HYPERSTORE OBJECT STORAGE SOFTWARE PLATFORM

SECURITY TARGET

VERSION 1.3

Cloudian, Inc. – www.cloudian,com
177 Bovet Road, Suite 450, San Mateo, CA 94402 USA

TABLE OF CONTENTS

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
ACL	Access Control List
AES	Advanced Encryption Standard
Amazon S3	Amazon Simple Storage Service
API	Application Programming Interface
AWS	Amazon Web Service
CC	Common Criteria for Information Technology Security Evaluation
CEM	Common Evaluation Methodology for Information Technology Security
CMC	Cloudian Management Console
EAL	Evaluation Assurance Level
FIPS PUB	Federal Information Processing Standards Publications
GbE	Gigabit Ethernet
HMAC	Hash-based Message Authentication Code
HSFS	HyperStore File System
HTTP	Hyper-Text Transport Protocol
HTTPS	Hyper-Text Transport Protocol Secure
KDF	Key Derivation Function
NIST SP	National Institute of Standards and Technology Special Publications
NTP	Network Time Protocol
ntpd	NTP daemon
QoS	Quality of Service
RESP	Redis Serialization Protocol
REST	REpresentational State Transfer
RSA	Rivest-Shamir-Adleman
SF	Security Function
SFR	Security Functional Requirement
SFP	Security Function Policy
SHA-1	Secure Hash Algorithm 1

Abbreviation	Description
SSE	Server-Side Encryption
SSH	Secure Shell
SSL	Secure Socket Layer
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
URI	Uniform Resource Identifier

DEFINITIONS

Definition	Description
Amazon S3	Cloud computing web service offered by AWS, which provides object storage through web services interfaces (REST and SOAP)
Apache Cassandra	Open source distributed database
API	Set of routines, protocols, and tools for building software and applications
Auto-tiering	Automated tiered storage (also automated storage tiering) is a policy based method of transferring of data across different tiers (types) of storage devices and media
Bucket	Logical container for storing of data objects (comparable to a folder in a conventional file system)
Metadata	Data that provides information about other data, e.g. administrative metadata that provides information to help manage a resource, such as when and how it was created, file type, and who can access it
Object storage	Computer data storage architecture that manages data as objects, where each object typically includes the data itself, a variable amount of metadata, and a globally unique identifier
Puppet	Open-source SW configuration management tool
RESP	Serializes different data types like integers, strings, and arrays
REST	Architectural style and approach to communications often used in web services development
RESTful	Typically used to refer to web services implementing such an architecture as REST
RESTful API	API that uses HTTP requests to GET, PUT, POST, and DELETE data; (RESTful web service) based on REST technology
Serialization	The process of translating data structures or object state into a format that can be stored or transmitted, and reconstructed later
SSL/TLS	Protocol that guarantees privacy and data integrity between client/server applications communicating over the Internet
TSF	Combined functionality of all hardware, software, and firmware of a TOE that must be relied upon for the correct enforcement of the SFRs

NOTATIONS AND FORMATTING

The notations and formatting used in this ST are consistent with version 3.1 Revision 5 of the Common Criteria (CC).

The **refinement** operation is used to add detail to a requirement, and thus further restricts a requirement. Refinement of security requirements is denoted by **bold text**. Deleted words are denoted by ~~strike-through text~~.

The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections are denoted by *italicized* text in square brackets, [*Selection value*].

The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignment is indicated by showing the value with bold face in square brackets, [**Assignment_value**].

The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing the iteration identifier following the component identifier.

Assets: Assets to be protected by the TOE are given names beginning with "AS." – e.g. AS.CLASSIFIED_INFO.

Assumptions: TOE security environment assumptions are given names beginning with "A."- e.g., A.Security_Procedures.

Threats: Threat agents are given names beginning with "TA." – e.g., TA.User. Threats to the TOE are given names beginning with "TT." – e.g., TT.Filter_Fails. TOE security environment threats are given names beginning with "TE."-- e.g., TE.Crypto_Fails.

Policies: TOE security environment policies are given names beginning with "P."—e.g., P.Information_AC.

Objectives: Security objectives for the TOE and the TOE environment are given names beginning with "O." and "OE.", respectively, - e.g., O.Filter-msg and OE.Clearance.

1. ST Introduction (ASE_INT)

1.1. ST and TOE References

The following table identifies the Security Target (ST).

Item	Identification
ST title	HyperStore Object Storage Software Platform Security Target
ST version	See document control information
ST author	System Sikkerhet AS

The following table identifies the Target of Evaluation (TOE).

Item	Identification
TOE name	HyperStore Object Storage Software Platform
TOE version	HyperStore version 7.2 Software

The following table identifies common references for the ST and the TOE.

Item	Identification
CC Version	3.1 Revision 5
Assurance level	EAL2 augmented with ALC_FLR.1
Protection Profile	None

1.2. TOE Overview

The HyperStore object storage software platform TOE is a multi-tenant object storage system, which consolidates unstructured data objects and data files to a single, limitlessly scalable storage. The TOE provides:

- Secure multi-tenancy, with capability to securely have multiple groups of users reside on a single, shared infrastructure. Data for each user is logically separated from other users' data and cannot be accessed by any other user or user group unless access permission is explicitly granted.
- Uncompromising data protection, where storage policies are ways of protecting data so that it's durable and highly available to users. When a bucket is created, a specific data protection policy is selected.
- Horizontal scalability, where the TOE provides near-unlimited capacity scalability. Running on commodity hardware, the TOE can scale up to thousands of nodes across multiple data centers, supporting millions of users and hundreds of petabytes of data.

The TOE supports the Amazon S3 API, and enables any service provider or enterprise to deploy an S3-compliant multi-tenant storage cloud. The TOE is close to 100% compatible with Amazon S3's REST API («S3 Native»), but there are parts of Amazon's S3 API that apply to internal Amazon capabilities that are not included with the TOE's API. Customers' existing S3 applications will work with the HyperStore service, and existing S3 development tools and libraries can be used for building HyperStore client applications.

The TOE has a fully distributed, peer-to-peer architecture, with no single point of failure; and is resilient to network and node failures with no data loss due to the automatic replication and recovery processes inherent to the architecture. A HyperStore cluster can be deployed across multiple data centers and multiple geographic regions to provide redundancy and resilience in the event of a single node, data center or even an entire region outage.

A HyperStore cluster can start small with initial deployments of just three HyperStore nodes. As demand grows, capacity can be expanded non-disruptively by adding nodes, of any size, to the cluster (the HyperStore cluster has flexibly to grow from terabytes to petabytes with no down time).

The figure below shows the conceptual and functional distinctions between the levels within a TOE (stated as "System" in the figure). From most granular to broadest the levels are: vNode, Node, Data Center and Region.

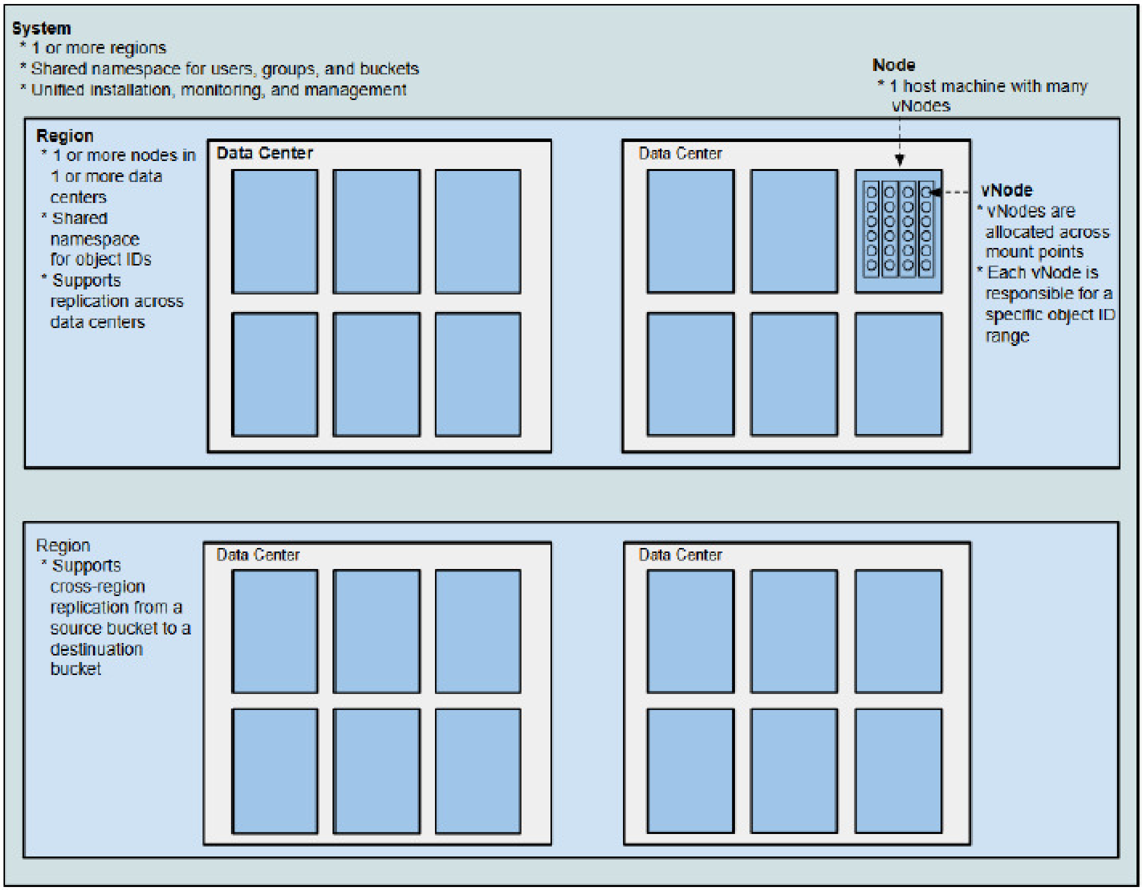
System
* 1 or more regions
* Shared namespace for users, groups, and buckets
* Unified installation, monitoring, and management
Node
* 1 host machine with many vNodes
Region
* 1 or more nodes in 1 or more data centers
* Shared namespace for object IDs
* Supports replication across data centers
Data Center
Data Center
vNode
* vNodes are allocated across mount points
* Each vNode is responsible for a specific object ID range
Region
* Supports cross-region replication from a source bucket to a destinuation bucket
Data Center
Data Center

Figure 1: TOE Levels

The TOE provides intuitive management tools, where system administrators can set storage quotas and usage rate limits on a per-group and per-user basis. System administrators can provision the user groups and individual users who they want authorize to use the TOE S3 service. A system administrator will provision groups first, and then once one or more groups exist individual users can be added to each group (all users must belong to a group). The system administrator can create one or more users who have group administrator privileges. Group administrators can set quotas and rate controls for individual members of their group.

Read and write access controls are supported at per-bucket and per-object granularity. Objects can also be exposed via public URLs for regular web access, subject to configurable expiration periods and number of allowable downloads.

A "bucket" is a logical container in which data objects can be stored, and is comparable to a folder in a conventional file system. At least one bucket must be created before any data objects can be stored. In an S3-based storage system, each data entity stored in a bucket is known as an "object". Each object has its own unique URI (derived from the object name) that S3 applications use when retrieving the object. An S3 storage system does not have a hierarchical structure like a conventional file system. However, objects can be named in such a way as to imply a hierarchy.

1.2.1. Usage and major security features of the TOE

TOE has been designed in such a way that it can be combined with commodity hardware.

TOE makes it possible to provide the following important security capabilities:

1. HyperStore AES-256 Server-Side Encryption enables enterprises and service providers to easily encrypt data stored at rest.
2. SSL/TLS encryption ensures data confidentiality for data in transit (HTTPS).
3. With S3-compatible object-level ACLs, system administrators can better manage access to buckets and objects.

1.2.2. TOE type

The TOE is categorized as a multi-tenant object storage system.

The TOE supports the Amazon Simple Storage System (S3) API, and enables any service provider or enterprise to deploy an S3 compatible multi-tenant storage cloud.

1.2.3. Required non-TOE hardware and software

TOE requires the following server appliance:

- A HyperStore node, which is a commodity computer/server (that is centrally managed by the HyperStore cluster)

TOE requires the following software/hardware requirements for a HyperStore node:

Processor and number of cores required	1CPU 8 cores
Memory required/ recommended	32GB/64GB RAM
Disc space required/ recommended	• 2x160/300 GB SSD (for RAID-1 mirrored hosting of the OS as well as Cassandra and Redis databases storing system metadata) • 12x2/4 TB HDD (for ext4 file systems storing object data) (JBOD, no RAID)
Operating system	One of the following endpoint OS: • Red Hat Enterprise Linux 7.0 or newer • CentOS 7.0 or newer
Networking connectivity required/ recommended	1x1/2x10 GbE Port(s)

Table 1: HyperStore node SW/HW requirements

1.3. TOE Description

The figure below shows the major service components that comprise the HyperStore Object Storage Software Platform TOE, the connections between those services, the direction of the connections, and the default listening ports to which connections are made.

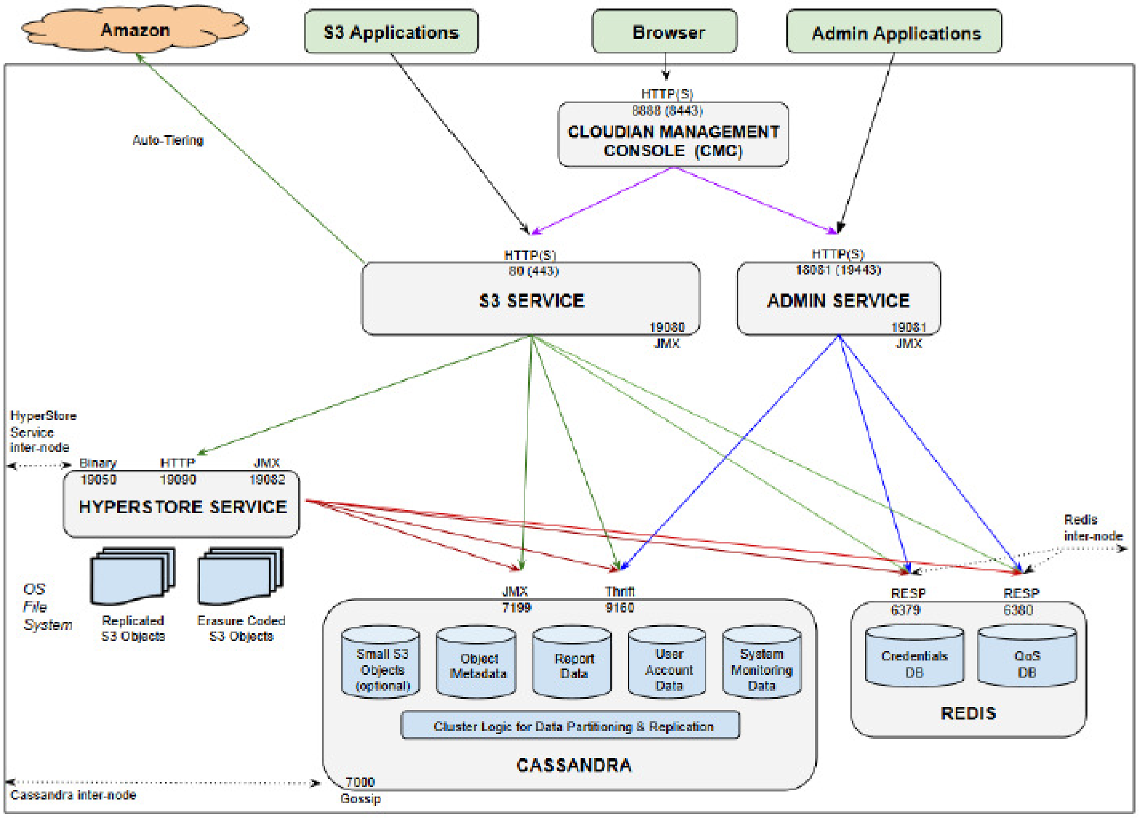

Figure 2: Major TOE services

The TOE is composed of several types of services each of which plays a role in implementing storage service. The table below shows the major TOE services and their corresponding <service-name>, and how the TOE installation script distributes these services across a multi-node cluster. There are common services that are installed to and run on every node, and specialized support services that are installed and run on only one or a sub-set of nodes.

Service Category	Service	<service name>	Where It Is Installed
Common services	Simple Storage System (S3)	*cloudian-s3*	Every TOE node
	Cloudian Management Console (CMC)	*cloudian-cmc*	Every TOE node
	Admin	*cloudian-s3*	Every TOE node
	HyperStore	*cloudian-hyperstore*	Every TOE node
	Cassandra	*cloudian-cassandra*	Every TOE node
Specialized support services	Redis Credentials DB Master	*cloudian-redis-credentials*	One node per entire TOE
	Redis Credentials DB Slaves	*cloudian-redis-credentials*	Two nodes per TOE data center, and slaves will not be on same node as master
	Redis QoS DB Master(s)	*cloudian-redis-qos*	One node per TOE service region
	Redis QoS DB	*cloudian-redis-qos*	One node per TOE data

Service Category	Service	<service name>	Where It Is Installed
	Slave(s)		center, and slave(s) will not be on same node as master(s)
	Redis Monitor	*cloudian-redismon*	One primary node and one backup node per TOE
	Monitoring Data Collector		One primary node and one backup node per TOE service region
	Cloudian Monitoring Agent	*cloudian-agent*	Every TOE node

Table 2: Major TOE services node installed

The three common services S3, CMC, an Admin have both TOE external and TOE internal interfaces, while the services HyperStore, Cassandra and the specialized support services only have TOE internal interfaces.

Simple Storage System (S3) service

The S3 service processes S3 REST requests incoming from S3 client applications including the Cloudian Management Console service. By default HTTPS (HTTP over SSL/TLS) requests to the S3 service from client applications are enabled and handled by self-signed certificates, but HTTP requests will also be responded to. HTTP support for the S3 service will be disabled and HTTPS will be used in the Common Criteria configuration.

On the back side, the S3 service interfaces with:

- The HyperStore service, Cassandra "UserData_<policyid>" keyspaces, and Cassandra "ECKeyspace" keyspace in order to store, retrieve, and delete users' S3 data objects
- The Cassandra User Account Data keyspace to update and retrieve user account information
- The Cassandra Report Data keyspace to update users' transaction history
- The Redis Credentials DB to implement user authentication, S3 bucket validation, and other functions in support of S3 request processing
- The Redis QoS DB to enforce group and user level quality of service restrictions

(Note: Cassandra keyspaces are approximately equivalent to databases).

The TOE supports an "auto-tiering" feature whereby objects can be automatically moved from local TOE storage to a remote storage system over HTTPS, on a defined schedule. The TOE supports auto-tiering from a local TOE bucket to any of several types of destinations systems:

- S3-compliant systems, like Amazon S3, Amazon Glacier, Google Storage Cloud, or a different S3-compliant system of choosing
- Microsoft Azure
- Spectra Logic BlackPearl

Admin service

The HyperStore Admin service implements a RESTful HTTPS API to perform administrative operations such as:

- Provisioning groups and users
- Managing quality of service (QoS) controls
- Creating and managing rating plans
- Generating usage data reports
- Charge-back/Show-back reporting

All configuration of TOE settings through the Admin service is only supported by HTTPS connections. A self-signed certificate comes bundled with the TOE and is used by the Admin service automatically, meaning that the administrator does not have to take any action in regard to an SSL/TLS certificate for the Admin Service (for HTTPS support).

The Cloudian Management Console (CMC) service is a client to the Admin service, which is closely integrated with the S3 service; and the Admin service access the Cassandra service.

Cloudian Management Console (CMC) service

The Cloudian Management Console (CMC) service is a web-based user interface for system administrators, group administrators, and users. All TOE management through the CMC service occurs by means of HTTP over SSL/TLS.

The CMC service can connect via SSH to the Puppet master node or HyperStore nodes when implementing node management functions. The Cloudian installation script automatically populates this setting.

The CMC acts as a client to the Admin service and the S3 service. The functionality available through the CMC depends on the user type associated with a user's login ID (system admin, group admin, or user).

System administrators can use the CMC service to perform a variety of system administration and service administration tasks:

- Provisioning groups and users
- Managing quality of service (QoS) controls
- Creating and managing rating plans
- Generating usage data reports
- Charge-back/Show-back reporting
- Viewing and managing users' stored data objects
- Setting access control rights on users' buckets and stored objects

System administrators do not have a storage service account and cannot upload their own objects to storage.

Group administrators can perform a more limited range of admin tasks pertaining to their own group. A group administrator is an administrator of a particular group of storage service users, and will be able to use the CMC to perform administrative functions for the group, such as adding users or setting user-level QoS profiles. A group administrator has a storage service account and can upload their own objects to storage. A group administrator can also manage stored objects on behalf of particular users within the group.

End users will be able to use the CMC to create storage buckets, upload and download S3 service objects, display reports on their service usage, and manage their service access credentials.

HyperStore service

As an object store, the Apache open source storage platform Cassandra provides a wealth of valuable built-in functionality including data partitioning, automatic replication, easy cluster expansion, quorum calculation, and so on. For storing small data objects, Cassandra also provides good performance. But as the data size increases, storing data on the Linux file system becomes more efficient than storing it in Cassandra.

The HyperStore system uses a hybrid storage solution where Cassandra can optionally be used for small S3 data objects while the Linux file system on Cassandra nodes is used for larger S3 data objects. The area of the Linux file system where S3 object data is stored is called the HyperStore File System (HSFS).

The HyperStore service manages the implementation of this hybrid solution. The general strategy is that Cassandra capabilities are used to determine the distributed data management information such as the nodes that a specific object should be written to. Then at the storage layer, the S3 object data is stored in either Cassandra or the HSFS depending on whether the object is larger than a configurable size threshold that is set for each storage policy. By default, the size threshold is set to 0 so that all S3 objects are stored in the HSFS rather than in Cassandra. Storing all S3 objects in the file system rather than splitting them between Cassandra and the file system makes certain cluster operations like repair and cleanup somewhat simpler.

Within the HSFS, objects can be stored and protected in either of two ways:

- Replicated storage, where a configurable number of copies of each data object are maintained in the system
- Erasure coded storage, where each object is encoded into a configurable number of unique fragment sets

CASSANDRA SERVICE

The TOE uses the Apache open source storage platform Cassandra to store several types of data. S3 client applications do not access Cassandra databases directly; all S3 client access is to the S3 service, which in turn accesses Cassandra in support of S3 operations.

The TOE creates and uses several distinct "keyspaces" (approximately equivalent to databases) within Cassandra:

- The UserData_<policyid> keyspaces store:
  - S3 user bucket information
  - S3 object metadata. (Note that an S3 object’s metadata is always stored in Cassandra even if the object itself is stored in the HSFS)
  - S3 objects up to the configurable size threshold. By default this threshold is set to 0 bytes and no S3 objects are stored in Cassandra
- The AccountInfo keyspace stores information about S3 user accounts and group accounts
- The Report Data keyspace stores system-wide, per-group, and per-user S3 usage data, in support of the HyperStore usage reporting functionality. It will also store per-bucket usage data if per bucket usage tracking is enabled
- The System Monitoring Data keyspace stores system monitoring statistics in support of TOE’s monitoring functionality. It also stores a history of per-node data repair operations; and stores token range maps that are used by the system when cluster nodes are added.
- The ECKeyspace keyspace does not actually store any erasure coded object data; rather, the TOE creates this keyspace so that the HyperStore erasure coding feature can leverage Cassandra functions for token-based mapping of objects (erasure coded object fragments, in this case) to nodes within the storage cluster

REDIS CREDENTIALS/QOS SERVICES

The TOE uses the lightweight, open source Redis key-value data store to store a variety of data that supports HyperStore S3 service features. The services S3, Admin, and HyperStore are the clients to Redis Credentials DB and Redis QoS DB. Communication is through Redis Serialization Protocol (RESP).

There are two types of Redis DBs in a HyperStore deployment:

- The Redis Credentials DB stores user credentials and additional S3 operation supporting data such as multi-part upload session information and public URL access counters
- The Redis QoS DB stores user-level and group-level Quality of Service (QoS) settings that have been established by system administrators. The DB is also used to keep count of user requests, so that QoS limits can be enforced by the system.

Each Redis DB is implemented across two or more nodes, with the nodes playing different roles:

- Master, where all write requests from Redis clients are implemented on the master node. There is only one master node for each Redis DB.
- Slave, where in each Redis DB, data from the Redis master node is asynchronously replicated on to one or more slave nodes (at least one slave node per data center). The slave nodes support doing reads for Redis clients but not writes. If a master node fails, the master role is automatically failed over to a slave node. This fail-over process is managed by the Redis Monitor service.

REDIS MONITOR SERVICE

The Redis Monitor service monitors Redis Credentials DB and Redis QoS DB cluster health and implements automatic failover of the Redis master node role within each of the two Redis DBs. For redundancy, the Redis Monitor service runs on two HyperStore nodes, configured as primary on one node and as backup on the other node.

If the Redis Monitor service detects that a Redis master node has gone down, it promotes an available slave node to the master node role; and informs the Redis cluster's clients (through the services S3, Admin, and HyperStore) of the identity of the new master.

MONITORING DATA COLLECTOR SERVICE

These services play a supporting role for the TOE:

- Cloudian Monitoring Agent, which runs on each HyperStore node and monitors node health and performance statistics. The Agent also plays a role in the triggering of event notification emails to system administrators. System and node statistics are viewable through the CMC; and event notification rules can be configured through the CMC
- Cloudian Monitoring Data Collector, which runs (together with the system maintenance crontab jobs) on one node in each of the service regions, and regularly collects data from the Monitoring Agents. The Monitoring Collector writes its collected node health statistics to Cassandra's System Monitoring Data keyspace. The Monitoring Data Collector is also configured (together with the crontab jobs) on a backup node, and automatic failover to the backup node occurs if the primary node goes offline or if the crontab job "crond" goes down on the primary node.

CLOUDIAN MONITORING AGENT

The Cloudian Monitoring Agent runs on each HyperStore node and monitors node health and performance statistics. The Agent also plays a role in the triggering of event notification emails to system administrators. System and node statistics are viewable through the CMC; and event notification rules can be configured through the CMC as well.

1.3.1. Physical scope

The HyperStore Object Storage Software Platform TOE is purely software and can be installed on several physical devices if the non-TOE requirements in section 1.2.3 are valid.

The supporting guidance documents are:

1. Cloudian HyperStore Administration Guide, Version 1.0
2. Cloudian HyperStore Installation Guide, Version 1.0
3. Cloudian Guidance Documentation, Version 1.1

1.3.2. Logical scope

The HyperStore Object Storage Software Platform TOE is comprised of several security features:

1. Security Audit
2. Identification and Authentication
3. Security Management
4. Protection of the TSF
5. TOE Access
6. Cryptographic Support

Each of the security features identified consists of several security functionalities, as identified below.

1.3.2.1. Security Audit

The TOE provides extensive auditing capabilities, thus generating a comprehensive set of audit logs that identify specific TOE operations. For each event, the TOE records the date and time of the event, the type of event, and the outcome of the event.

1.3.2.2. Identification and Authentication

The TOE provides authentication services for connections to:

- The S3 REST API trough the S3 service
- The secure web-based user interface through the CMC service
- The secure RESTful HTTPS API through the Admin service

The S3/IAM REST requests incoming from client applications require identification and authentication via an access/secret key pair. Initially, the end user logs in using the login REST endpoint, which provides a session token that is included in all subsequent requests. This token is invalidated after the user explicitly logs out, or after a customer configurable inactivity timeout expires.

The TOE requires authorized system administrators to authenticate via a username and a password, prior to being granted access to any of the management functionalities provided by the secure RESTful HTTPS API through the Admin service. The TOE provides authentication services for system/group administrative users wishing to connect to the Admin service directly through a secure command-line tool like cURL or a REST client application.

The TOE requires authorized system/group administrators and end users to authenticate via a username and a password, prior to being granted access to any of the management functionalities provided by the secure web-based user interface through the CMC service. After successful authentication through the CMC service, the TOE determines the permitted level of access for a user based on the user type associated with the user's login ID (system admin, group admin, or end user).

1.3.2.3. Security Management

The TOE provides secure functionalities for management of general TOE configuration and the security functionality provided by the TOE. All TOE management occurs by means of HTTP over SSL/TLS. There are three primary user cases that shape the default TOE roles: System Admin, Group Admin, and User.

Configuration of auto-tiering rules for individual buckets can be managed by a system administrator or by an end user.

The functionality available through the CMC service depends on the TOE role. System administrators can perform a wide range of system maintenance and operational tasks as well as provisioning and managing user accounts. Group admins can perform a much more limited range of tasks, pertaining specifically to their group. End users can use the CMC service to create and configure storage buckets and to upload or download data.

Managing through the Admin service, implemented by a RESTful HTTPS API, is designed to be used by system/group administrators from within an internal network. System/Group administrators can provision users and groups, manage rating plans and QoS controls, retrieve monitoring data, and perform other administrative tasks.

1.3.2.4. Protection of the TSF

The TOE protects against interference and tampering by untrusted subjects by implementing identification, authentication, and access controls to limit configuration to authorized administrators and users.

The TOE internally maintains the date and time. This date and time is used as the timestamp that is applied to audit records generated by the TOE.

1.3.2.5. TOE Access

System/Group administrator sessions and end user sessions will be terminated after a configurable period of inactivity, requiring administrators/users to re-authenticate.

1.3.2.6. Cryptographic Support

The TOE provides cryptography in support of management by means of HTTP over SSL/TLS.

The TOE supports server-side encryption (SSE) to protect the confidentiality of data at rest in the TOE region and data at flight to a remote storage system.

2. Conformance Claims (ASE_CCL)

This TOE and ST are conformant with the following specifications.

Item	Identification
Part 2 of the ISO/IEC 15408 international standard	Common Criteria security functional components, April 2017, Version 3.1, Revision 5, conformant
Part 3 of the ISO/IEC 15408 international standard	Common Criteria security assurance components, April 2017, Version 3.1, Revision 5, conformant
Extended SFRs	None
Protection Profiles	None
Packages	None

3. Security Problem Definition (ASE_SPD)

3.1. Threats to security

3.1.1. Assets

Assets	Description
AS.DATA	Sensitive or security functional data, either contained in the TOE storage or moved from the TOE storage to a remote storage system.
AS.KEY	Cryptographic keys contained in the TOE, for encryption of data at rest and data in flight.

3.1.2. Threat Agents

Threat Agents	Description
TA.ATTACKER	A person/company or process with skills and resources to mislead the system in any way necessary to reveal/divulge/misuse data and prevent the system from operating.
TA.ADMIN	Authorized person/process that performs installation and configuration/setup of the TOE to ensure that the TOE operates according to the needs of the enterprise/organization.
TA.USER	Authorized person/process may unintentionally perform unauthorized actions.

3.1.3. Identification of Threats

3.1.3.1. Threats to the TOE

Threats to the TOE	Description
TT.ADMIN_ERROR	The TOE may be incorrectly configured that may result in the TOE's acquisition of ineffective security mechanisms.
Threat agents:	TA.ADMIN
Assets:	AS.DATA
Attack method:	During operation, the administrator unintentionally configures the TOE incorrectly, making the TOE inoperable or resulting in ineffective security mechanisms.
TT.ADMIN_EXPLOIT	A person/company may gain access to an administrator account.
Threat agents:	TA.ATTACKER
Assets:	AS.DATA
Attack method:	A person/company uses hacking methods to exploit missing, weak or incorrectly implemented access control in the TOE.
TT_AUDIT_COMPROMISE	A person/company may modify or remove audit records to mask actions in the past or prevent logging of actions in the future.
Threat agents:	TA.ATTACKER
Assets:	AS.DATA

Threats to the TOE	Description
Attack method:	A person/company uses hacking methods to exploit weakness in the TOE.
TT.CRYPTO_ COMPROMISE	An attacker may compromise cryptographic keys and the data protected by the cryptographic mechanisms.
Threat agents:	TA.ATTACKER
Assets:	AS.DATA and AS.KEY
Attack method:	An attacker causes key or data associated with the cryptographic functionality to be inappropriately accessed (viewed/modified/deleted), thus compromising the cryptographic mechanisms and the data protected by those mechanisms.
TT.HACK_ACCESS	A person/company gets undetected system access to the TOE due to missing, weak and/or incorrectly implemented access control, causing potential violations of integrity, confidentiality or availability.
Threat agents:	TA.ATTACKER
Assets:	AS.DATA
Attack method:	A person/company uses hacking methods to exploit missing, weak or incorrectly implemented access control in the TOE.
TT.MALFUNCTION	A) The TOE may malfunction which may compromise information and data processing. B) The TOE may malfunction which may compromise roles and permissions.
Threat agents:	A) TA.ATTACKER B) TA.ADMIN or TA.USER
Assets:	AS.DATA and AS.KEY
Attack method:	A) A malfunction in the TOE implies unauthorized access to TOE resources. B) A malfunction in the TOE implies that a person will gain unauthorized roles and permissions in TOE.

3.1.3.2. Threats to the TOE Environment

Threats to the TOE environment	Description
TE.DATALOSS	End users may store or transmit sensitive data in a manner that is inconsistent with a defined organizational policy, leading to loss of confidentiality.
Threat agents:	TA.USER
Assets:	AS.DATA
Attack method:	A person may operate the TOE incorrectly according to system policies.

3.2. Organizational Security Policies (OSP)

Organizational security Policies	Description
P.ACCOUNTABILITY	The authorized users of the TOE shall be held accountable for their actions within the TOE.

Organizational security Policies	Description
P.ADMIN_ACCESS	An authorized administrator must manage the TOE securely.
P.CRYPTOGRAPHIC	The TOE shall provide cryptographic functions for its own use, including encryption/decryption operations.

3.3. Assumptions

Assumptions	Description
A.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.
A.PHYSICAL	The TOE shall presumably be located in physically secure environment that can be accessed only by the authorized administrators.
A.TRUSTED_ADMIN	The administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.

4. Security objectives (ASE_OBJ)

This chapter defines the security objectives for the TOE and its supporting environment. The security objectives are intended to counter identified threats, comply with defined organizational security policies, and address applicable assumptions.

4.1. TOE Security Objectives

This section defines the security objectives that are to be addressed by the TOE.

Security Objectives	Description
O.ACCESS	The TOE must allow authorized users to access to only appropriate TOE functions and data.
O.AUDIT	The TOE shall record and maintain security-related events associated with users in order to enable tracing of responsibilities for security-related acts and shall provide means to review the recorded data.
O.CRYPTOGRAPHY	The TOE shall provide cryptographic functions to maintain the confidentiality of data at rest and data in flight.
O.IDAUTH	The TOE must be able to identify and authenticate users prior to allowing access to their functions and data.
O.MANAGE	The TOE must include a set of functions that allow effective management of its functions and data.
O.MONITOR	The TOE must be able to provide notifications and evidence of defined traffic.
O.PROTECTION	The TOE must protect itself and its resources from unauthorized modifications and access to its functions and data.
O.TIME_STAMPS	The TOE shall provide reliable time stamps and the capability for the administrator to set the time-source used for these time stamps, in that the administrator can configure the TOE to synchronize its clocks with NTP servers.

4.2. Operational Environment Security Objectives

This section defines the security objectives that are to be addressed by the operational environment of the TOE.

Security Objectives	Description
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.PHYSICAL	Those responsible for the TOE must ensure that the hardware on which the TOE and OS are installed is protected from any physical attack.
OE.TRUSTED_ADMIN	The administrator of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.

4.3. Security Objectives Rationale

The following tracing shows which security objectives address which threats, policies (OSPs) and assumptions.

	Threats							Policies			Assumptions		
	TT.ADMIN_ERROR	TT.ADMIN_EXPLOIT	TT.AUDIT_COMPROMISE	TT.CRYPTO_COMPROMISE	TT.HACK_ACCESS	TT.MALFUNCTION	TE.DATALOSS	P.ACCOUNTABILITY	P.ADMIN_ACCESS	P.CRYPTOGRAPHIC	A.NO_GENERAL_PURPOSE	A.PHYSICAL	A.TRUSTED_ADMIN
TOE Security Objectives													
O.ACCESS	X	X	X		X				X				
O.AUDIT			X		X	X	X	X					
O.CRYPTOGRAPHY						X				X			
O.IDAUTH		X	X		X			X					
O.MANAGE	X				X		X		X				
O.MONOTOR							X						
O.PROTECTION				X	X	X							
O.TIME_STAMPS								X					
Operational Environment Security Objectives													
OE.NO_GENERAL_PURPOSE											X		
OE.PHYSICAL			X									X	
OE.TRUSTED_ADMIN	X	X			X			X	X				X

Table 3: Mapping of Objectives to Threats, Policies and Assumptions

The following table is a set of justifications that shows that all threats, policies (OSPs), and assumptions are effectively addressed by the security objectives.

Threat/Policy/Assumption	Security Objective Rationale
TT.ADMIN_ERROR	*The TOE may be incorrectly configured that may result in the TOE's acquisition of ineffective security mechanisms.* O.ACCESS and O.MANAGE provide authorized users the capability to view and manage configuration settings. OE.TRUSTED_ADMIN ensures that the administrators are non-hostile and are trained to appropriately manage and administer the TOE.
TT.ADMIN_EXPLOIT	*A person/company may gain access to an administrator account.* O.ACCESS restricts access to administrative functions to the authorized users. O.IDAUTH includes mechanisms to authenticate TOE users and place controls on user sessions.

	OE.TRUSTED_ADMIN ensures that the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner.
TT_AUDIT_COMPROMISE	*A person/company may modify or remove audit records to mask actions in the past or prevent logging of actions in the future.* O.ACCESS requires that users shall access only appropriate TOE functions and data. O.AUDIT specifies that management actions are audited, allowing such access to be monitored. O.IDAUTH requires that users must be identified and authenticated before access is granted, thus inhibiting unauthorized users from gaining access to TOE data. OE.PHYSICAL aims to prevent access to the TOE server appliances by those aiming to access TOE data.
TT.CRYPTO_ COMPROMISE	*An attacker may compromise cryptographic keys and the data protected by the cryptographic mechanisms.* O.PROTECTION ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked.
TT.HACK_ACCESS	*A person/company gets undetected system access to the TOE due to missing, weak and/or incorrectly implemented access control, causing potential violations of integrity, confidentiality or availability.* O.ACCESS and O.IDAUTH provide the means to identify and authenticate the TOE users. The correct identity of the user is the basis for any decision of the TOE about an attempt of a user to access data. O.AUDIT provides the TOE the capability to detect and create records of security-relevant events associated with users. O.MANAGE restricts the ability to modify the security attributes associated with access control rules, access to authenticated and unauthenticated services, etc. to the authorized users. These objectives ensure that no other user can modify the information flow policy to bypass the intended TOE security policy. O.PROTECTION ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked. OE.TRUSTED_ADMIN ensures that the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner.
TT.MALFUNCTION	*The TOE may malfunction which may compromise information and data processing. The TOE may malfunction which may compromise roles and permissions.* O.AUDIT provides the TOE the capability to detect and create records of security-relevant events associated with users. O.CRYPTOGRAPHY requires the TOE to implement cryptographic services to provide confidentiality protection of data in rest and flight.

	O.PROTECTION ensures that the TOE will have adequate protection from external sources and that all TOE Security Policy functions are invoked.
TE.DATALOSS	*End users may store or transmit sensitive data in a manner that is inconsistent with a defined organizational policy, leading to loss of confidentiality.* O.AUDIT covers the recording of information gathered and policy violations, and requires that it can be reviewed by authorized administrators. O.MANAGE addresses the need for an effective set of management functions that allow data loss policy to be specified, applied, and its implementation monitored. O.MONOTOR addresses examination of administrator notifications regarding system events.
P.ACCOUNTABILITY	*The authorized users of the TOE shall be held accountable for their actions within the TOE.* O.AUDIT provides the administrator with the capability of recording the actions of a specific user, or review the audit trail based on the identity of the user. Additionally, the administrator's user identifier is recorded when any security relevant change is made to the TOE (e.g. modifying TSF data, start-stop of the audit mechanism). O.IDAUTH requires the TOE to identify and authenticate users prior to allowing any TOE access on behalf of those users. O.TIME_STAMPS requires the TOE to provide a reliable time stamp (settable only by the authorized administrator). The audit mechanism is required to include the current date and time in each audit record. OE.TRUSTED_ADMIN ensures that the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner.
P.ADMIN_ACCESS	*An authorized administrator must manage the TOE securely.* O.ACCESS and O.MANAGE provide authorized users the capability to view and manage configuration settings. OE.TRUSTED_ADMIN ensures that the administrators are non-hostile and are trained to appropriately manage and administer the TOE.
P.CRYPTOGRAPHIC	*The TOE shall provide cryptographic functions for its own use, including encryption/decryption operations.* O.CRYPTOGRAPHY requires the TOE to implement cryptographic services to provide integrity and confidentiality protection of the TOE.
A.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.NO_GENERAL_PURPOSE ensures that there are no general-purpose computing capabilities (e.g., the ability to execute

	arbitrary code or applications) on the TOE.
A.PHYSICAL	*The TOE shall presumably be located in physically secure environment that can be accessed only by the authorized administrators.* OE.PHYSICAL ensures that the environment provides physical security, commensurate with the value of the TOE and the data it contains.
A.TRUSTED_ADMIN	*The administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.* OE.TRUSTED_ADMIN ensures that the administrators of the TOE shall not have any malicious intention, shall receive proper training on the TOE management, and shall follow the administrator guidelines.

Table 4: Rationale between Objectives and SPDs

5. Extended Components Definition (ASE_ECD)

If applicable, this chapter defines security components for the TOE not already defined in CC part 2 or CC part 3. However, this Security Target does not need additional security components in addition to those provided by the Common Criteria.

6. Security requirements (ASE_REQ)

6.1. Security Functional Requirements (SFRs)

Functional Class	Functional Component	
FAU: Security audit	FAU_GEN.1	Audit data generation
	FAU_GEN.2	User identity association
	FAU_SAR.1	Audit review
	FAU_STG.1	Protected audit trail storage
FCS: Cryptographic support	FCS_CKM.1	Cryptographic key generation
	FCS_CKM.4	Cryptographic key destruction
	FCS_COP.1	Cryptographic operation
FDP: User Data Protection	FDP_ACC.1a	Subset access control (S3)
	FDP_ACF.1a	Security attribute based access control (S3)
	FDP_ACC.1b	Subset access control (ADMIN)
	FDP_ACF.1b	Security attribute based access control (ADMIN)
	FDP_ACC.1c	Subset access control (CMC)
	FDP_ACF.1c	Security attribute based access control (CMC)
	FDP_IFC.1a	Subset information flow control (S3)
	FDP_IFF.1a	Simple security attributes (S3)
	FDP_IFC.1b	Subset information flow control (Upload)
	FDP_IFF.1b	Simple security attributes control (Upload)
FIA: Identification and authentication	FIA_ATD.1	User attribute definition
	FIA_UAU.1	Timing of authentication
	FIA_UAU.5	Multiple authentication mechanisms
	FIA_UID.1	Timing of identification
FMT: Security management	FMT_MSA.1	Management of security attributes
	FMT_MSA.3	Static attribute initialisation
	FMT_MTD.1	Management of TSF data

Functional Class	Functional Component	
	FMT_SMF.1	Specification of Management Functions
	FMT_SMR.1	Security roles
FPT: Protection of the TSF	FPT_STM.1	Reliable time stamps

Table 5: Security Functional Requirements

6.1.1. Security Audit (FAU)

6.1.1.1. FAU_GEN.1 Audit data generation

Dependencies: FPT_STM.1 Reliable time stamps

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;
b) All auditable events for the [*not specified*] level of audit; and
c) [**Security events (alerts) from application logs and transaction (request) logs generated by the TOE major services**].

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:
a) Date and time of the event, type of event, subject identity (if applicable), 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, [**no additional information**].

6.1.1.2. FAU_GEN.2 User identity association

Dependencies: FAU_GEN.1 Audit data generation
FIA_UID.1 Timing of identification

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.

6.1.1.3. FAU_SAR.1 Audit review

Dependencies: FAU_GEN.1 Audit data generation

FAU_SAR.1.1 The TSF shall provide [**System Administrators**] 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 user to interpret the information.

6.1.1.4. FAU_STG.1 Protected audit trail storage

Dependencies: FAU_GEN.1 Audit data generation

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2 The TSF shall be able to [*prevent*] unauthorised modifications to the stored audit records in the audit trail.

6.1.2. Cryptographic Support (FCS)

6.1.2.1. FCS_CKM.1 Cryptographic key generation

Dependencies: [FCS_CKM.2 Cryptographic key distribution, or
FCS_COP.1 Cryptographic operation]
FCS_CKM.4 Cryptographic key destruction

FCS_CKM.1.1 The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [**NIST Special Publication 800-135**] and specified cryptographic key sizes [**128, 256**] that meet the following: [**FIPS PUB 197**].

6.1.2.2. FCS_CKM.4 Cryptographic key destruction

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation]

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [**zeroization of keys**] that meets the following: [**FIPS PUB 140-2**].

6.1.2.3. FCS_COP.1 Cryptographic operation

Dependencies: [FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1.1 The TSF shall perform [**cryptographic operations listed in Table 6: Cryptographic Operations**] in accordance with a specified cryptographic algorithm [**listed in Table 6: Cryptographic Operations**] and cryptographic key sizes [**listed in Table 6: Cryptographic Operations**] that meet the following: [**standards listed in Table 6: Cryptographic Operations**].

Cryptographic operations	Cryptographic algorithm	Key sizes (bits)	Standards
Encryption/decryption	AES	128, 256	FIPS PUB 197
Encryption/decryption	RSA	2048	NIST SP 800-131A
Key Generation	RSA	2048	FIPS PUB 186-4
Digital Signature	RSA	2048	FIPS PUB 186-4
TLS Session Keys Generation	TLS KDF	All TLS Session Key Sizes	NIST SP 800-135
SSH Session Key Generation	SSH KDF	All SSH Session Key Sizes	NIST SP 800-135
HMAC	HMAC-SHA-1	160	FIPS PUB 198-1

Table 6: Cryptographic Operations

6.1.3. User Data Protection (FDP)

6.1.3.1. FDP_ACC.1a Subset access control (S3)

Dependencies: FDP_ACF.1 Security attribute-based access control

FDP_ACC.1.1 The TSF shall enforce the [**S3 Access Control Policy**] on [
Subjects: **Group Administrators, End Users**;
Objects: **S3 buckets and S3 objects**;
Operations: **S3 REST requests incoming from S3 client applications including CMC**].

6.1.3.2. FDP_ACF.1a Security attribute based access control (S3)

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialisation

FDP_ACF.1.1 The TSF shall enforce the [**S3 Access Control Policy**] to objects based on the following: [
Requests from (external) S3/IAM client applications
Subjects: **Group Administrators, End Users, IAM Users;**
Subject security attributes: **Access key, Secret key;**

Requests from the CMC service (internal S3 client application)
Subjects: **Group Administrators, End Users**;
Subject security attributes: **Access key, Secret key;**

Requests from all S3 client applications
Objects: **S3 buckets and S3 objects;**
Object security attributes: **Permission settings according to Access Control List (ACL)**]**.**

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [**The users can perform supported S3 operations on S3 buckets and S3 objects**].

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [**None**].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [**None**].

6.1.3.3. FDP_ACC.1b Subset access control (ADMIN)

Dependencies: FDP_ACF.1 Security attribute-based access control

FDP_ACC.1.1 The TSF shall enforce the [**Admin Access Control Policy**] on [
Subjects: **System Administrators**;
Objects: **Configuration settings/files**;
Operations: **Inbound RESTful HTTPS API requests**].

6.1.3.4. FDP_ACF.1B Security attribute based access control (Admin)

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialisation

FDP_ACF.1.1 The TSF shall enforce the [**Admin Access Control Policy**] to objects based on the following: [
Subjects: **System Administrators**;
Subject attributes: **Username, password**;
Objects: **Configuration settings/files**;
Object attributes: **None**].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [**The administrator can read and modify data and configuration files**].

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [**None**].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [**None**].

6.1.3.5. FDP_ACC.1C Subset access control (CMC)

Dependencies: FDP_ACF.1 Security attribute-based access control

FDP_ACC.1.1 The TSF shall enforce the [**CMC Access Control Policy**] on [
Subjects: **System Administrators**;
Objects and operations: **Admin Access Control Policy**;

Subjects: **Group Administrators, End Users;**
Objects and Operations: **S3 Access Control Policy**].

6.1.3.6. FDP_ACF.1C Security attribute based access control (CMC)

Dependencies: FDP_ACC.1 Subset access control
FMT_MSA.3 Static attribute initialisation

FDP_ACF.1.1 The TSF shall enforce the [**CMC Access Control Policy**] to objects based on the following: [
Subjects: **System Administrators**;
Subject attributes: **Username, password**;
Objects and all security attributes: **Admin Access Control Policy**;

Subjects: **Group Administrators, End Users;**
Subject attributes: **Username, password**;
Objects with security attributes: **S3 Access Control Policy**].

FDP_ACF.1.2 The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [**Admin Access Control Policy, S3 Access Control Policy**].

FDP_ACF.1.3 The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [**None**].

FDP_ACF.1.4 The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [**None**].

6.1.3.7. FDP_IFC.1A Subset information flow control (S3)

Dependencies: FDP_IFF.1 Simple security attributes

FDP_IFC.1.1 The TSF shall enforce the [**S3 Information Flow Control SFP**] on [
Subjects: **S3 client applications including CMC;**
Information: **Data for S3 objects and S3 buckets;**
Operations: **Allowing or rejecting S3 REST requests containing supported S3 operations on S3 buckets and S3 objects**].

6.1.3.8. FDP_IFF.1A Simple security attributes (S3)

Dependencies: FDP_IFC.1 Subset information flow control
FMT_MSA.3 Static attribute initialisation

FDP_IFF.1.1 The TSF shall enforce the [**S3 Information Flow Control SFP**] based on the following types of subject and information security attributes: [
Subjects: **S3 client applications including CMC;**
Subject security attributes for external S3 client applications: **Access key, Secret key;**
Subject security attributes for CMC: **Access key, Secret key;**
Information: **Data for S3 objects and S3 buckets;**
Information security attributes: **Object name, bucket name, bucket destination address**].

FDP_IFF.1.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [**All S3 REST requests containing supported S3 operations on data for S3 objects and S3 buckets, using the information security attributes, will be handled according to the following: Each S3 bucket and S3 object has an ACL attached to it as a sub-resource, defining which user-accounts or user-groups are granted access and the type of access. When a request is received against a resource, TOE checks the corresponding ACL to verify the requester has the necessary access permissions**].

FDP_IFF.1.3 The TSF shall enforce the [**no additional rules**].

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: [**None**].

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: [**None**].

6.1.3.9. FDP_IFC.1B Subset information flow control (Upload)

Dependencies: FDP_IFF.1 Simple security attributes

FDP_IFC.1.1 The TSF shall enforce the [**Upload Information Flow Control SFP**] on [
Subjects: **Auto-tiering feature;**
Information: **Data in S3 objects transferred from S3 buckets**;
Operations: **Move to a remote storage system**].

6.1.3.10. FDP_IFF.1B Simple security attributes control (Upload)

Dependencies: FDP_IFC.1 Subset information flow control
FMT_MSA.3 Static attribute initialisation

FDP_IFF.1.1 The TSF shall enforce the [**Upload Information Flow Control SFP**] based on the following types of subject and information security attributes: [
Subjects: **Auto-tiering feature;**
Subject security attributes: **Defined schedule;**
Information: **Data in S3 objects transferred from S3 buckets;**
Information security attributes: **Destination storage systems**].

FDP_IFF.1.2 The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold: [**Data in S3 objects stored at S3 buckets, which are set up for auto-tiering, are moved from TOE, using the subject and information security attributes**].

FDP_IFF.1.3 The TSF shall enforce the [**no additional rules**].

FDP_IFF.1.4 The TSF shall explicitly authorise an information flow based on the following rules: [**None**].

FDP_IFF.1.5 The TSF shall explicitly deny an information flow based on the following rules: [**None**].

6.1.4. Identification and Authentication (FIA)

6.1.4.1. FIA_ATD.1 User attribute definition

Dependences: None.

FIA_ATD.1.1 The TSF shall maintain the following list of security attributes belonging to individual users: [
S3 REST requests from external S3 client applications
Access key, Secret key;

IAM REST requests from external IAM client applications
Access key, Secret key;

Request from admin applications through RESTful HTTPS API
Username and password;

Requests from web-based user interface
Username and password;

HyperStore Shell CLI requests through SSH for System Administrators
Username and password].

6.1.4.2. FIA_UAU.1 Timing of authentication

Dependences: FIA_UID.1 Timing of identification

FIA_UAU.1.1 The TSF shall allow [**HTTPS and SSH connection establishment**] on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

6.1.4.3. FIA_UAU.5 Multiple authentication mechanisms

Dependencies: No dependencies.

FIA_UAU.5.1 The TSF shall provide [**password mechanisms, and key mechanism**] to support user authentication.

FIA_UAU.5.2 The TSF shall authenticate any user's claimed identity according to the [**if the user requests are incoming from external IAM or S3 client applications a key mechanism shall be used; for all other users a password mechanism shall be used**].

6.1.4.4. FIA_UID.1 Timing of identification

Dependences: None.

FIA_UID.1.1 The TSF shall allow [**HTTPS and SSH connection establishment**] on behalf of the user to be performed before the user is identified.

FIA_UID.1.2 The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user.

6.1.5. Security management (FMT)

6.1.5.1. FMT_MSA.1 Management of security attributes

Dependencies: [FDP_ACC.1 Subset access control, or
FDP_IFC.1 Subset information flow control]
FMT_SMR.1 Security roles
FMT_SMF.1 Specification of Management Functions

FMT_MSA.1.1 The TSF shall enforce the [**S3 Access Control Policy**] to restrict the ability to [*query, modify,* ***create***] the security attributes [**permissions settings for S3 objects and S3 buckets according to ACL**] to [**Group Administrators, and End Users**].

6.1.5.2. FMT_MSA.3 Static attribute initialisation

Dependencies: FMT_MSA.1 Management of security attributes
FMT_SMR.1 Security roles

FMT_MSA.3.1 The TSF shall enforce the [**S3 Information Flow Control SFP**] to provide [*restrictive*] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2 The TSF shall allow the [**Group Administrators, and End Users**] to specify alternative initial values to override the default values when an object or information is created.

6.1.5.3. FMT_MTD.1 Management of TSF data

Dependencies: FMT_SMR.1 Security roles

FMT_SMF.1 Specification of Management Functions

FMT_MTD.1.1 The TSF shall restrict the ability to [***manage***] the [**TSF data identified in table 7: Management of TSF Data**] to [**System Administrators, Group Administrators, End Users**].

TSF Data	Operation	Role
S3 buckets and S3 objects	Manage	Group Administrator, End User
Permission settings on S3 Buckets	Manage	Group Administrator, End User
Permission settings on S3 Objects	Manage	Group Administrator, End User
Users and groups	Manage	System Administrator, Group Administrator
All other Admin data	Manage	System Administrator

Table 7: Assurance requirements

6.1.5.4. FMT_SMF.1 Specification of Management Functions

Dependencies: None.

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [**Permission set management on S3 buckets and S3 objects; administrative configuration tasks at the Admin service**].

6.1.5.5. FMT_SMR.1 Security roles

Dependencies: FIA_UID.1 Timing of identification

FMT_SMR.1.1 The TSF shall maintain the roles [**System Administrator, Group Administrator, and End User**].

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

6.1.6. Protection of the TSF (FPT)

6.1.6.1. FPT_STM.1 Reliable time stamps

Dependencies: None.

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps.

6.2. Security assurance requirements (SARs)

The assurance level of the TOE is EAL2, augmented with ALC_FLR.1.

Assurance Class	Assurance Components
ADV: Development	ADV_ARC.1 Security architecture description
	ADV_FSP.2 Security-enforcing functional specification
	ADV_TDS.1 Basic design
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
ALC: Life-cycle support	ALC_CMC.2 Use of a CM system
	ALC_CMS.2 Parts of the TOE CM coverage
	ALC_FLR.1 Basic Flaw Remediation
	ALC_DEL.1 Delivery procedures
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.2 Security objectives
	ASE_REQ.2 Derived security requirements
	ASE_SPD.1 Security problem definition
	ASE_TSS.1 TOE summary specification
ATE: Tests	ATE_COV.1 Evidence of coverage
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
AVA: Vulnerability assessment	AVA_VAN.2 Vulnerability analysis

Table 8: Assurance requirements

6.3. Security Requirements Rationale

6.3.1. Relation between SFRs and security objectives

The following tracing shows which SFRs address which security objectives for the TOE.

Requirements Objectives	FAU_GEN.1	FAU_GEN.2	FAU_SAR.1	FAU_STG.1	FCS_CKM.1	FCS_CKM.4	FCS_COP.1	FDP_ACC.1a,b,c	FDP_ACF.1a,b,c	FDP_IFC.1a,b	FDP_IFF.1a,b	FIA_ATD.1	FIA_UAU.1	FIA_UAU.5	FIA_UID.1	FMT_MSA.1	FMT_MSA.3	FMT_MTD.1	FMT_SMF.1	FMT_SMR.1	FPT_STM.1
O.ACCESS								X	X	X	X										
O.AUDIT	X	X	X																		X
O.CRYPTOGRAPHY					X	X	X														
O.IDAUTH												X	X	X	X						
O.MANAGE																X	X	X	X	X	
O.MONITOR	X															X		X	X	X	
O.PROTECTION				X														X	X		
O.TIME_STAMPS																		X			X

Table 9: Tracing of functional requirements to Objectives

The following set of justifications shows that all security objectives for the TOE are effectively addressed by the SFRs.

Security Objectives	Security Functional Requirement Rationale

O.ACCESS	*The TOE must allow authorized users to access to only appropriate TOE functions and data.* FDP_ACC.1a, FDP_ACC.1b and FDP_ACC.1c define the Access Control policy that will be enforced on subjects acting on the behalf of users attempting to gain access to objects. All operations that involve access to the data are controlled by the policy. These objects contain the TOE data to be protected. FDP_ACF.1a, FDP_ACF.1b and FDP_ACF.1c define the security attribute used to provide access control to objects based on the TOE's access control policy. FDP_IFC.1a identifies the external IT entities in the Information Flow Control SFP that send information towards the TOE. The SFP will either reject or allow the information flow. FDP_IFF.1a identifies the external IT entity and its security attributes as part of the information flow control SFP. TOE will permit or deny the information flow based on the configured rules. FDP_IFC.1b identifies the internal entities in the Information Flow Control SFP that send information from the TOE. The SFP will either reject or allow the information flow. FDP_IFF.1b identifies the internal entity and its security attributes as part of the information flow control SFP. TOE will permit or deny the information flow based on the configured rules.
O.AUDIT	*The TOE shall record and maintain security-related events associated with users in order to enable tracing of responsibilities for security-related acts and shall provide means to review the recorded data.* FAU_GEN.1 defines the set of events that the TOE must be capable of recording. This requirement ensures that the administrator has the ability to audit any security relevant event that takes place in the TOE. FAU_GEN.2 ensures that the audit records associate a user identity with the auditable event. In the case of authorized users, the association is accomplished with the user ID. In all other cases, the association is based on the source identifier, which is presumed to be the correct identity, but cannot be confirmed since these subjects are not authenticated. FPT_STM.1 supports the audit functionality by ensuring that the TOE is capable of obtaining a time stamp for use in recording audit events. FAU_SAR.1 The administrators are allowed to read the audit records.
O.CRYPTO-GRAPHY	*The TOE shall provide cryptographic functions to maintain the confidentiality of data at rest and data in flight.* FCS_CKM.1 ensures that the TOE is capable of generating cryptographic keys. FCS_CKM.4 provides the functionality for ensuring that keys and key material is zeroized. FCS_COP.1 requires that for data decryption and encryption an approved algorithm is used, and that the algorithm meets the

	standard.
O.IDAUTH	*The TOE must be able to identify and authenticate users prior to allowing access to their functions and data.* FIA_ATD.1 defines the attributes of users, including a user identifier that is used by the TOE to determine a user's identity and enforce what type of access the user has to the TOE, and ensures that untrusted users cannot be associated with a role and reduces the possibility of a user obtaining unauthorized privileges. FIA_UID.1 and FIA_UAU.1 ensures that users are connected, identified and authenticated before they are provided access to the TOE. In order to control logical access to the TOE an authentication mechanism is required. FIA_UAU.5 defines the available authentication mechanisms in the TOE, and specifies the rules that describe how the authentication mechanisms provide authentication and when each is to be used.
O.MANAGE	*The TOE must include a set of functions that allow effective management of its functions and data.* The FMT requirements are used to satisfy this management objective, as well as other objectives that specify the control of functionality. The requirement's rationale for this objective focuses on the capability to perform management functions in order to control the behavior of security functions. FMT_MSA.1 allows users acting in certain roles to manage identified security attributes. The users are assigned to a role according to FMT_SMR.1 Security roles. FMT_MSA.3 requires that the TSF provide default values for relevant object security attributes, which can be overridden by an initial value. FMT_MTD.1 allows users with a certain role to manage values of TSF data. The users are assigned to a role according to FMT_SMR.1 Security roles. FMT_SMR.1 and FMT_SMF.1 specify the roles that are recognized by the TSF for the management functions to be provided by the TSF.
O.MONITOR	*The TOE must be able to provide notifications and evidence of defined traffic.* FAU_GEN.1 requires that the results of scanning activity must be recorded. FMT_MSA.1, FMT_MTD.1, FMT_SMF.1, and FMT_SMR.1 require that the ability to define and distribute policy is restricted to authorized users.
O.PROTECTION	*The TOE must protect itself and its resources from unauthorized modifications and access to its functions and data.* FAU_STG.1 requires the TOE to protect the audit data from deletion and modification. FMT_MTD.1 and FMT_SMF.1 ensure that only authorized users of

	the TOE may manage audit data.
O.TIME_STAMPS	*The TOE shall provide reliable time stamps and the capability for the administrator to set the time-source used for these time stamps, in that the administrator can configure the TOE to synchronize its clocks with NTP servers.* FMT_MTD.1 provides the capability to set the time used for generating time stamps to the authorized user. This functionality allows the authorized user to ensure the time and date are correctly set, while restricting this function from unauthorized use. FPT_STM.1 requires that the TOE be able to provide reliable time stamps for its own use. Time stamps include date and time and are reliable in that they are always available to the TOE.

Table 10: Rationale between Objectives and SFRs

6.3.2. SFR Dependencies

The table below shows the dependencies of the security functional requirement of the TOE and gives a rationale for each of them if they are included or not.

Security functional requirement	Dependency	Dependency Rationale
FAU_GEN.1 Audit data generation	FPT_STM.1 Reliable time stamps	Included
FAU_GEN.2 User identity association	FAU_GEN.1 Audit data generation FIA_UID.1 Timing of identification	Included
FAU_SAR.1 Audit review	FAU_GEN.1 Audit data generation	Included
FAU_STG.1 Protected audit trail storage	FAU_GEN.1 Audit data generation	Included
FCS_CKM.1 Cryptographic key generation	[FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic operation] FCS_CKM.4 Cryptographic key destruction	Included
FCS_CKM.4 Cryptographic key destruction	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]	Included
FCS_COP.1 Cryptographic Operation	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction	Included

Security functional requirement	Dependency	Dependency Rationale
FDP_ACC.1a Subset access control (S3)	FDP_ACF.1 Security attribute based access control	Included
FDP_ACF.1a Security attribute based access control (S3)	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation	Included
FDP_ACC.1b Subset access control (ADMIN)	FDP_ACF.1 Security attribute based access control	Included
FDP_ACF.1b Security attribute based access control (ADMIN)	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation	Included
FDP_ACC.1c Subset access control (CMC)	FDP_ACF.1 Security attribute based access control	Included
FDP_ACF.1c Security attribute based access control (CMC)	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation	Included
FDP_IFC.1a Subset information flow control (S3)	FDP_IFF.1 Simple security attributes	Included
FDP_IFF.1a Simple security attributes (S3)	FDP_IFC.1 Subset information flow control FMT_MSA.3 Static attribute initialisation	Included
FDP_IFC.1b Subset information flow control (Upload)	FDP_IFF.1 Simple security attributes	Included
FDP_IFF.1b Simple security attributes control (Upload)	FDP_IFC.1 Subset information flow control FMT_MSA.3 Static attribute initialisation	Included
FIA_ATD.1 User attribute definition	None	
FIA_UAU.1 Timing of authentication	FIA_UID.1 Timing of identification	Included
FIA_UAU.5 Multiple authentication mechanisms	None	
FIA_UID.1 Timing of identification	None	
FMT_MSA.1 Management of security attributes	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FMT_SMF.1 Specification of Management Functions FMT_SMR.1 Security roles	Included
FMT_MSA.3 Static attribute initialisation	FMT_MSA.1 Management of security attributes FMT_SMR.1 Security roles	Included
FMT_MTD.1 Management of TSF data	FMT_SMF.1 Specification of Management Functions FMT_SMR.1 Security roles	Included

Security functional requirement	Dependency	Dependency Rationale
FMT_SMF.1 Specification of Management Functions	None	
FMT_SMR.1 Security roles	FIA_UID.1 Timing of identification	Included
FPT_STM.1 Reliable time stamps	None	

Table 11: SFR's dependencies and rationale

6.3.3. SAR Rationale

The SARs specified in this ST are according to EAL2, augmented with ALC_FLR.1.

7. TOE Summary Specification (ASE_TSS)

7.1. TOE Security Functions Specification

This section describes the security functions provided by the TOE to meet the security functional requirements specified for the TOE in section 6.1 Security Functional Requirements (SFRs).

The table below shows the mapping between the SFRs and the implementing security functions, and a description is given in the following subsections.

Requirements Functions	FAU_GEN.1	FAU_GEN.2	FAU_SAR.1	FAU_STG.1	FCS_CKM.1	FCS_CKM.4	FCS_COP.1	FDP_ACC.1a,b,c	FDP_ACF.1a,b,c	FDP_IFC.1a,b	FDP_IFF.1a,b	FIA_ATD.1	FIA_UAU.1	FIA_UAU.5	FIA_UID.1	FMT_MSA.1	FMT_MSA.3	FMT_MTD.1	FMT_SMF.1	FMT_SMR.1	FPT_STM.1
SF.ACCESS								X	X												
SF.AUDIT	X	X	X	X																	X
SF.AUTHENTICATION												X	X	X	X						
SF.COMMUNICATION										X	X										
SF.CRYPTOGRAPHY					X	X	X														
SF.MANAGEMENT																X	X	X	X	X	

Table 12: Mapping SFRs to security functions

7.1.1. SF.ACCESS

(FDP_ACC.1a, FDP_ACF.1a, FDP_ACC.1b, FDP_ACF.1b, FDP_ACC.1c, FDP_ACF.1c)
The TOE provides the user data protection security function requirement to manage user access and interaction with TOE data. The TOE enforces access control policy which limits access to the TOE data and to the TOE configuration settings. Access to TOE data and settings is enforced by user account privileges and specific permissions on S3 objects and S3 buckets. A user attempting to access the TOE or TOE data with the incorrect privileges or permissions will be denied access.

End Users access the TOE by means of an external S3 client application; System administrators, Group Administrators and End Users access the TOE by means of the internal S3 client application CMC (web-based user interface); System administrators access TOE settings by means of the RESTful HTTPS API. Access to the TOE through these interfaces requires the correct access type associated with each TOE user. Once granted access to the TOE, the users with the correct privileges and permissions can manage TOE data and settings. TOE users can be denied access to the TOE, if they are attempting to access the TOE from an interface for which they do not have the correct access type.

The Admin Service uses a thread pool to process incoming HTTPS requests from clients. Idle threads are as a main rule terminated if not used within a thread timeout period.

7.1.2. SF. AUDIT

(FAU_GEN.1, FAU_GEN.2)
The TOE generates a comprehensive set of audit logs that identify specific TOE operations whenever an auditable event occurs. Shutdown and start-up of the audit

functions are logged by events for reloading the TOE, and the events when the TOE comes back up.

The major TOE services each generate their own application log. The S3 service, Admin service, and HyperStore service, in addition to generating application logs, also generate transaction (request) logs. In addition, Admin API transactions are logged in the Admin Service application log.

The application logs have descriptions that indicate the following for the major services:

- The date and time the action was initiated
- The default logging level according to the defined types of events (OFF, ERROR, WARN, INFO, DEBUG, TRACE, ALL)
- The message containing the outcome of the event

The transaction (request) logs have descriptions that indicate some of the following for the major services:

- The date and time the action was initiated
- The default logging level according to the defined types of events (OFF, ERROR, WARN, INFO, DEBUG, TRACE, ALL)
- The message containing the outcome of the event
- IP address of the client (subject identity)

In addition, the logging for the S3 service shall indicate the username of the S3 bucket owner.

When logging is enabled on a source bucket, every 10 minutes the TOE generates an access log file for the bucket (if there has been access to the bucket during the past 10 minutes) and stores the log file in a specified destination bucket.

(FAU_SAR.1, FAU_STG.1)
The TOE stores the audit records locally in a limited logging buffer, and protects the records from deletion and modification. The System Administrators are allowed to read the audit records, but no mechanisms are provided for modification or deleting of audit log entries. The audit log entries are stored in the database; if space is exhausted, old entries are discarded.

(FPT_STM.1)
The TOE provides a source of date and time information used in audit event timestamps, receiving clock updates from an NTP server.

7.1.3. SF. AUTHENTICATION

(FIA_ATD.1)
User account information is stored in the TOE and contains the following attributes for the users:

- Requests from external S3/IAM client applications: Access key and Secret key;
- Request from admin applications: Username and password;
- Requests from web-based user interface: Username and password.
- Requests from SSH CLI user interface: Username and password.

(FIA_UAU.1, FIA_UAU.5, FIA_UID.1)
Each user must have an approved HTTPS connection, and be successfully identified and authenticated by the TSF with an access key or a username and password, before access is allowed to the TOE. User identification and authentication by the TSF uses the security attributes of the user account types described above.

When identification and authentication data is entered with a username and a password, the TOE attempts to identify the applicable user account from the provided identity and if a match is found, the password provided is compared against that stored with the user account information in the internal database. If a user account cannot be associated with the provided identity or the provided password does not match that stored with the user account information, identification and authentication will fail.

7.1.4. SF.COMMUNICATION

(FDP_IFC.1a, FDP_IFF.1a)
The TOE data in S3 objects and S3 buckets are protected from unauthorized usage from S3 client applications by using information flow control. The TOE will check the S3 REST requests from S3 client applications before allowing or rejecting the S3 REST requests containing supported S3 operations on S3 buckets and S3 objects. The decision to allow or reject requests will be based on the ACL rules attached to each S3 bucket and to each S3 object, where the TOE will check the corresponding ACLs to verify the requester has the necessary access permissions.

(FDP_IFC.1b, FDP_IFF.1b)
If S3 objects in S3 buckets are set up for auto-tiering, the TOE data in these S3 objects and S3 buckets are moved to a remote storage system at defined schedule.

7.1.5. SF.CRYPTOGRAPHY

(FCS_CKM.1)
In support of secure cryptographic protocols, the TOE supports several key generation schemes, including TLS and SSH as specified in NIST SP 800-135, and RSA as specified in FIPS PUB 186-4. The TOE is fully compliant to SP 800-135 and FIPS PUB 186-4.

(FCS_CKM.4)
The TOE meets all requirements specified in FIPS 140-2 for destruction of keys. All keys within the TOE are zeroizable.

(FCS_COP.1)
The TOE provides encryption and decryption capabilities:

- Using 128 and 256 bits AES, described in FIPS PUB 197;
- Using 2048 bits RSA, described in NIST SP 800-131A.

The TOE provides key generation capabilities:

- Using TLS, described in NIST SP 800-135;
- Using SSH, described in NIST SP 800-135;
- Using 2048 bits RSA, described in FIPS PUB 186-4.

The TOE provides digital signature capabilities:

- Using 2048 bits RSA, described in FIPS PUB 186-4

The TOE provides HMAC capabilities:

- Using HMAC-SHA-1, described in FIPS PUB 198-1.

7.1.6. SF.MANAGEMENT

(FMT_MSA.1, FMT_MSA.3)
The TOE ensures that only authorized users are able to specify the policy definitions to enforce availability. By default, the TOE provides a restrictive information flow policy rule set.

(FMT_MTD.1)
The TOE provides the ability for authorized users to access TOE data, such as S3 buckets/objects, audit data, configuration data, and auto-tiering rules. Each of the

predefined access right levels has a set of permissions that will grant them access to the TOE data.

(FMT_SMF.1)
The TOE provides all the capabilities necessary to securely manage the TOE. The authorized user can connect to the TOE either through the TOE Web UI, SSH, or RESTful API to perform these functions. All general administration is expected to take place through the Web UI or SSH. The specific management capabilities available from the TOE include:

- Permission set management on S3 buckets and S3 objects;
- Administrative configuration tasks at the Admin service.

(FMT_SMR.1)
The TOE authenticates all access to the administrative and management interfaces using a username and password. The TOE supports local administration through the RESTful API and management through Web UI or SSH by means of the roles:

- System Administrator;
- Group Administrator;
- End User.