ENDL
TEXAS

Date: 24 June 2004
To: T10 Technical Committee and SNIA OSD TWG
From: Ralph O. Weber
Subject: Rewrite of OSD Security and Policy/Storage Manager Models

Revision History

This document is being presented in a 'build up' fashion. That is r0 contains an initial amount of changes and each revision after that contains additional details and changes.

r0 Shows only the model split between Capabilities (aka Policy/Storage Management) and Credentials (aka Security) and provides a rudimentary description for fencing
r1 Contains the same normative text as r0, but the strikeouts are removed.

Review Suggestions for r0/r1

- How much more discussion is required regarding fencing?
- Are the attributes pages changes the way the group wants to go:
  - Can Policies and Security share one attributes page and one permissions bit?
  - Are two pages and two bits really needed?
  - Hopefully, there are no complaints about the 'policy version tag'

Summary

The document shows all the changes to OSD r09 needed to:

- Separate Capability handling (aka Policy/Storage Management) from Credential handling (aka Security)
- More TBD

Changes made by this document are shown as red text additions and ~~red text removals~~.

The author believes that this revision of this document addresses the following previously unresolved OSD Letter Ballot comments:

- EMC 3 should be addressed by the content of 4.9.m

This document also shows all approved changes for the clauses modified by this document, specifically: Agilent 3, Agilent 8, Agilent 9, Agilent 10, EMC 12, HP 30, HP 31, HP 32, HP 34, HP 36, HP 40, HP 42, HP 43, HP 47, HP 48, HP 49, HP 50, HP 51, HP 54, HP 55, HP 56, HP 57, HP 58, HP 76, HP 77, HP 120, IBM 39, IBM 40, IBM 41, IBM 43, IBM 44, IBM 46, IBM 47, IBM 48, IBM 49, IBM 50, IBM 52, IBM 53, IBM 55, IBM 56, IBM 57, IBM 58, IBM 59, IBM 142, Intel 21, Lingua 20, Lingua 22, Lingua 23, Lingua 24, Lingua 25, Lingua 26, Lingua 27, Lingua 28, Lingua 30, Lingua 31, Lingua 32, Lingua 36, Lingua 37, LSI 12, Seagate 7, Seagate 10, Seagate 11, Seagate 13, Seagate 14, Seagate 15, Seagate 18, Seagate 19, Seagate 20, Seagate 21, Seagate 51, Veritas 45, Veritas 47, Veritas 49, Veritas 50, Veritas 51, Veritas 52, Veritas 53, Veritas 57, Veritas 58, Veritas 59, Veritas 61, Veritas 62, Veritas 63, Veritas 64, Veritas 66, Veritas 67, Veritas 68, Veritas 69, Veritas 70, Veritas 71, and Veritas 72. These changes are shows as blue text additions and ~~blue text removals~~.

18484 Preston Road, Suite 102 PMB 178, Dallas, TX 75252 214-912-1373 972-596-2775 Fx ENDL_TX@computer.org

4.4 Elements of the example configuration

The example in this subclause (see figure 1) illustrates the three mandatory and two optional constituents of an OSD configuration:

a) Object-Based Storage Devices;
b) Service delivery subsystem;
c) Host systems (i.e., initiator devices);
d) Optionally, a security manager; and
e) Optionally, a policy/storage manager.

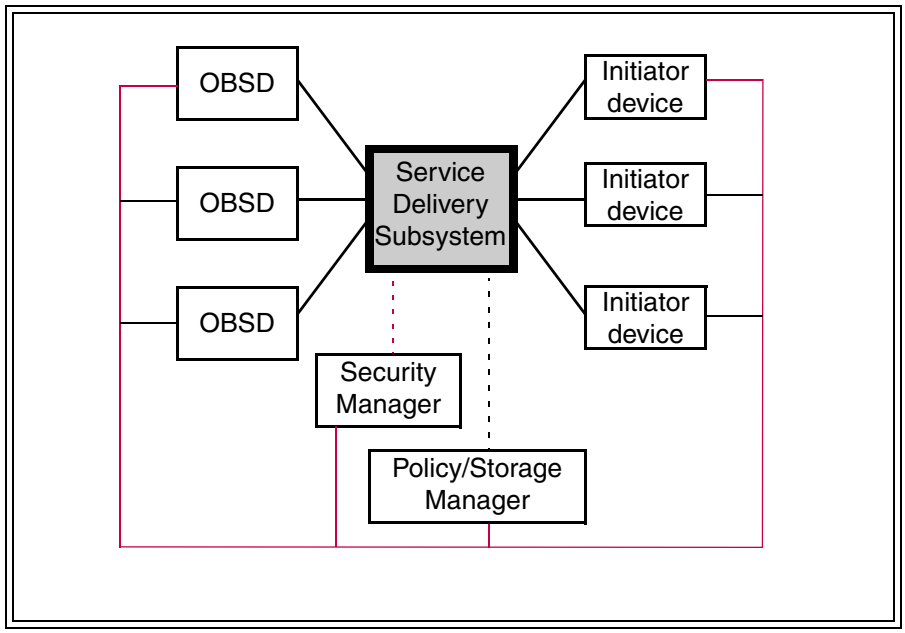

Figure 1 — Example OSD Configuration

The OBSDs are the storage components of the system to be shared (e.g., disc drives, RAID subsystems, tape drives, tape libraries, optical drives, jukeboxes, or other storage devices).

Application clients using multiple SCSI initiator ports share and directly access an OBSD (see 3.1.26) via the service delivery subsystem. The service delivery subsystem is used by the components in the OSD model, except possibly the security manager, to intercommunicate. The OSD security model (see 4.9) does not require the service delivery subsystem to provide security-related services (i.e., authentication and data privacy), but is designed to take advantage of whatever security-related services are provided.

The security manager, if present, ~~coordinates access~~ secures access capabilities in cryptographically protected credentials ~~between~~ for OSD device servers and application clients (see 4.9). The security manager may use the service delivery subsystem and be an application client, but the security manager also may use another mechanism to communicate with the OSD device servers and application clients. When sending credentials to an application client, the security manager shall use a private, authenticated communications mechanism. The security manager may reside in the OBSD, in applications clients, or as a separate entity, but the security requirements on the communications mechanism shall not change based on the location of the security manager.

The policy/storage manager, if present, coordinates access constraints between OSD device servers and application clients (see 4.x). ~~maintains storage policies in a manner that is outside the scope of this standard~~.

4.x Policy/storage management

4.x.1 Overview

The policy/storage manager:

a) Provides access policy controls to application clients via preparation of policy-coordinated capabilities; and
b) Fences OSD objects from access, using the policy version tag attribute to prevent unsafe or temporarily undesirable utilization of OBSD storage.

4.x.2 Policy capabilities

4.x.2.1 Introduction

Each CDB defined by this standard includes a capability (see 4.x.2.2) whose contents specify the command functions (see 3.1.10) that the device server may process in response to the command.

The device server validates that the requested command functions are permitted by the capability based on:

a) The type of functions (e.g., read, write, attributes setting, attributes retrieval); and
b) The OSD object on which the command functions are to be ~~performed~~ processed.

The policies that determine which capabilities are provided to which application clients are outside the scope of this standard.

The policy/storage manager shall coordinate the delivery of capabilities to application clients with the security manager (see 4.9) as follows:

a) If the security method for all partitions in the OSD logical unit is NOSEC (see 4.9.3.2), then the policy/storage manager may:
   A) Allow application clients to prepare their own capabilities;
   B) Coordinate the preparation of capabilities for application client in response to requests from the application clients, the format and transport mechanisms for which are outside the scope of this standard; or
   C) Coordinate the preparation of capabilities with the security manager as described in item b);
   or
b) If the security method other than NOSEC is in use by any partition in the OSD logical unit, then the policy/storage manager shall coordinate the preparation of capabilities with the security manager:
   A) Requiring application clients to request capabilities from the security manager; and
   B) Preparing capabilities only is response to requests from the security manager.

4.x.2.2 Capability format

A capability (see table 1) ~~is the portion of a credential (see 4.9.4.1) that~~ is included in a CDB to enable the device server to verify that the sender is allowed to perform the ~~function~~ command functions (see 3.1.10) specified ~~described~~ by the CDB ~~command~~.

Table 1 — Capability format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				CREDENTIAL/CAPABILITY FORMAT (1h)			
1	KEY VERSION				INTEGRITY CHECK VALUE ALGORITHM			
2	(MSB)							
7				CAPABILITY EXPIRATION TIME				(LSB)
8								
11				AUDIT				
12	(MSB)							
23				CAPABILITY DISCRIMINATOR				(LSB)
24	(MSB)							
29				OBJECT ~~CREATION~~ CREATED TIME				(LSB)
30	OBJECT TYPE							
31								
35				PERMISSIONS BIT MASK				
36	Reserved							
37	OBJECT DESCRIPTOR TYPE				Reserved			
38								
61				OBJECT DESCRIPTOR				

The CREDENTIAL/CAPABILITY FORMAT field (see table 2) specifies the format of the ~~credential and~~ capability. When capabilities are coordinated with the security manager, the capability format also is the credential format.

Table 2 — Credential/capability format values

Value	Description
0h	No ~~credential~~ capability
1h	The format defined by this standard
2h - Fh	Reserved

The KEY VERSION field and INTEGRITY CHECK VALUE ALGORITHM field are used by the security manager. When capabilities are not coordinated with the security manager, the KEY VERSION field and INTEGRITY CHECK VALUE ALGORITHM field are reserved.

The CAPABILITY EXPIRATION TIME field specifies the value of the clock attribute in the Root Information attributes page (see 7.1.2.8) after which this capability is no longer valid.

The AUDIT field is a vendor specific value that the security manager may use to associate the capability and credential with a specific application client.

The CAPABILITY DISCRIMINATOR field contains a nonce (see 3.1.23) that differentiates one capability ~~and credential~~ from another.

~~The OBJECT CREATION TIME field specifies the contents of the creation time attribute in the User Object Timestamps attributes page (see 7.1.2.18), Collection Timestamps attributes page (see 7.1.2.17), Partition Timestamps attributes page (see 7.1.2.16), or Root Timestamps attributes page (see 7.1.2.15), for the OSD object to which the credential applies. A value of zero specifies that any object creation time is allowed.~~

The OBJECT ~~CREATION~~ CREATED TIME field specifies the contents of the ~~creation~~ created time attribute for the OSD object (see table 3) to which the ~~credential~~ capability applies. A value of zero specifies that any object ~~creation~~ created time is allowed.

Table 3 — ~~Creation~~ Created time OSD objects

Object Type (see table 4)	**Attributes page containing ~~creation~~ created time attribute to which ~~credential~~ capability OBJECT ~~CREATION~~ CREATED TIME field is applies**
ROOT	Root Timestamps attributes page (see 7.1.2.15)
PARTITION	Partition Timestamps attributes page (see 7.1.2.16)
COLLECTION	Collection Timestamps attributes page (see 7.1.2.17)
USER	User Object Timestamps attributes page (see 7.1.2.18)

The OBJECT TYPE field (see table 4) specifies the type of OSD object to which this capability allows access ~~and aids in the determination of how to validate the credential and capability~~. When capabilities are coordinated with the security manager, the OBJECT TYPE field is used to select the secret key that is used in validating the credential.

Table 4 — Object type values

Value	**Name**	**~~OSD~~ OSD object type to which access is allowed**
01h	ROOT	Root object
02h	PARTITION	Partition
40h	COLLECTION	Collection
80h	USER	User objects
all other values	Reserved	

The PERMISSIONS BIT MASK field (see table 5) specifies which functions are allowed by this capability. More than one permissions bit may be set within the constraints specified in 4.x.2.3 resulting in a single ~~credential and~~ capability that allows more than one command function.

Table 5 — Permissions bit mask format

Bit / Byte	7	6	5	4	3	2	1	0
31	READ	WRITE	GET_ATTR	SET_ATTR	CREATE	REMOVE	OBJ_MGMT	APPEND ~~Reserved~~
32	DEV_MGMT	GLOBAL	~~SECURITY~~ POL/SEC	Reserved				
33	Reserved							
34	Reserved							
35	Reserved							

A READ bit set to one allows read access to the data in an OSD object, but not to the attributes. For the root object, partitions, and collections the data in the OSD object is the list of other objects contained in the OSD object. A READ bit set to zero prohibits read access to the data in an OSD object.

A WRITE bit set to one allows processing of the WRITE command (see 6.21) ~~write access to the data in a user object~~, but not ~~to the~~ access to user object attributes. A WRITE bit set to zero prohibits processing of the WRITE command ~~write access to the data in a user object~~.

A GET_ATTR (get attributes) bit set to one allows retrieval of (i.e., read access to) the attributes associated with an OSD object. A GET_ATTR bit set to zero prohibits retrieval of attributes except for the attributes in the Current Command attributes page (see 7.1.2.24).

A SET_ATTR (set attributes) bit set to one allows the setting of (i.e., write access to) the attributes associated with an OSD object except for attributes located in the OSD object's policy/security attributes page (e.g., the User Object Policy/Security attributes page (see 7.1.2.23) if the OSD object is a user object). The setting of attributes located in the OSD object's policy/security attributes page is allowed only if both the SET_ATTR bit and the ~~SECURITY~~ POL/SEC bit are set to one. A SET_ATTR bit set to zero prohibits the setting of the attributes associated with an OSD object.

A CREATE bit set to one allows the creation of OSD objects. A CREATE bit set to zero prohibits the creation of OSD objects.

A REMOVE bit set to one allows the removal of OSD objects. A REMOVE bit set to zero prohibits the removal of OSD objects.

An OBJ_MGMT (object management) bit set to one allows command functions that may change how the OSD logical unit handles an OSD object without affecting the stored data, stored attributes, commands in the task set, policies, or security for the OSD object. A OBJ_MGMT bit set to zero prohibits such command functions.

An APPEND bit set to one allows processing of the APPEND command (see 6.2), but not access to user object attributes. A APPEND bit set to zero prohibits processing of the APPEND command.

A DEV_MGMT (device management) bit set to one allows command functions that affect the OSD logical unit. A DEV_MGMT bit set to zero prohibits command functions that affect the OSD logical unit.

A GLOBAL bit set to one allows command functions that may affect all the OSD objects in the OSD logical unit. A GLOBAL bit set to zero prohibits command functions that may affect all the OSD objects in the OSD logical unit.

A ~~SECURITY~~ POL/SEC bit set to one allows command functions that affect the policy/security functions performed for one or more OSD objects. A ~~SECURITY~~ POL/SEC bit set to zero prohibits command functions that affect the policy/security functions performed for one or more OSD objects.

The OBJECT DESCRIPTOR TYPE field (see table 6) specifies the format of information that appears in the OBJECT DESCRIPTOR field.

Table 6 — Object descriptor types

Object Descriptor Type	Name	Description
0h	NONE	The OBJECT DESCRIPTOR field shall be ignored
1h	1OBJECT	A single partition, collection, or user object
2h - Fh		Reserved

If the object descriptor type is 1OBJECT (i.e., 1h), the OBJECT DESCRIPTOR field shall have the format shown in table 7, specifying a single partition, collection, or user object to which the capability allows access.

Table 7 — Single object descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
38	(MSB)			SINGLE OBJECT_ID				
45								(LSB)
46	(MSB)			~~SECURITY~~ POLICY VERSION TAG				
49								(LSB)
50				Reserved				
61								

The SINGLE OBJECT_ID field contains the Partition_ID (see 4.6.4), Collection_Object_ID (see 4.6.6), or User_Object_ID (see 4.6.5) of the OSD object to which the capability allows access, with the type of OSD object being identified by the OBJECT TYPE field (see table 4).

~~NOTE 1 A Partition_ID of zero specifies that access is allowed to both the partition numbered zero and the root object.~~

If the object type field contains 01h (i.e., ROOT) a Partition_ID of zero specifies that access is allowed to the root object. If the object type field contains 02h (i.e., PARTITION) a Partition_ID of zero specifies that access is allowed to partition zero (see 3.1.31).

If the ~~SECURITY~~ POLICY VERSION TAG field contains a value other than zero, the ~~security~~ policy version tag attribute identified by the object type field (see table 8) is compared to the ~~SECURITY~~ POLICY VERSION TAG field contents as part of the algorithm for validating ~~credentials and~~ capabilities described in 4.9.5.2. If the ~~SECURITY~~ POLICY VERSION TAG field contains zero, then no comparison is made to any ~~security~~ policy version tag attribute. The policy/storage manager changes the policy version tag to prevent unsafe or temporarily undesirable accesses to an OSD object. ~~Changing the security version tag attribute is one way the security manager may invalidate credentials (see 4.9.5.5).~~

Table 8 — ~~Security~~ Policy version tag OSD objects

Object Type (see table 4)	**Attributes page containing ~~security~~ policy version tag attribute to which credential ~~SECURITY~~ POLICY VERSION TAG field is compared**
ROOT	Partition Policy/Security attributes page (see 7.1.2.21) for partition ~~0h~~ zero (see 3.1.31)
PARTITION	Partition Policy/Security attributes page (see 7.1.2.21)
COLLECTION	Collection Policy/Security attributes page (see 7.1.2.22)
USER	User Object Policy/Security attributes page (see 7.1.2.23)

4.x.2.3 Credentials and commands allowed

The validity of a specific command and some of the ~~function-related~~ command function (see 3.1.10) related fields in that command is determined by the presence of specific combinations of values in capability fields as shown in table 9. A command function is allowed if at least one row in table 9 allows it, even if a different row that applies does not allow it.

Any command may retrieve or set attributes. ~~and combinations~~ The combinations of capability fields that allow those functions are shown in table 10. Retrieving or setting attributes is allowed if at least one row in table 10 allows it, even if a different row that applies does not allow it.

A single credential for a single object type may allow processing of multiple command functions (e.g., read and write) as well as ~~the retrieval~~ retrieving and setting ~~of~~ attributes by combining the permission bits values described in multiple rows of table 9 and table 10.

Table 9 — Commands allowed by specific capability field values (Sheet 1 of 5)

Commands allowed and CDB fields whose contents are restricted by credential field contents	**Capability Field values that allow a command**		
	Object Type Name	**Permission Bits That Are Set To One**	**Object Descriptor Name**
An APPEND command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	APPEND ~~WRITE~~	1OBJECT
Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 10 are reserved. The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.			
[a] The object descriptor SINGLE OBJECT_ID field shall contain zero.			

Table 9 — Commands allowed by specific capability field values (Sheet 2 of 5)

Commands allowed and CDB fields whose contents are restricted by credential field contents	Capability Field values that allow a command		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
A CREATE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB REQUESTED USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	CREATE	1OBJECT
A CREATE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB REQUESTED USER_OBJECT_ID field equal to zero.	USER	CREATE	NONE
A CREATE AND WRITE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB REQUESTED USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	CREATE and WRITE	1OBJECT
A CREATE AND WRITE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB REQUESTED USER_OBJECT_ID field equal to zero.	USER	CREATE and WRITE	NONE
A CREATE COLLECTION command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB REQUESTED COLLECTION_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	COLLECTION	CREATE	1OBJECT
A CREATE COLLECTION command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB REQUESTED COLLECTION_OBJECT_ID field equal to zero.	COLLECTION	CREATE	NONE
A CREATE PARTITION command with the CDB REQUESTED PARTITION_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	PARTITION	CREATE	1OBJECT
A CREATE PARTITION command with the CDB REQUESTED PARTITION_ID field equal to zero.	PARTITION	CREATE	NONE
Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 10 are reserved. The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.			
[a] The object descriptor SINGLE OBJECT_ID field shall contain zero.			

Table 9 — Commands allowed by specific capability field values (Sheet 3 of 5)

Commands allowed and CDB fields whose contents are restricted by credential field contents	Capability Field values that allow a command		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
A FLUSH ~~OBJECT~~ command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field~~, and the USER_OBJECT_ID field not containing a Collection_Object_ID~~.	USER	OBJ_MGMT	1OBJECT
A FLUSH COLLECTION ~~OBJECT~~ command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB~~ USER_~~ COLLECTION_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field~~, and the USER_OBJECT_ID field containing a Collection_Object_ID~~.	COLLECTION	OBJ_MGMT	1OBJECT
A FLUSH PARTITION ~~OBJECT~~ command with the CDB PARTITION_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field~~ and the CDB USER_OBJECT_ID field equal to zero~~.	PARTITION	OBJ_MGMT	1OBJECT
A FLUSH OSD ~~OBJECT~~ command w~~ith the CDB PARTITION_ID field and the CDB USER_OBJECT_ID field both equal to zero or a FORMAT OSD command~~.	ROOT	OBJ_MGMT	1OBJECT [a]
A FORMAT OSD command.	ROOT	OBJ_MGMT and GLOBAL	1OBJECT [a]
A LIST COLLECTION command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB COLLECTION_OBJECT_ID field containing a value other than zero that matches the contents of the object descriptor SINGLE OBJECT_ID field. ~~If the object descriptor SINGLE OBJECT_ID field contains zero, a LIST COLLECTION command is allowed to list the collections defined in the partition.~~	COLLECTION	READ	1OBJECT
A LIST COLLECTION command with the CDB PARTITION_ID field containing a value that matches the object descriptor SINGLE OBJECT_ID field and the CDB COLLECTION_OBJECT_ID field equal to zero.	PARTITION	READ	1OBJECT

Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 10 are reserved.
The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.

[a] The object descriptor SINGLE OBJECT_ID field shall contain zero.

Table 9 — Commands allowed by specific capability field values (Sheet 4 of 5)

Commands allowed and CDB fields whose contents are restricted by credential field contents	Capability Field values that allow a command		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
A LIST command with the CDB PARTITION_ID field containing a value other than zero that matches the contents of the object descriptor SINGLE OBJECT_ID field.	PARTITION	READ	1OBJECT
A LIST command with the CDB PARTITION_ID field equal to zero.	ROOT	READ	1OBJECT [a]
A PERFORM TASK MANAGEMENT command with function code of ABORT TASK or QUERY TASK, the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field, and the USER_OBJECT_ID field not containing a Collection_Object_ID.	USER	DEV_MGMT	1OBJECT
A PERFORM TASK MANAGEMENT command with function code of ABORT TASK or QUERY TASK, the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field, the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field, and the USER_OBJECT_ID field containing a Collection_Object_ID.	COLLECTION	DEV_MGMT	1OBJECT
A PERFORM TASK MANAGEMENT command with function code of ABORT TASK or QUERY TASK, the CDB PARTITION_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field, and a CDB USER_OBJECT_ID field equal to zero.	PARTITION	DEV_MGMT	1OBJECT
A PERFORM TASK MANAGEMENT command with function code of ABORT TASK or QUERY TASK and the CDB PARTITION_ID field and CDB USER_OBJECT_ID field both equal to zero.	ROOT	DEV_MGMT	1OBJECT [a]
Any PERFORM TASK MANAGEMENT command or PERFORM SCSI COMMAND command.	ROOT	DEV_MGMT and GLOBAL	1OBJECT [a]
A READ command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	READ	1OBJECT
Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 10 are reserved. The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.			
[a] The object descriptor SINGLE OBJECT_ID field shall contain zero.			

Table 9 — Commands allowed by specific capability field values (Sheet 5 of 5)

Commands allowed and CDB fields whose contents are restricted by credential field contents	Capability Field values that allow a command		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
A REMOVE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	REMOVE	1OBJECT
A REMOVE COLLECTION command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB COLLECTION_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	COLLECTION	REMOVE	1OBJECT
A REMOVE PARTITION command with the CDB PARTITION_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	PARTITION	REMOVE	1OBJECT
A SET KEY command with KEY TO SET field equal to 10b or 11b and the CDB PARTITION_ID field containing a non-zero value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	PARTITION	DEV_MGMT and ~~SECURITY~~ POL/SEC	1OBJECT
Any SET KEY command with the CDB PARTITION_ID field equal to zero.	ROOT	DEV_MGMT and ~~SECURITY~~ POL/SEC	1OBJECT [a]
Any SET MASTER KEY command.	ROOT	DEV_MGMT and ~~SECURITY~~ POL/SEC	1OBJECT [a]
A WRITE command with the CDB PARTITION_ID field containing a value that matches the contents of the credential PARTITION_ID field and the CDB USER_OBJECT_ID field containing a value that matches the contents of the object descriptor SINGLE OBJECT_ID field.	USER	WRITE	1OBJECT
Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 10 are reserved. The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.			
[a] The object descriptor SINGLE OBJECT_ID field shall contain zero.			

Table 10 — Capability fields that allow attribute retrieval and setting (Sheet 1 of 3)

Attribute-Related Functions Allowed	Capability Field values that allow attribute-related functions		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
The retrieval of attributes from any attributes page associated with the user object with a Partition_ID matching the value in the credential PARTITION_ID field and a User_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [a, b]	USER	GET_ATTR	1OBJECT
As part of a CREATE command or CREATE AND WRITE command, the retrieval of attributes from any attributes page associated with any user object created by the command. [a] ~~the user object with a Partition_ID matching the value in the credential PARTITION_ID field and the largest valued User_Object_ID created.~~ [a]	USER	GET_ATTR	NONE
The retrieval of attributes from any attributes page associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and a Collection_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [a, b]	COLLECTION	GET_ATTR	1OBJECT
As part of a CREATE COLLECTION command, the retrieval of attributes from any attributes page associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and the Collection_Object_ID created. [a]	COLLECTION	GET_ATTR	NONE
The retrieval of attributes from any attributes page associated with the partition with a Partition_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [a, b]	PARTITION	GET_ATTR	1OBJECT
As part of a CREATE PARTITION command, the retrieval of attributes from any attributes page associated with the created partition. [a]	PARTITION	GET_ATTR	NONE
The retrieval of attributes from any attributes page associated with the root object or in any attributes page associated with partition zero (see 3.1.31). [a]	ROOT	GET_ATTR	~~NONE or~~ 1OBJECT

Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 9 are reserved.
The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.

[a] Attributes in the Current Command attributes page may be retrieved even if the GET_ATTR bit is set to zero.

[b] The command functions allowed for the GET ATTRIBUTES command and SET ATTRIBUTES command are specified by the GET_ATTR bit and SET_ATTR bit, respectively. The GET ATTRIBUTES command and SET ATTRIBUTES command are allowed if only the GET_ATTR bit, or SET_ATTR bit, or both are set to one.

Table 10 — Capability fields that allow attribute retrieval and setting (Sheet 2 of 3)

Attribute-Related Functions Allowed	Capability Field values that allow attribute-related functions		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
The setting of attributes in any attributes page, other than User Object Security attributes page, associated with the user object with a Partition_ID matching the value in the credential PARTITION_ID field and a User_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	USER	SET_ATTR	1OBJECT
As part of a CREATE command or CREATE AND WRITE command, the setting of attributes in any attributes page, other than User Object Security attributes page, associated any user object created by the command. ~~the user object with a Partition_ID matching the value in the credential PARTITION_ID field and the largest valued User_Object_ID created.~~	USER	SET_ATTR	NONE
The setting of attributes in any attributes page, other than Collection Security attributes page, associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and a Collection_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	COLLECTION	SET_ATTR	1OBJECT
As part of a CREATE COLLECTION command, the setting of attributes in any attributes page, other than Collection Security attributes page, associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and the Collection_Object_ID created.	COLLECTION	SET_ATTR	NONE
The setting of attributes in any attributes page, other than Partition Security attributes page, associated with the partition with a Partition_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	PARTITION	SET_ATTR	1OBJECT
As part of a CREATE PARTITION command, the setting of attributes in any attributes page, other than Partition Security attributes page, associated with the created partition.	PARTITION	SET_ATTR	NONE
The setting of attributes in any attributes page, other than Root Security attributes page, associated with the root object or in any attributes page, other than Partition Security attributes page, associated with partition zero (see 3.1.31). [b]	ROOT	SET_ATTR	~~NONE or~~ 1OBJECT

Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 9 are reserved.
The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.

[a] Attributes in the Current Command attributes page may be retrieved even if the GET_ATTR bit is set to zero.
[b] The command functions allowed for the GET ATTRIBUTES command and SET ATTRIBUTES command are specified by the GET_ATTR bit and SET_ATTR bit, respectively. The GET ATTRIBUTES command and SET ATTRIBUTES command are allowed if only the GET_ATTR bit, or SET_ATTR bit, or both are set to one.

Table 10 — Capability fields that allow attribute retrieval and setting (Sheet 3 of 3)

Attribute-Related Functions Allowed	Capability Field values that allow attribute-related functions		
	Object Type Name	Permission Bits That Are Set To One	Object Descriptor Name
The setting of attributes in any attributes page associated with the user object with a Partition_ID matching the value in the credential PARTITION_ID field and a User_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	USER	SET_ATTR and ~~SECURITY~~ POL/SEC	1OBJECT
As part of a CREATE command or CREATE AND WRITE command, the setting of attributes in any attributes page associated any user object created by the command. ~~the user object with a Partition_ID matching the value in the credential PARTITION_ID field and the largest valued User_Object_ID created.~~	USER	SET_ATTR and ~~SECURITY~~ POL/SEC	NONE
The setting of attributes in any attributes page associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and a Collection_Object_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	COLLECTION	SET_ATTR and ~~SECURITY~~ POL/SEC	1OBJECT
As part of a CREATE COLLECTION command, the setting of attributes in any attributes page associated with the collection with a Partition_ID matching the value in the credential PARTITION_ID field and the Collection_Object_ID created.	COLLECTION	SET_ATTR and ~~SECURITY~~ POL/SEC	NONE
The setting of attributes in any attributes page associated with the partition with a Partition_ID matching the value in the object descriptor SINGLE OBJECT_ID field. [b]	PARTITION	SET_ATTR and ~~SECURITY~~ POL/SEC	1OBJECT
As part of a CREATE PARTITION command, the setting of attributes in any attributes page associated with the created partition.	PARTITION	SET_ATTR and ~~SECURITY~~ POL/SEC	NONE
The setting of attributes in any attributes page associated with the root object or in any attributes page associated with partition zero (see 3.1.31). [b]	ROOT	SET_ATTR and ~~SECURITY~~ POL/SEC	~~NONE or~~ 1OBJECT

Combinations of OBJECT TYPE field, PERMISSION BITS field, and OBJECT DESCRIPTOR TYPE field values not shown in this table and table 9 are reserved.
The credential and capability fields not shown in this table may place additional limits on the objects that are allowed to be accessed.

[a] Attributes in the Current Command attributes page may be retrieved even if the GET_ATTR bit is set to zero.
[b] The command functions allowed for the GET ATTRIBUTES command and SET ATTRIBUTES command are specified by the GET_ATTR bit and SET_ATTR bit, respectively. The GET ATTRIBUTES command and SET ATTRIBUTES command are allowed if only the GET_ATTR bit, or SET_ATTR bit, or both are set to one.

4.x.3 Storage fencing

The policy/storage manager is responsible for preventing unsafe or temporarily undesirable utilization of OBSD storage. The conditions under which the policy/storage manager may be call on to do this include:

a) Recovery from errors that make accesses to one or more OSD object unsafe; and
b) Receipt of a request to block accesses from the security manager (see 4.9.5.5).

The policy/storage manager performs this fencing function by changing the policy version tag attribute in the OSD object's policy/security attributes page (e.g., the User Object Policy/Security attributes page (see 7.1.2.23) if the OSD object is a user object) to a non-zero value. After the policy version tag is changed, the device server terminates any command received with a capability whose POLICY VERSION TAG field contains the previous policy version tag value.

4.9 Security

4.9.1 Basic security model

The OSD security model is ~~a credential-based access control system~~ composed of the following components:

a) An OBSD (see 3.1.26);
b) A policy/storage manager (see 4.x);
c) A security manager; and
d) Application clients.

~~Because the OSD security model is a credential based access control system, all requests to the OSD logical unit should be accompanied by a valid credential that allows the application client to perform the requested command functions.~~ The principal function of the security manager is preparing credentials in response to application client requests. A credential is a data structure ~~prepared by the security manager (see 3.1.38)~~ containing a capability prepared by the policy/storage manager and protected by an integrity check value (see 3.1.18), having the following properties:

a) The capability in the credential grants defined access to an OSD logical unit for specific command functions (see 3.1.10); and
b) The integrity check value and other information in the credential but not in the capability protects the capability and commands that include the capability from various attacks described in 4.9

~~that is sent to an application client in order to grant defined access to an OSD logical unit for specific command functions (see 3.1.10) performed on specific OSD objects. The credential includes a capability (see 3.1.4) that the application client copies to each CDB that requests the specified command functions. a cryptographically secured capability and a capability is a set of access rights allowed to the holder of the credential on an OSD object or set of OSD objects.~~

Figure 2 shows the flow of transactions between the components of the OSD security model.

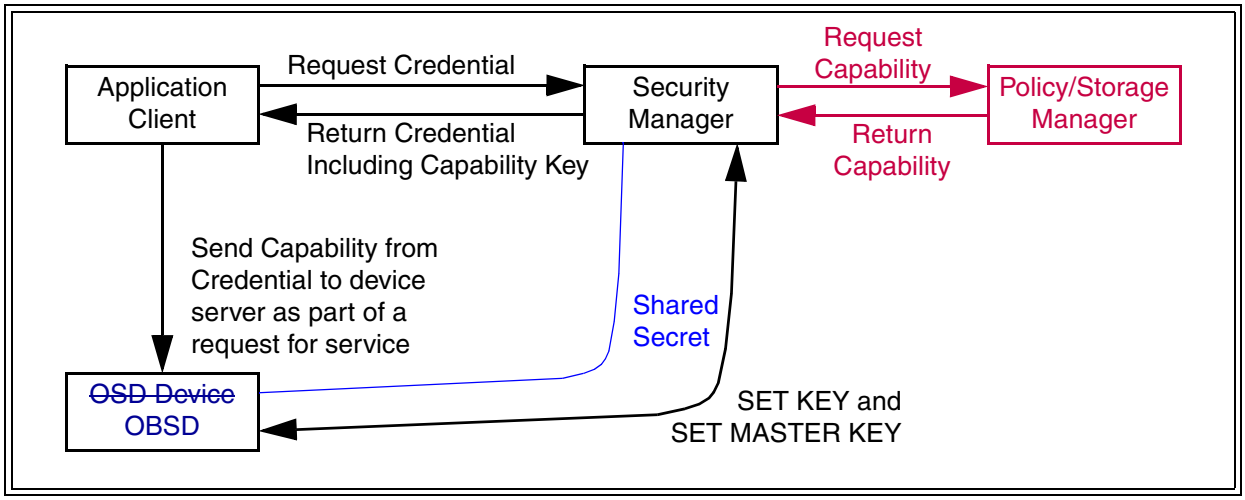

Figure 2 — OSD security model transactions

The security manager generates credentials, including capabilities prepared by the policy/storage manager, for authorized application clients at the request of an application client. The security manager returns a capability key with each credential. The credential gives the application client access to specific OSD components. The capability key allows the application client and device server to authenticate the commands and data they exchange with an integrity check value (see 4.9.7).

The protocol between the application client and the security manager is not defined by this standard~~;~~. ~~h~~However, the structure of the credential returned from the security manager to the application client is.

The device server validates each command received from an application client to confirm that:

a) The credential has not been tampered with (i.e., that the credential was generated by the security manager and includes an integrity check value using a secret key known only to the security manager and OSD device server); and
b) The credential was rightfully obtained by the application client from the security manager or through delegation by another application client (i.e., that the application client knows the capability key that is associated with the credential and has used the capability key to provide a proper integrity check value or values for the command).~~; and~~
~~c) The requested command function is permitted by the capability in the credential based on:~~
~~A) The type of functions (e.g., read, write, attributes setting, attributes retrieval); and~~
~~B) The OSD object on which the command functions are to be performed.~~

The capability key allows the OSD device server to validate that an application client rightfully obtained a credential and that the capability has not been tampered with. An application client that has just the capability (e.g., obtained by monitoring CDBs sent to the OSD device server) but not the capability key is unable to generate commands with valid integrity check value, meaning that application client is denied access to the OSD logical unit. This protocol ~~does~~ allows delegation of a credential if a application client delegates both the credential and the capability key.

The application client requests credentials and capability keys from the security manager for the command functions it needs to perform and sends those capabilities in those credentials to the OSD device server as part of commands that include an integrity check value using the capability key. While the application client is not trusted to follow this protocol, an application client that does not follow the protocol is unlikely to receive service from the OSD device server. ~~While the application client is not trusted to follow this protocol, the protocol is structured in a~~

~~way that makes it in the application client's self-interest to follow the protocol. An application client that does not follow the protocol is unlikely to receive service from the OSD device server.~~

The security manager may authenticate the application client, but the OSD device server does not ~~need to~~ authenticate the application client. It is sufficient for the OSD device server to verify the capabilities and integrity check values ~~send~~ sent by the application client.

4.9.2 Trust assumptions

This subclause describes how each component of the OSD security model trusts the other components.

The OBSD is a trusted component, meaning that once an application client authenticates that it is communicating with a specific OSD logical unit using methods outside the scope of this standard, it trusts the OBSD to:

a) Provide integrity for stored data;
b) Perform the security protocol and functions defined for it by this standard; and
c) Not be controlled in a way that operates to the detriment of the application client's interests.

The security manager is a trusted component. After the security manager is authenticated by the application client and by the OBSD using methods outside the scope of this standard, the security manager is trusted to:

a) Safely store long-lived keys;
b) Apply access controls correctly according to requirements that are outside the scope of this standard;
c) Perform the security functions defined for it by this standard; and
d) Not be controlled in a way that operates to the detriment of the application client's or OSD logical unit's interests.

The application client is not a trusted component. However, the OSD security model is defined so that the application client receives service from the OSD device server only if it interacts with both the security manager and the OSD device server in ways that assure the propriety of the application client's actions.

4.9.m Preparing credentials

In response to a request from an application client, the security manager shall prepare and return a credential as follows:

1) Forward the access requests from the application client to the policy/storage manager. If the policy/storage manager denies the forwarded request an error shall be returned to the requesting application client;
2) Insert the capability returned by the policy/storage manager in the credential;
3) Set the capability KEY VERSION field to the number of the working key ~~contains a value that the device server uses to identify~~ the secret key ~~to be~~ used ~~in computing~~ to compute the credential integrity check value. If a secret key other than a working key is used to compute the credential integrity check value (e.g. for a~~. For capabilities used with the~~ SET KEY command (see 6.23) or ~~to update secret keys other than working keys and~~ SET MASTER KEY command (see 6.24)), then set the capability KEY VERSION field to zero. ~~the key version shall be zero.~~;
4) Set the capability INTEGRITY CHECK VALUE ALGORITHM field to the value that specifies the algorithm used to compute all integrity check values related to this credential ~~command~~. The algorithm may be identified using the supported integrity check value algorithm attributes in the Root Policy/Security attributes page (see 7.1.2.20). The value in the Root Security attribute whose attribute number equals the contents of the INTEGRITY CHECK VALUE ALGORITHM field plus 8000 0000h identifies the integrity check value algorithm;
5) Set the credential OSD SYSTEM ID field to the value in the OSD system ID attribute in the Root Information attributes page (see 7.1.2.8) of the OSD logical unit to which the credential applies;
6) Compute the credential integrity check value as described in 4.9.5.4 and place the result in the CREDENTIAL INTEGRITY CHECK VALUE field in the credential; and

7) Return the credential thus constructed to the application client with the credential integrity check value serving as the capability key.