/details?source=A&number=1657

7.3.2 FCS_CDP_EXT.1 Cryptographic Dependencies

Hardware Security Module

Jellyfish CA uses application interfaces to a dedicated HSM in the Operational Environment to invoke cryptographic functionality on the system.

The following HSMs are configurable within Jellyfish CA:

- AWS KMS
- Azure KeyVault
- Salesforce
- Google CSE
- Other HSMs with a PKCS#11 Interface

The HSM provider is configurable by the user and invoked by the usCrypto microservice in Jellyfish CA.

Consul and usWebserver

Consul and usWebserver use Go cryptographic libraries compiled with the boringcrypto experimental flag to ensure dependencies utilise previously validated FIPS 140-3 libraries in their functions.

7.3.3 FCS_CKM.1 Cryptographic Key Generation

Hardware Security Module

Jellyfish CA interfaces with a HSM to generate the following keys for use in certificates:

- RSA 4096
- RSA 3072
- RSA 2048
- ECC NIST P-521
- ECC NIST P-384
- ECC NIST P-256

Consul and usWebserver

Consul and usWebserver are able to generate keys as defined in the GoLang Crypto package https://pkg.go.dev/crypto

7.3.4 FCS_CKM.2 Cryptographic Key Establishment

HSM

All cryptographic key establishment activities are invoked from the HSM in the Operational Environment.

Consul and usWebserver

Cryptographic services are provided by the GoLang Crypto package to implement following HTTPs connectivity as a sender:

TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.

7.3.5 FCS_CKM_EXT.1(2) Key Generation Key Encryption Keys

HSM

Jellyfish CA uses HSMs in the Operational Environment to generate Key Encryption Keys.

Jellyfish CA restricts key size in configuration to 256 bits.

7.3.6 FCS_CKM_EXT.1(3) Key Generation for Key Encryption Keys (TOE Key Archival)

Key Encryption Keys are generated in the Operational Environment's HSM. System users are advised to procure and use FIPS 140-3 HSMs to ensure the effectiveness of random bit generation services.

7.3.7 FCS_CKM_EXT.1(4) Generation of Key Shares

Key shares are generated in the Operational Environment's HSM. System users are advised to procure and use FIPS 140-3 HSMs to ensure the effectiveness of random bit generation services.

7.3.8 FCS_CKM_EXT.4 Cryptographic Key Destruction

All user and system private keys utilised by the system are stored and managed in the Operational Environment's HSM. No keys are provided to the TOE as part of normal system operation.

Keys are destroyed in the HSM by invoking the appropriate mechanisms through the TOE.

7.3.9 FCS_CKM_EXT.5 Public Key Integrity

All public keys are stored in the Operational Environment's HSM and invoked by the TOE when and as required.

Integrity is preserved through digital signatures and checked through validation in the system.

7.3.10 FCS_CKM_EXT.6 TOE Key Archival

Key archival is implemented through interface to a HSM in the Operational Environment with parameters as identified in the security functional requirement.

7.3.11 FCS_CKM_EXT.7 Key Generation for KEKs

Key generation is implemented through interface to a HSM in the Operational Environment.

HSMs available for selection in Jellyfish CA are FIPS 140-2 certified where possible.

All REKs are generated and stored within the HSM to prevent access.

7.3.12 FCS_CKM_EXT.8 Key Hierarchy Entropy

HSMs are used to enforce key hierarchy entropy and are independent of the system.

7.3.13 FCS_COP.1(1) Cryptographic Operation (AES Encryption/Decryption)

Hardware Security Modules

HSMs are used to provide AES encryption/decryption and implement the following algorithms:

- AES-CBC (as defined in NIST SP 800-38A) mode,
- AES-GCM (as defined in NIST SP 800-38D) mode,

Cryptographic key size is 256-bit or greater.

Consul and usWebserver

Consul and usWebserver are built using Go and use the internal cryptography libraries to provide:

- AES-CBC (as defined in NIST SP 800-38A) mode,
- AES-GCM (as defined in NIST SP 800-38D) mode,

Cryptographic key size is 256-bit or greater.

7.3.14 FCS_COP.1(2) Cryptographic Operation (Cryptographic Signature)

Hardware Security Modules

Interfaces to the HSM are used to invoke signature services.

The following algorithms are used:

- RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater that meets FIPS-PUB 186-4, "Digital Signature Standard",
- Elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater that meets FIPS PUB 186-4, "Digital Signature Standard" with "NIST curves" P-256, P-384 and P-521, no other curves] (as defined in FIPS PUB 186-4, "Digital Signature Standard").

Hardware Security Modules

ACVP validated cryptographic libraries are used in Go to implement cryptographic signatures.

The following algorithms are used:

- RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater that meets FIPS-PUB 186-4, "Digital Signature Standard",
- Elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater that meets FIPS PUB 186-4, "Digital Signature Standard" with "NIST curves" P-256, P-384 and P-521, no other curves] (as defined in FIPS PUB 186-4, "Digital Signature Standard").

7.3.15 FCS_COP.1(3) Cryptographic Operation (Cryptographic Hashing)

Interfaces to the HSM are used to invoke signature services.

The following algorithms are used: SHA-256, SHA-384, SHA-512 and message digest sizes 256, 384, 512 bits.

Users are directed to use HSMs that have been validated per the NIST CMVP where possible.

Consul and usWebserver

Interfaces to ACVP compliant libraries in GoLang are used to invoke signature services.

The following algorithms are used: SHA-256, SHA-384, SHA-512 and message digest sizes 256, 384, 512 bits.

7.3.16 FCS_COP.1(4) Cryptographic Operation (Keyed-Hash Message Authentication)

Hardware Security Modules

Interfaces to the HSM are used to invoke keyed-hash message authentication services.

The following algorithms are used: HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, key size 256 bits, and message digest sizes 256, 384, 512 bits.

Users are directed to use HSMs that have been validated per the NIST CMVP where possible.

Hardware Security Modules

Interfaces to ACVP compliant libraries in GoLang are used to invoke keyed-hash message authentication services.

The following algorithms are used: HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, key size 256 bits, and message digest sizes 256, 384, 512 bits.

7.3.17 FCS_HTTPS_EXT.1 HTTPS Protocol

The Jellyfish CA UI provides connectivity to administrative users via API using HTTPS configured in line with RFC 2818 from the standard Go libraries using ACVP assessed primitives.

The Jellyfish CA UI is protected by TLS 1.3.

7.3.18 FCS_RBG_EXT.1 Cryptographic Random Bit Generation

Hardware Security Modules

All random bit generation functions are invoked from the HSM and utilise non-deterministic entropy sources.

A random noise source is invoked in an Operational Environment-based HSM to generate at least 128 bits of entropy in accordance with NIST SP 800-57. HSMs are chosen by the operator to ensure compliance with NIST SP 800-57.

Consul and usWebserver
Consul and usWebserver are compiled using the boringcrypto flag that replaces internal Go libraries with FIPS 140-3 compliant libraries that invoke CTR_DBRG using AES-256.

7.3.19 FCS_STG_EXT.1 Cryptographic Key Storage

All persistent private and secret keys utilised by the Jellyfish CA are stored within a hardware cryptographic module ensuring secure storage when not in use. uses two certificates distinct types of certificates.

The table below lists the secret and private keys and their purpose, protection and location

Key	Purpose	Storage	Protection
Certificate Authority Issuer (usCA-Leviathan)	Signing certificates and revocation lists	PKCS#11 Cryptographic Module	Protected by the PKCS#11 Cryptographic Module
Registration Authority Enrollment Agent Signer (usPKI)	Signing CMC responses	Operating system key store	Protected by operating system permissions and access control lists
Validation Authority OCSP Signer (CogVA)	Signing OCSP responses	Operating system key store	Protected by operating system permissions and access control lists
TLS HTTP Authentication Certificate	Server and Client authentication (MTLS)	Operating system key store	Protected by operating system permissions and access control lists

7.3.20 FCS_TLSS_EXT.1 TLS Server Protocol

Jellyfish CA utilises the underlying RFC 3268, 4492, 5246, 5288, 5299 and 5289 compliant infrastructure withing CogVa to provide secure ciphers with minimum key sizes.

The following ciphersuites are configured in the Common Criteria build when using an RSA certificate:

TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,

TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.

The Jellyfish build guide provides guidance on securely configuring communication channels between services; additionally,-Jellyfish Common Criteria build does not natively support TLS 1.1 due to limitations of enforced ciphersuites. Ciphersuites listed above do not support TLSv1.1 – minimum supported version is TLSv1.2.

No key sizes are explicitly supported. RSA key sizes greater than 8192 are NOT supported.

7.4 FDP: User Data Protection

7.4.1 FDP_CER_EXT.1 Certificate Profiles

The Jellyfish CA implements certificate profiles directly through the text based json files in which they are stored . Once these templates are created, they can be altered by an Administrator via the Manage Licensed CSR Rules screen (https://web.jellyfish.server/pki/templates/manage/licensed).

When a certificate is requested to be issued by the CA, the CA verifies the certificate request against the requested Certificate Profile to ensure that only certificates consistent with configured profiles are issued.

The configurable Certificate Profiles will provide the ability to specify the attributes contained within RFC 5280 through CA microservices.

The Jellyfish CA is able to generate 20 bits of random data for use in issued certificate through invocation of HSM interfaces that is included in the serialNumber field. The random values are generated in accordance with NIST SP 800-90.

All certificates are generated to meet RFC 5280. usPKI validates certificates have been formatted to meet RFC 5280.

20 bits of random data for the serialNumber field is provided by invoking an interface to the HSM.

7.4.2 FDP_CER_EXT.2 Certificate Request Matching

When issuing certificates through the web application interface, the Jellyfish CA maintains a linkage between the certificate request and the issued certificate by database lookups. Certificate objects maintain a foreign key link to the Certificate Signing Request (CSR) that created it.

7.4.3 FDP_CER_EXT.3 Certificate Issuance Approval

Jellyfish CA provides the ability to configure approvals for requested certificates. This approval can be either automated rules based, or via the Jellyfish RA user interface by an authorised Registration Officer/CA Operator role holder.

Instructions for issuance and approval is included in the user guide.

7.4.4 FDP_CRL_EXT.1 Certificate Revocation List Validation

Generation of CRLs is performed by the CA microservice.

Fields included are:

- Version
- Issuer
- CA Version
- Signature Algorithm
- Signature Hash
- Authority Key Identifier
- CRL number
- Effective Date
- Next Update Date

- Revocation list
  - Certificate serial number
  - Revocation date-time
  - Revocation reason code

Generation of the CRL can be invoked from the Manage CA section of jellyfish.

7.4.5 FDP_CSI_EXT.1 Certificate Status Information

The Jellyfish CA supports the configuration for multiple users required to perform certificate revocations using the web interface. Both RA staff and CA operators can perform certificate revocations. Approval of certificate status changes is enforced through role-based access control as provided by the system. CA operators can approve status changes as needed.

Certificates are revoked using a Certificate Revocation List (CRL) in accordance with RFC 5280. CRLs can be requested from the by VA microservices and CA operator users. The CRLs are signed by the CA's private key stored in a Hardware Security Module (HSM).

Certificates can be revoked by submitting a certificate revocation request through one of three methods:

- CMC simple request containing a CMC revocation request
- Jellyfish API revocation request
- Direct revocation request to through the Command Line Interface (CLI)

The VA component CogVA accepts OCSP requests and returns OCSP responses in accordance with RFC 6960. OCSP responses are signed by an OCSP signing certificate issued by the CA. The VA component receives a new CRL to base its OCSP responses on whenever any certificate is revoked, and every 24 hours.

Once a certificate is revoked, it cannot be "unrevoked".

7.4.6 FDP_ITT.1 Basic Internal Transfer Protection

Jellyfish CA microservices are managed in Consul and enforces TLS for communication pathways when the TOE is used in a high-availability cluster.

7.4.7 FDP_OCSPG_EXT.1 OCSP Basic Response Generation

OCSP responses are provided by the VA microservice and are accessible from http://ocsp.<FQDN> by default.

The OCSP subdomain is configurable and preserved in the AIA field on the certificate. OCSP requests and responses align to RFC6960.

The OCSP Request contains the following data:

- Certificate serial number
- Subject name hash of the CA certificate
- Public key hash of the CA certificate
- Hash algorithm (usually SHA1, requesting software dependent, outside of cogito configuration controls)

The OCSP Response contains the following fields at a minimum:

- version – default, v1.
- signatureAlgorithm: Contains the object identifier (OID) for a digital signature algorithm in accordance with FCS_COP.1(2). This defaults to SHA1(industry standard) but can be configured to SHA256.
- thisUpdate: Is the time the response was generated, backdated to the start of the hour
- producedAt: Is the time the response was generated, backdated to the start of the minute.
- nextUpdate: Is always later than “thisUpdate”. It is set based on the CRL expiry or the configured max OCSP response validity period (default: 250 hours), whichever expires sooner.

7.4.8 FDP_RIP.1 Subset Residual Information Protection

All sensitive information related to keying material and keys is managed through the HSM in the Operational Environment. Keyed material is never loaded by and keyed material does not exist in any buffer. Keyed material ONLY exists in the HSM and never leaves the HSM. When keyed material is required for certificate, cmc, or crl signing, the data to be signed is sent to the HSM, a signature is generated within the HSM, and the signed material is returned. Buffers are not zeroized as key material does not exist in any buffer ever.

The steps for signing are as follows:

- Certificate request is submitted to the RA
- The RA verified the request meets the data and restriction protection requirements
- The RA submits the request to the CA
- The CA directs the certificate request to the HSM containing the CA’s certificate and private key
- The CA submits the signing request to the HSM containing the key
- The HSM returns the signed material
- The CA records the certificate
- The CA returns the signed certificate to the requestor

7.4.9 FDP_STG_EXT.1 Public Key Protection

All public keys and certificates are restricted to RBAC controlled access only access to the database in the operating system as part of the Operational Environment. Certificate Authority certificates and public keys are stored in the HSM. Access to the HSM is granted through HSM vendor specific authentication methods, e.g. through the use of a pin or smartcard and enforced by the usAuthentication microservice.

7.5 FIA: Identification and Authentication

7.5.1 FIA_AFL.1 Authentication Hardening Failure

Jellyfish CA tracks successful and unsuccessful logins through the central database, with logic controlled by usAuthentication.

The Cogito application locks out users after 3 attempts or as configured by the Administrator and can only be reactivated:

- By an administrator
- After 24 hours or as configured by the Administrator

7.5.2 FIA_PMG_EXT.1 Password Management

Default password length is 14 characters with a combination of at least one of: upper character, lower character, numbers and special character. This is fully configurable by the Administrator in the tenancy configuration section.

Implementation of multi-factor authentication is also available to enhance user security.

7.5.3 FIA_UAU.7 Protected Authentication Feedback

Jellyfish provides obscured feedback, in that it does not echo or display the characters of an entered password or pin in the portals. When an authentication attempt fails, no indication of the exact failure is provided. Enumeration/timing attack protection is implemented for user logon via the GUI.

7.5.4 FIA_CMCS_EXT.1 Certificate Management over CMS (CMC) Server

The Jellyfish CA server handles certificate management using the HSM for key generation and cryptographic operations, supporting RSA and ECC algorithms. Certificate requests can be validated, issued, and revoked, with all actions logged via the usAudit microservice. The server uses cryptographic operations, AES and RSA for issuing certificates. Access controls are in place to ensure that only authorised roles can manage certificate-related functions.

Jellyfish CA is able to process simple and full CMC requests as well as generate simple and full CMC responses. Prior to accepting requests, a user is required to authenticate and obtain a JWT using API keys per FIA_UAU_EXT.1 for use in making requests to the CMC HTTPS service, thus preserving identity and authorisation for the servuce.

7.5.5 FIA_UAU_EXT.1 Authentication Mechanism

Authentication via the Jellyfish RA can be achieved through:

- Username and password
- Application Programming Interface (API) keys submitted in HTTP header
- Passkey (FIDO)
- Certificate based (smart card and server).

7.5.6 FIA_UIA_EXT.1User Identification and Authentication

The Jellyfish CA allows an anonymous to performed certain activities without authentication to the system through an anonymous interface. These functions include submitting a certificate request. The anonymous portal can be configured to be available or disabled depending on the environment in which it is installed. All other activity within the interface requires authentication through HTTPS to perform authentication. All

privileged user actions require successfully authenticating to the TOE using TLS/HTTPS with client authentication using certificates.

7.5.7 FIA_X509_EXT.1 Certificate Validation

The usPKI microservice is used to validate certificates in line with the X.509 certificate RFC 5280. Further, all CA certificates must have the basicConstraints extension present and CA flag set to TRUE to be recognised as a CA certificate.

7.5.8 FIA_X509_EXT.2 Certificate-Based Authentication

Certificates are the primary authentication mechanism to authenticate users accessing the GUI via HTTPS. Only X.509 v3 certificates are generated in the system.
Certificates are parsed by Jellyfish CA and access requests are rejected if the certificate is revoked, expired or invalid.

7.6 FMT: Security Management

Cogito allows users to define and edit roles within the system.
As part of the TOE, the following Cogito provided JSON profiles are required to be loaded into the system:

- CC-Administrators.json
- CC-Auditor.json
- CC-CA-Operations.json
- CC-RA-Operator.json
- CC-Registration Officer.json

Roles can only be edited by users with access to the ‘Edit Roles’ permissions. This permissions is not included on any standard roles, and is only granted to cogito operators maintaining the system.
Users cannot auto-elevate permissions by adding or removing roles from themselves.

7.6.1 FMT_MOF.1(1) Management of Security Functions Behaviour (Administrator Functions)

The TOE restricts the following tasks, performed through the Administrator role.

The Administrator can do the following in regards to PKI:

- Review audit logs
- Manage and consume cryptographic keys for arbitrary encryption/signing operations
- View Certificate, authority, and template reporting information
- Upload manage and review existing certificates

The Administrator can NOT do the following in regards to PKI:

- Issue certificates
- Revoke certificates
- Request certificates be issues
- Approve certificate issuance requests
- Manage certificate authority and template access
- Manage certificate revocation lists
- Manage OCSP responder configurations

The following activities are limited to the administrator function:

- No actions are limited to administrator role, the registration officer and audit role are subsets of the admin role and cover much of the same permissions.

7.6.2 FMT_MOF.1(2) Management of Security Functions Behaviour (CA/RA Functions)

The TOE restricts the following tasks, performed through the CA/RA role.

The CA/RA Operator can do the following in regards to PKI:
- Manage certificate infrastructure configuration:
  - Domain whitelisting
  - CSR rules whitelisting
  - Certificate KU/EKU restriction
  - Certificate subject and san restriction
- Review pending/failed revocations
- Manage certificate requests and approvals
- Manage user access to certificate issuance
- Manage certificate templates
- Manage revocation lists and OCSP responder configurations
- Collect issued certificates

The CA/RA Operator can NOT do the following in regards to PKI:
- Issue certificates
- Revoke certificates
- Request certificate be issued
- Audit
- Reporting

The following activities are limited to the RA operator
- Manage revocation and OCSP configuration
- Manage whitelist/restriction rules

7.6.3 FMT_MOF.1(3) Management of Security Functions Behaviour (CA Operations Functions)

The TOE restricts the following tasks, performed through the CA Operations role:

The CA Operations function can do the following in regards to PKI:
- Manage devices (which certificate can be assigned to)
- Manage users (which certificates can be assigned to)
- Request certificates
- Upload certificate requests
- Revoke certificates
- Approve certificate revocations
- Manage auto-enrolment platforms
- Approve certificate requests
- Upload externally managed certificates

The CA Operations function can NOT do the following in regards to PKI:
- Auditing
- Reporting
- Manage whitelist/restriction rules
- Manage revocation and ocsp configuration

The following is limited to CA operator:
- Manage autoenrollment platforms
- Manage external certificates
- Issue certificates
- Revoke certificates
- Request certificate be issued

7.6.4 FMT_MOF.1(4) Management of Security Functions Behaviour (Admin/Officer Functions)

The TOE restricts the following tasks, performed through the Admin/Officer role: The Admin/Officer function is aligned to the “CC-Registration Officer” role provided in the json file.

The Admin/Officer function can do the following in regards to PKI:
- Add users
- Edit users
- View users Evidence of Identity (EOI)
- Edit users EOI
- Manager users

The Admin/Officer function can NOT do the following in regards to PKI:
- Reporting
- Certificate issuance
- Certificate revocations
- Manage devices
- Manage authority information
- Manage revocation information
- Submit certificate/revocation requests
- Manage requests

The following is limited to Admin/Officer:
- Management of EOI

7.6.5 FMT_MOF.1(5) Management of Security Functions Behaviour (Auditor Functions)

The TOE restricts the following tasks, performed through the Auditor role:

The Auditor function can do the following in regards to PKI:
- Manage audit configurations
- Manage tenancy configurations
- View audit reports and dashboards
- Search
- Access reporting and analytics
- Generate and export reports
- View SIEM information
- Manage SIEM configurations

The Auditor function can NOT do the following in regards to PKI:
- Certificate issuance
- Certificate revocations
- Manage devices
- Manage authority information
- Manage revocation information
- Submit certificate/revocation requests
- Manage requests

The following is limited to Auditor role:
- Manage audit config
- Manage and view SIEM
- Export reports

7.6.6 FMT_MTD.1 Management of TSF Data

All users with logon or API key access to the TSF are privileged users and, therefore, any activity is performed only by privileged users. No activity is permitted prior to login.

All data in the TSF is only accessed by privileged users.

7.6.7 FMT_SMF.1 Specification of Management Functions

Management functions are specified in the Jellyfish Client Admin Guide.

Guidance on secure execution of these roles is provided in the administrative guidance provided with the TOE.

7.6.8 FMT_SMR.2 Restrictions on Security Roles

All user authentication is performed in the TSF using usAuthentication. Auditor and RA Operator roles are restricted in standard administration and can only be granted by modifying the central database.

Interfaces available to users are articulated in the administrative guidance, which also discusses role separation requirements.

7.7 FPT: Protection of the TSF

7.7.1 FPT_APW_EXT.1 Protection of Privileged User Passwords

All passwords are stored using salting and hashing within the Postgresql database. All authentication data is subject to this storage and cannot be accessed in plaintext.

7.7.2 FPT_FLS.1 Failure with Preservation of Secure State

The following failure modes result in the system defaulting to a secure state where the system is no longer accessible:

- Exhaustion of resources in the audit database
- Failure to communicate with the HSM

An Operational Environment administrator will be required to remediate the issues prior to system re-activation.
All key material and user data is preserved in the HSM and the TOE database respectively, leading to containment from manipulation during the failure.

7.7.3 FPT_ITT.1 Basic Internal Data Transfer Protections

All data transferred within the system is protected by mTLS as supported by the Consul microservices manager.

7.7.4 FPT_KST_EXT.1 No Plaintext Key Export

The HSM prevents plaintext key export of all private keys, and Jellyfish CA is designed to omit this interface from the system.

7.7.5 FPT_KST_EXT.2 TSF Key Protection

Jellyfish CA ensures that all key operations must be performed by authenticated users and that the keys are stored in the HSM for protection

7.7.6 FPT_NPE_EXT.1 NPE Constraints

All NPE interaction with the system is through certificate signing requests. CSRs are validated in line with rules of the system for all users, including custom certificate profiles.

7.7.7 FPT_RCV.1 Manual Trusted Recovery

Health status checks informs microservices to disconnect from target if it reaches an unhealthy state. Maintenance mode shuts down remote accessible interfaces until systemctl restart is run in the Operational Environment to restore microservices.

7.7.8 FPT_SKP_EXT.1 Protection of Keys

All keys for the system are stored in a HSM in the Operational Environment. Access is controlled by the TOE and all keys are encrypted at rest.

7.7.9 FPT_STM.1 Reliable Time Stamps

All services within the Jellyfish CA and associated services are able to be configured to synchronise to a network time source to allow time synchronisation across services to allow correlation of event log and auditing information.

7.7.10 FPT_TUD_EXT.1 Trusted Updates

Cogito maintain an email list to maintain communication with system owners with regards to future updates to the system.
Updates are sourced from an authenticated source managed by Cogito,and packaged as .deb files for use within the System Under Test environment. Owners are instructed in the build guide to follow deployment instructions that ensure cryptographic signatures are validated using X509 certificates and GPG to ensure Code Signing EKU was used as well as valid signatures prior to installation by Operational Environment administrators.
Where source components are not provided through .deb files, hashes will be validated as part of the standard acquisition process.
Jellyfish CA TOE does not verify software builds due to the secure acquisition channel and reliance on underlying operating environment components.

7.8 FTA: TOE Access

7.8.1 FTA_SSL.3 TSF-Initiated Termination

The TOE shall terminate user access in a configurable period defaulting to 15 minutes in the Common Criteria build.
The user will be required to reauthenticate using the mechanisms as identified prior to being given access to the TSFIs they are authorised to access.

7.8.2 FTA_SSL.4 User-Initiated Termination

The TOE automatically logs users out after the main windows is closed.

7.8.3 FTA_TAB.1 Default TOE Access Banners

Users accessing the Operational Environment via SSH will have the banner set in the Linux Message of the Day function.
Users accessing the system via HTTPS will be presented a modal access banner that can be configured through the administrative interface.

7.9 FTP: Protection of the TSF

7.9.1 FTP_ITC.1 Inter-TSF Trusted Channel

The TOE uses HTTPS and TLS channels to ensure secure communication between its services.
Management of the security for these Inter-TSF trusted channels is handled via Consul.
External cryptographic modules are connected through proprietary DLLs provided by HSM vendors.
Authentication for PKCS#11 requires a label and PIN, facilitated by the DLL. Additional authentication and encryption mechanisms are provided by the specific DLL from each vendor.

SoftHSM2's DLL does not involve networking and operates directly on the file system. HSM management is performed through usCrypto, which can securely connect to any HSM using the provider's DLL.

7.9.2 FTP_TRP.1 Trusted Path

Trusted communication channels are established between the:
- HSM,
- Database,
- Directory Services,
- CRL Publication points, and
- RA.

Communication over the trusted channels are performed using HTTPS or TLS, providing trusted communication and preventing disclosure of transmitted or received information.
Only authenticated and authorised components are able to initiate communication via the trusted channels and client and server components perform authentication as part of the channel establishment procedure.
All communications between external entities and the Jellyfish CA, and Jellyfish RA will be via HTTPS or TLS protocols. The establishment of all trusted channels will require authentication before the channel can be established.

8 References

US Government. (2017, December 1). U.S. Government Approved Protection Profile - Protection Profile for Certification Authorities Version 2.1. NIAP. https://www.niap-ccevs.org/protectionprofiles/420

CCRA. (2017, April). Common Criteria for Information Technology Security Evaluation - Part 1: Introduction and General Model v3.1R5. Common Criteria Portal. https://commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf

CCRA. (2017, April). Common Criteria for Information Technology Security Evaluation - Part 2: Security Functional Requirements v3.1R5. Common Criteria Portal. https://commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R5.pdf

CCRA. (2017, April). Common Criteria for Information Technology Security Evaluation - Part 3: Security Assurance Requirements v3.1R5. Common Criteria Portal. https://commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R5.pdf

CCRA. (2017, April). Common Methodology for Information Technology Security Evaluation – Evaluation Methodology v3.1R5. Common Criteria Portal. https://commoncriteriaportal.org/files/ccfiles/CEMV3.1R5.pdf

Boeyen, S., Santesson, S., Polk, T., Housley, R., Farrell, S., & Cooper, D. (2008). Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC Editor. https://doi.org/10.17487/RFC5280

Housley, R. (2004). Cryptographic Message Syntax (CMS). RFC Editor. https://doi.org/10.17487/RFC3852

Myers, M., & Schaad, J. (2008). Certificate Management over CMS (CMC). RFC Editor. https://doi.org/10.17487/RFC5272

Schaad, J. (2005). Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF). RFC Editor. https://doi.org/10.17487/RFC4211

9 Acronyms

Acronym	Meaning
AES	Advanced Encryption Standard
AOR	Authorised Organisational Representative
API	Application Programming Interface
CA	Certification Authority
CBC	Cipher Block Chaining
CC	Common Criteria
CCM	Counter with CBC-Message Authentication Code
CCMP	CCM Protocol
CCTL	Common Criteria Testing Laboratory
CMC	Certificate Management over CMS
CMS	Cryptographic Message Syntax
CRL	Certificate Revocation List
CSS	Certificate Status Server
DEK	Data Encryption Key
DES	Data Encryption Standard
DH	Diffie-Hellman
DHE	Diffie Hellman Key Exchange
DKM	Derived Keying Material
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
EDC	Error Detection Code
EEPROM	Electrically Erasable Programmable Read-Only Memory
ESP	Encapsulating Security Payload (IPsec)
FFC	Finite-Field Cryptography
FIPS	Federal Information Processing Standards
GCM	Galois/Counter Mode
HMAC	Keyed Hash Message Authentication Code
HSM	Hardware Security Module
HTTPS	HyperText Transfer Protocol Secure
I&A	Identification and Authentication
IKE	Internet key Exchange
IPsec	Internet Protocol Security
IUT	Implementation Under Test
IV	Initialisation Vector
JWT	JSON Web Token
KAT	Known Answer Tests
KDF	Key Derivation Function
KEK	Key Encryption Key
KW	Key Wrap
KWP	Key Wrapping with Padding
MAC	Message Authentication Code
MODP	Modular Exponential
NAT	Network Address Translation

NIST	National Institute of Standards and Technology
NPE	Non-person Entity
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
OID	Object Identifier
PGP	Pretty Good Privacy
PKI	Public Key Infrastructure
PKV	Public Key Verification
PP	Protection Profile
RA	Registration Authority
RAM	Random Access Memory
RBG	Random Bit Generator
rDSA	RSA Digital Signature Algorithm
REK	Root Encryption Key
RFC	Request for Comment
RNGVS	Random Number Generator Validation System
RSA	Rivest Shamir Adleman
SA	Security Association (IPsec)
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TPM	Trusted Platform Module
TSF	TOE Security Function
TSS	TOE Summary Specification

Annex A – Protobuf Audit Event

```
// Event is an Audit Event, as specified in the audit plugins project.
message Event {
        // EventId is the unique identifier for this event, used for consistency when resolving events routed to multiple audit services.
        string event_id = 1;
        // What happened to cause the event.
        What what = 2;
        // When the event occurred.
        When when = 3;
        // Where the event originally occurred, and where it was caught and logged.
        Where where = 4;
        // Who (users, machines, services) was involved in the event.
        Who who = 5;
}

message What {
        // EventType is the type of CRUD action the event is associated with or represets.
        EventType event_type = 1;
        // EventLevel is the log severity with which to treat this event.
        EventLevel event_level = 2;
        // SecurityEvent is whether this event represnets a security-related action.
        bool security_event = 3;
        // Name is the name of the event being logged, e.g. CreateUser.
        string name = 4;
        // Description is a description of the event being logged, e.g. "Creates a user in the database".
        string description = 5;
        // Data is the JSON-encoded bytes of any data relevant to the event, such as a request message body.
        bytes data = 6;
}

message When {
        // Txid is the transaction ID of the request, a unique identifier for requests and actions generated by a single user-facing endpoint call.
        string txid = 1;
        // LoggedTime is the time at which this even was logged, this must be modified by each system re-logging this event to be consistent with when it was logged.
        google.protobuf.Timestamp logged_time = 2;
```

```
        // HappenedTime is the time at which an event actually occurred, independent of when it was received
by the audit service.
        google.protobuf.Timestamp happened_time = 3;
}

message Where {
        // TenancyId is the id of the customer tenancy in which the event occurred, for example the tenancy
of a user that was edited (or attempted to be edited).
        string tenancy_id = 1;
        // ServiceName is the name of the service which originated the event.
        string service_name = 2;
        // ServiceVersion is the version of the service which originated the event, e.g. "v2.7.6-rc.2+debug1"
or "v3.0.0".
        string service_version = 3;
        // ServiceLocation is the network location of the service, such as an IP or DNS name and Port number
        string service_location = 8;
        // ServiceId is the unique ID for a specific service node, such as the one registered in consul.
        string service_id = 4;
        // GeoLocation is the geographic location at which the event occurred, if available.
        GeoLocation geo_location = 5;
        // ResourceLocation is the URL and/or HTTP method and/or similar details about the way in which the
system was accessed when the event was generated.
        string resource_location = 6;
        // CodeLocation is the project, file, and/or line of code responsible for the event.
        string code_location = 7;
}

message Who {
        // UserId is the ID of the user responsible for the request which caused the event.
        string user_id = 1;
        // TenancyId is the customer tenancy ID which the user responsible for the event belongs to.
        string tenancy_id = 2;
        // SessionId is the ID of the session which authenticated the user to perform the action which caused
the event.
        string session_id = 3;
        // CallerLocation is information about the caller of the request which caused the event. This may be
a client service, a peer, a user directly, etc. Information such as IP, DNS, name, port number etc. should be
recorded if available, but not standard for structuring this detail has yet been decided.
        string caller_location = 4;
        // SourceLocation is information, where available, aobut the user's connection, such as IP, DNS name,
port number which may help identify the machine resource used to generate the request which caused the event.
        string source_location = 5;
```

```
        // user_name is a human-readable conversion of the user_id to display in the front-end
        string user_name = 6;
}

// EventType is the type of CRUD action the event is associated with or represets.
enum EventType {
        UNKNOWN_EVENT_TYPE = 0;
        AUTHENTICATE = 1;
        CREATE = 2;
        READ = 3;
        UPDATE = 4;
        DELETE = 5;
        CRUD_COMPOSITE = 6;
        SYNC = 7;
        COMPLEX = 8;
}

// EventLevel is the log severity with which to treat this event.
enum EventLevel {
        UNKNOWN_EVENT_LEVEL = 0;
        UNSAFE = 1;
        TRACE = 2;
        INFO = 3;
        WARNING = 4;
        ERROR = 5;
}

// GeoLocation is the geographic location at which the event occurred, if available.
message GeoLocation {
        double latitude = 1;
        double longitude = 2;
        string time_zone = 3;
        string country = 4;
        string state = 5;
        string locality = 6;
        string post_code = 7;
        string street_address = 8;
}
```