ENDL
TEXAS

Date: 2 July 2007
To: T10 Technical Committee
From: Ralph O. Weber
Subject: Several OSD-2 Corrections and Clarifications

Introduction

The SNIA OSD TWG has identified several wording snafus in OSD-2 r01 and proposed the changes shown below.

Question to T10: Does multi-object need a glossary entry? Or, can the comment English usage suffice?

Revision History

r0	Initial revision
r1	Revised per requests from the SNIA OSD TWG (see change 2) and to address the comments from Rob Elliott shown below

Differences between r0 and r1 are identified by change bars.

Unless otherwise indicated additions are shown in blue, deletions in ~~red strikethrough~~, and comments in green.

Rob Elliott comments that motivated r1

[r0] 3.6: Change "shall be" to "are"

[r0] 3.6: per http://en.wikipedia.org/wiki/Two's_complement "twos-complement" should be "two’s complement"

[r0] Change 10 (see change 11): b) and c) ; s/b ,

Change 1 – Signed and unsigned integers

Problem Description

The SNIA OSD TWG has requested that the next OSD-2 revision provide for signed integer values.

Proposed changes in OSD-2 r01

3.5 Bit and byte ordering

…

3.6 Signed and unsigned integers values

Unless otherwise stated, all values defined by this standard are transferred as unsigned integers.

Signed integers are:

a) Positive valued unsigned integers when the most significant bit is set to zero; and
b) Negative valued integers in two's complement form (e.g., in a one-byte field, a negative valued one is represented as FFh) when the most significant bit is set to one.

3.7 ~~3.6~~ Notation conventions

…

Change 2 – Reduce the number of unused bytes in the Data-In Buffer and Data-Out Buffer

Problem Description

Since unused bytes in the Data-In Buffer and Data-Out Buffer are usually transferred (as per changes made in OSD-2 r01), aligning the segments on a minimum of 256-byte boundaries presents an overhead problem. Furthermore, most modern HBAs can scatter-gather memory with eight-byte alignments. Therefore, the buffer segment alignment exponent should be changed from an unsigned integer to a signed integer so that eight-byte alignments can be expressed.

Although this change is not 100% backwards compatible, it is close. Since all OSD service action codes have already been made obsolete in OSD-2 r01 and new ones assigned, the change need not be 100% backwards compatible.

Proposed changes in OSD-2 r01

4.12.5 Data-In and Data-Out buffer offsets

Offset fields (see table 34) in the CDB (e.g., the ~~retrieved attributes offset~~ RETRIEVED ATTRIBUTES OFFSET field described in 4.12.3 and the SET ATTRIBUTES LIST OFFSET field described in 4.12.4) specify the starting byte of segments of the Data-In Buffer or Data-Out Buffer other than the command data or parameter data segment.

Table 34 — CDB Data-In Buffer and Data-Out Buffer offset field format

Byte \ Bit	7	6	5	4	3	2	1	0
0	EXPONENT				(MSB) MANTISSA			
1	MANTISSA							
2	MANTISSA							
3	MANTISSA							(LSB)

The EXPONENT field specifies the signed integer (see 3.6) power of two to be used in computing the byte offset. The power of two shall be the value in the EXPONENT field plus eight. If the offset field does not contain FFFF FFFFh and the EXPONENT field contains -6, -7, or -8, then the command shall be terminated with a CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to INVALID FIELD IN CDB.

The MANTISSA field specifies the value to be multiplied by two raised to the power specified by the EXPONENT field.

The byte offset represented by a field having the format described in this subclause shall be:

$$\text{byte offset} = \text{mantissa} * (2^{(\text{exponent}+8)})$$

An offset field containing zero specifies a byte offset value of zero.

If the offset field for a Data-In Buffer or Data-Out Buffer segment that is not being used is not set to FFFF FFFFh, the results are unpredictable. {{Note added period at the end of this sentence.}}

Change 3 – Forced Collection Removal & Attributes Cleanup

Problem Description

In the REMOVE COLLECTION command, the definition of the FCR bit is ambiguous relative to the order in which operations shall be performed when the FCR bit is set to one. The ambiguity is such that implementations allow errors to leave dangling collection membership information.

Proposed changes in OSD-2 r01 6.21

The FCR (force collection removal) bit specifies the actions to be taken if the collection contains user objects. ~~If the FCR bit is set to one, the collection shall be removed even if it contains user objects and the attributes pages of the user objects in the collection shall be modified to indicate that the user object no longer is a member of the collection.~~ If the FCR bit is set to zero and the collection contains user objects, the command shall be terminated with a CHECK CONDITION status, the sense key shall be set to ILLEGAL REQUEST, and the additional sense code shall be set to PARTITION OR COLLECTION CONTAINS USER OBJECTS. If the FCR bit is set to one, the collection shall be removed as follows even if it contains user objects:

1) Each user object in the collection shall be modified to indicate that the user object no longer is a member of the collection; and
2) The collection shall be removed.

If the FCR bit is set to one, the REMOVE COLLECTION command shall not be completed with GOOD status until all user object members of the collection have been modified as described in this subclause and the collection has been removed.

Change 4 – Unexpected Multi-Object Command Termination

Problem Description

The current discussion of unexpected termination of multi-object OSD commands anticipates the effects of task management functions but not the effects of resets.

Proposed changes in OSD-2 r01 4.6.6.2

If a multi-object command is terminated as part of processing any task management function or as the result of a condition (e.g., logical unit reset) established in response to an event (see SAM-4), then the device server shall either:

a) Establish a consistent, stable state for each user object being processed; or
b) Set the policy access tag attribute in the User Object Policy/Security attributes page described in 4.9.3 for any user object for which it is not possible to establish consistent state.

Change 5 – Object ID Reuse & Quota Testing

Problems Description

The collections used by multi-object operations do not enjoy the benefits of back links (i.e., the user objects do not have entries in the Collections attributes page that indicate their membership in the special kind of collection used by multi-object operations). Therefore, it is not possible to delete entries from the multi-object-operation collection when the identified user object is deleted. As a result, the list of user objects in a multi-object-operation collection can become stale over time.

This issue needs to be addressed by requiring multi-object operations to ignore user objects in the collection whose creation date/time is greater than that of the multi-object-operation collection itself.

In addition to the reuse problem, the testing of quotas is not explicitly covered by the current requirements. Therefore, text in the quotas subclause applies and all quotas for all multI-object commands must be tested and reserved before any user object is processed. This is impossible to implement.

Appropriate modifications are needed to allow quotas to be tested on each user object in a multi-object command at the time that user object is processed.

Proposed changes in OSD-2 r01

4.6.6.2 Commands that use collections to affect multiple user objects

…

~~After the specified operations have been successfully completed on a user object, that user object shall be removed from the specified collection. As a result of this requirement, the following conditions apply:~~

~~a) After an error condition~~ …

Each user object in the specified collection shall be processed as follows:

1) If the creation time attribute in the User Object Timestamps attributes page (see 7.1.2.18) is earlier than or equal to (i.e., less than or equal to) the creation time attribute in the Collection Timestamps attributes page (see 7.1.2.17), then the quotas (see 4.8) that apply to the specified operation shall be evaluated and processing of the operation shall be handled as follows:
   A) If a quota error condition is detected the multi-object command shall be terminated as described in this subclause; or
   B) If no quota error condition is detected, the specified operation shall be performed on the user object and whether or not an error is detected shall be noted;
2) If the creation time attribute in the User Object Timestamps attributes page is later than (i.e., greater than) the creation time attribute in the Collection Timestamps attributes page, then the specified operation shall not be performed on the user object. This shall not be considered to be an error; and
3) If no error has been detected, the user object shall be removed from the specified collection.

As a result of these requirements, the following conditions apply:

a) After an error condition …

4.8.2 Quota errors

…

For multi-object commands, quota error processing shall be handled as described in 4.6.6.2. For individual objects, the ~~The~~ device server shall not terminate a command for quota errors after any user data or attributes have been modified.

4.8.3 Quota testing

…

Change 6 – Attributes List Length

Problem Description

The definition of the list length field in the attributes list header does not explain the requirements when the parameter list is truncated by allocation length.

Proposed changes in OSD-2 r01 7.1.3.1

The LIST LENGTH field indicates the number of bytes of attributes list entries that follow or the number of bytes that would follow if the parameter data were not truncated (see 5.2.2.3). For attributes lists sent from the application client to the device server, the LIST LENGTH field may contain zero.

Change 7 – Attributes returned by the LIST command with LIST_ATTR set to one

Problems Description

When listing the contents of the Root Object, the objects listed are partitions. However, the Root Object is also partition 0. The handling of this needs to be clarified with respect to the LIST parameter data and the attributes returned.

With the exception of attribute page FFFF FFFEh, requesting any information in attribute pages F000 0000h through FFFFF FFFFh should be an error in a LIST command and a LIST COLLECTION command.

Attributes page FFFF FFFFh is supposed to specify all attributes pages. This makes sense for a single object but is prohibitively complex for multiple objects.

Attributes pages F000 0000h through FFFF FFFDh are reserved.

Proposed changes in OSD-2 r01

6.14 LIST command

…

Table 69 — Specifying objects and attributes to be listed

PARTITION_ID field	LIST_ATTR field	Description
zero	zero	The Partition_IDs (see 4.6.4) in the root object shall be returned in the parameter data. Partition_ID zero shall not be returned in the parameter data, but any requested attributes shall be returned in the retrieved attributes segment of the Data-In Buffer (see 4.12.3).
	one	The Partition_IDs in the root object and attributes specified by the get and set attributes parameters (see 5.2.2) for each partition shall be returned in the parameter data. Partition_ID zero shall not be returned in the parameter data, but any requested attributes shall be returned in the retrieved attributes segment of the Data-In Buffer.
non-zero	…	…

…

Table 71 — Attributes processing requirements for LIST commands with the LIST_ATTR bit set to one

PARTITION_ID field	Attribute page number values	Description
zero	R+0h to R+2FFF FFFFh	The retrieved attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer (see 4.12.3) in the format defined by 7.1.3.
	P+0h to P+2FFF FFFFh	~~The retrieved attribute values shall be returned in the LIST command parameter data as defined in 6.14.2.~~ a) For Partition_ID zero, the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3; and b) For any Partition_ID other than zero, the retrieved attribute values shall be returned in the LIST command parameter data as defined in 6.14.2.
	FFFF FFFEh	The retrieved Current Command attributes page (see 7.1.2.24) attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	~~F000 0000h to FFFF FFFFh~~	~~a) If the object associated with the attribute is the root object, the attribute value shall be returned in the retrieved attributes segment of the Data-In Buffer; or~~ ~~b) If the object associated with the attribute is a partition, the attribute value shall be returned in the LIST command parameter data.~~
	~~0h to 2FFF FFFFh, and C+0h to C+2FFF FFFFh~~ All other values	The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.
non-zero	P+0h to P+2FFF FFFFh	The retrieved attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	0h to 2FFF FFFFh	The retrieved attribute values shall be returned in the LIST command parameter data as defined in 6.14.2.
	FFFF FFFEh	The retrieved Current Command attributes page attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	~~F000 0000h to FFFF FFFFh~~	~~a) If the object associated with the attribute is the partition specified in the LIST command, the attribute value shall be returned in the retrieved attributes segment of the Data-In Buffer; or~~ ~~b) If the object associated with the attribute is a user object, the attribute value shall be returned in the LIST command parameter data.~~
	~~R+0h to R+2FFF FFFFh, and C+0h to C+2FFF FFFFh~~ All other values	The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

6.15 LIST COLLECTION command

…

Table 82 — Attributes processing requirements for LIST COLLECTION commands with the LIST_ATTR bit set to one

COLLECTION_ OBJECT_ID field	Attribute page number values	Description
zero	P+0h to P+2FFF FFFFh	The retrieved attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer (see 4.12.3) in the format defined by 7.1.3.
	C+0h to C+2FFF FFFFh	The retrieved attribute values shall be returned in the LIST COLLECTION command parameter data as defined in 6.14.2.
	FFFF FFFEh	The retrieved Current Command attributes page (see 7.1.2.24) attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	~~F000 0000h to FFFF FFFFh~~	~~a) If the object associated with the attribute is a partition, the attribute value shall be returned in the retrieved attributes segment of the Data-In Buffer; or~~ ~~b) If the object associated with the attribute is a collection, the attribute value shall be returned in the LIST COLLECTION command parameter data.~~
	~~0h to 2FFF FFFFh, and R+0h to R+2FFF FFFFh~~ All other values	The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.
non-zero	C+0h to C+2FFF FFFFh	The retrieved attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	0h to 2FFF FFFFh	The retrieved attribute values shall be returned in the LIST COLLECTION command parameter data as defined in 6.14.2.
	FFFF FFFEh	The retrieved Current Command attributes page attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer in the format defined by 7.1.3.
	~~F000 0000h to FFFF FFFFh~~	~~a) If the object associated with the attribute is the collection specified in the LIST COLLECTION command, the attribute value shall be returned in the retrieved attributes segment of the Data-In Buffer; or~~ ~~b) If the object associated with the attribute is a user object, the attribute value shall be returned in the LIST COLLECTION command parameter data.~~
	~~R+0h to R+2FFF FFFFh, and P+0h to P+2FFF FFFFh~~ All other values	The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

Change 8 – Only GET/SET MEMBER ATTRIBUTES process member user object attributes

Problem Description

Table 4 allows any multi-object command to retrieve the attributes from collection member user objects. The design intention was that only the GET MEMBER ATTRIBUTES command and SET MEMBER ATTRIBUTES command be allowed to do this.

Also, Table 5 will need to be modified whenever new multi-object commands are defined.

Proposed changes in OSD-2 r01

4.6.6.2 Commands that use collections to affect multiple user objects

…

NOTE 2 The LIST COLLECTION command and LIST command are ~~is~~ not a multi-object commands ~~command~~.

…

If the CDB GET/SET CDBFMT field contains 11b (i.e., when list format attributes processing is specified), multi-object commands allow setting and retrieving of both collection attributes and user object attributes. The get and set attributes parameters are defined in 5.2.2.~~,~~ The ~~the~~ list format is defined in 7.1.3. Attribute retrieval and setting shall be processed as shown in table x1.~~, attribute page number values in the get attributes list shall be processed as shown in table 4, and attribute page numbers in the set attributes list shall be processed as shown in table 5.~~

Table x1 — Attributes retrieval and setting requirements for multi-object commands

Attribute page number values	Command	Description
C+0h to C+2FFF FFFFh and FFFF FFFEh	Any multi-object command	The attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer (see 4.12.3) as defined in 5.2.2.3 using list type Eh (see 7.1.3.5). The setting of attributes shall be processed as defined in 5.2.2.3.
0h to 2FFF FFFFh	GET MEMBER ATTRIBUTES or SET MEMBER ATTRIBUTES	The attribute values for every user object that is a member of the collection shall be returned in the retrieved attributes segment of the Data-In Buffer as defined in 5.2.2.3 using list type Eh (see 7.1.3.5). The setting of attributes shall be processed as defined in 5.2.2.3 and the same user object attribute values shall be set in every user object that is a member of the collection.
	All other multi-object commands	The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.
All other page number values	Any multi-object command	

{{Table 4 and table 5 are reproduced here for reference. They are to be replaced by table x1.}}

~~**Table 4 — Attributes retrieval requirements for multi-object commands**~~

~~**Attribute page number values**~~	~~**Command**~~	~~**Description**~~
~~C+0h to C+2FFF FFFFh and F000 0000h to FFFF FFFFh~~	~~Any multi-object command~~	~~The attribute values shall be returned in the retrieved attributes segment of the Data-In Buffer (see 4.12.3) as defined in 5.2.2.3 using list type Fh (see 7.1.3.4).~~
~~0h to 2FFF FFFFh~~		~~The attribute values for every user object that is a member of the collection shall be returned in the retrieved attributes segment of the Data-In Buffer as defined in 5.2.2.3 using list type Fh (see 7.1.3.4).~~
~~P+0h to P+2FFF FFFFh and R+0h to R+2FFF FFFFh~~		~~The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.~~

~~**Table 5 — Attributes setting requirements for multi-object commands**~~

~~**Attribute page number values**~~	~~**Command**~~	~~**Description**~~
~~C+0h to C+2FFF FFFFh and F000 0000h to FFFF FFFFh~~	~~Any multi-object command~~	~~The setting of attributes shall be processed as defined in 5.2.2.3.~~
~~0h to 2FFF FFFFh~~	~~GET MEMBER ATTRIBUTES or SET MEMBER ATTRIBUTES~~	~~The setting of attributes shall be processed as defined in 5.2.2.3 and the same user object attribute values shall be set in every user object that is a member of the collection.~~
	~~QUERY or REMOVE MEMBER OBJECTS~~	~~The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.~~
~~P+0h to P+2FFF FFFFh and R+0h to R+2FFF FFFFh~~	~~Any multi-object command~~	~~The command to be terminated with a CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.~~

Change 9 – GET MEMBER ATTRIBUTES and SET MEMBER ATTRIBUTES Behavior

Problem Description

The GET MEMBER ATTRIBUTES and SET MEMBER ATTRIBUTES process the same attributes (i.e., the same combinations of attribute page and attribute number) for all user objects in the specified collection, but this is not clearly stated in enough places.

Proposed changes in OSD-2 r01

4.13 GET MEMBER ATTRIBUTES

…

The get and set attributes parameters are defined in 5.2.2. The format of the Data-In Buffer and Data-Out Buffer when attributes are being retrieved or set is described in 4.12. Get and set attributes processing requirements specific to multi-object commands are defined in 4.6.6.2.

The same attributes (i.e., the same combinations of attribute page and attribute number) are retrieved and/or set for all user objects in the specified collection. If user object attributes are set, all such attributes are set to the same values in all user objects in the specified collection.

…

4.27 SET MEMBER ATTRIBUTES

…

The get and set attributes parameters are defined in 5.2.2. The format of the Data-In Buffer and Data-Out Buffer when attributes are being retrieved or set is described in 4.12. Get and set attributes processing requirements specific to multi-object commands are defined in 4.6.6.2.

The same attributes (i.e., the same combinations of attribute page and attribute number) are retrieved and/or set for all user objects in the specified collection. If user object attributes are set, all such attributes are set to the same values in all user objects in the specified collection.

…

7.1.3.2 List entry format for retrieving attributes for ~~the~~ a specified OSD object

The attributes list entry format shown in table 152 is used for specifying the attributes to be retrieved by a GET ATTRIBUTES command (see 6.12) or equivalent command function.

For the GET MEMBER ATTRIBUTES command (see 4.13) and the SET MEMBER ATTRIBUTES command (see 4.27), this list entry format specifies retrieval of the same attributes (i.e., the same combinations of attribute page and attribute number) for all user objects in the specified collection.

Table 152 — List entry format for retrieving attributes for ~~the~~ a specified OSD object
{{No changes in the contents of table 152.}}

…

{{No other changes in 7.1.3.2.}}

7.1.3.3 List entry format for retrieved attributes and for setting attributes for ~~the~~ a specified OSD object

The attributes list entry format shown in table 153 is used for returning the each attribute value to be retrieved by a GET ATTRIBUTES command (see 6.12) and for specifying each attribute value to be set by a SET ATTRIBUTES command (see 6.24) or equivalent command functions.

For the GET MEMBER ATTRIBUTES command (see 4.13) and the SET MEMBER ATTRIBUTES command (see 4.27), this list entry format specifies the same attributes values to set in all user objects in the specified collection. The retrieved attributes format for the GET MEMBER ATTRIBUTES command and SET MEMBER ATTRIBUTES command is defined in 7.1.3.5.

Table 153 — List entry format for retrieved attributes and for setting attributes for ~~the~~ a specified OSD object

{{No changes in the contents of table 153.}}

…

{{No other changes in 7.1.3.3.}}

Change 10 – Multi-objects attributes lists have a different format from the LIST command

Problems Description

7.1.3.1 has not been updated to accommodate the GET MEMBER ATTRIBUTES command and SET MEMBER ATTRIBUTES command. If it were updated, the list format would be different from the list format used by that used by the LIST command with the LIST_ATTR bit set to one.

Since the LIST command format is more space efficient than the current 7.1.3 attributes list format, the attributes list format is being made obsolete and the LIST command format inserted as its replacement for all OSD-1 and all new uses.

Also, the entries in the LIST command format are not eight-byte aligned. The new format proposed here is identical to the OSD-2 r01 LIST command format with the exception that the list entries are eight-byte aligned.

Proposed changes in OSD-2 r01

6.14.2 LIST command parameter data

…

The OBJECT DESCRIPTOR FORMAT field (see table 73) indicates the format of the object descriptors.

Table 73 — LIST command OBJECT DESCRIPTOR FORMAT field values

Code	Description	Reference
00h	The LIST command parameter data format shall be as specified in OSD (see 2.2).	
01h	The object descriptors are a list of Partition_IDs each of which is eight bytes and has the format shown in table 74.	6.14.3.1
02h	Each object descriptor ~~(see table 75)~~ has a multi-object retrieved attributes format (see table x2), and contains a Partition_ID followed by attribute parameters associated with the indicated partition.	~~6.14.3.2~~ 7.1.3.5
03h to 20h	Reserved	
21h	The object descriptors are a list of User_Object_IDs each of which is eight bytes and has the format shown in table 78.	6.14.3.5
22h	Each object descriptor ~~(see table 79)~~ has a multi-object retrieved attributes format (see table x2), and contains a User_Object_ID followed by attribute parameters associated with the indicated user object.	~~6.14.3.6~~ 7.1.3.5
23h to 3Fh	Reserved	

6.14.3 LIST command and LIST COLLECTION command object descriptor formats

…

~~**6.14.3.2 Partition_ID with partition attributes object descriptor format**~~

~~…~~

~~Each attributes list entry shall have the format shown in 7.1.3.3 and contain information about one attribute with an attribute page number between:~~

a) ~~P+0h and P+2FFF FFFFh inclusive; or~~
b) ~~F000 0000h and FFFF FFFEh inclusive.~~

{{All of 6.14.3.2 is to be removed. The particular text shown contains the only requirements that are unique to 6.14.3.2 and is provided for comparison with table 71. This proposal contends that the shown requirements can be removed because they replicate requirements defined by table 71 and modified by this proposal.}}

…

~~**6.14.3.4 Collection_Object_ID with collection attributes object descriptor format**~~

~~…~~

~~Each attributes list entry shall have the format shown in 7.1.3.3 and contains information about one attribute with an attribute page number between:~~

a) ~~C+0h and C+2FFF FFFFh inclusive; or~~
b) ~~F000 0000h and FFFF FFFEh inclusive.~~

{{All of 6.14.3.4 is to be removed. The particular text shown contains the only requirements that are unique to 6.14.3.4 and is provided for comparison with table 82. This proposal contends that the shown requirements can be removed because they replicate requirements defined by table 82 and modified by this proposal.}}

…

~~**6.14.3.6 User_Object_ID with user object attributes object descriptor format**~~

~~…~~

~~Each attributes list entry shall have the format shown in 7.1.3.3 and contain information about one attribute with an attribute page number between:~~

a) ~~0h and 2FFF FFFFh inclusive; or~~
b) ~~F000 0000h and FFFF FFFEh inclusive.~~

{{All of 6.14.3.6 is to be removed. The particular text shown contains the only requirements that are unique to 6.14.3.6 and is provided for comparison with table 71 and table 82. This proposal contends that the shown requirements can be removed because they replicate requirements defined by table 71 and table 82 and modified by this proposal.}}

6.15 LIST COLLECTION

…

The OBJECT DESCRIPTOR FORMAT field (see table 84) indicates the format of the object descriptors.

Table 84 — LIST COLLECTION command OBJECT DESCRIPTOR FORMAT field values

Code	Description	Reference
00h	The LIST COLLECTION command parameter data format shall be as specified in OSD (see 2.2).	
01h to 10h	Reserved	
11h	The object descriptors are a list of Collection_Object_IDs each of which is eight bytes and has the format shown in table 76.	6.14.3.3
12h	Each object descriptor ~~(see table 77)~~ has a multi-object retrieved attributes format (see table x2), and contains a Collection_Object_ID followed by attribute parameters associated with the indicated partition.	~~6.14.3.4~~ 7.1.3.5
13h to 20h	Reserved	
21h	The object descriptors are a list of User_Object_IDs each of which is eight bytes and has the format shown in table 78.	6.14.3.5
22h	Each object descriptor ~~(see table 79)~~ has a multi-object retrieved attributes format (see table x2), and contains a User_Object_ID followed by attribute parameters associated with the indicated user object.	~~6.14.3.6~~ 7.1.3.5
23h to 3Fh	Reserved	

…

7.1.3.1 Attributes lists overview

…

The LIST TYPE field (see table 151) specifies the format of all attributes list entries in the attributes list.

Table 151 — List type values

List Type	Description	Support Requirement	Reference	Allowed Use: Get Attributes List	Allowed Use: Get Attributes Response	Allowed Use: Set Attributes List
0h	Reserved			No	No	No
1h	Retrieve attributes for the specified OSD object or objects	Mandatory	7.1.3.2	Yes	No	No
2h - 8h	Reserved			No	No	No
9h	Retrieved/Set attributes for the specified OSD object	Mandatory	7.1.3.3	No	Yes	Yes
Ah - ~~Eh~~ Dh	Reserved			No	No	No
Eh	Retrieved attributes for more than one OSD object	Mandatory	7.1.3.5	No	Yes	No
Fh	Obsolete ~~Retrieved attributes for a CREATE command (see 6.3) that creates more than one user object~~	~~Mandatory~~	~~7.1.3.4~~	~~No~~	~~Yes~~	~~No~~

If list type 1h (see 7.1.3.2) is used to retrieve attributes for the specified OSD object, the list type of the list containing the retrieved objects shall be:

a) ~~Fh (see 7.1.3.4)~~ Eh (see 7.1.3.5) for: ~~a CREATE command that creates more than one user object; or~~
   A) A CREATE command that creates more than one user object;
   B) A GET MEMBER ATTRIBUTES command; or
   C) A SET MEMBER ATTRIBUTES command;
   or
b) 9h (see 7.1.3.3) for all other commands and for a CREATE command that creates only one user object.

…

~~7.1.3.4 List entry format for attributes retrieved by CREATE command that creates multiple user objects~~

{{Delete all of 7.1.3.4.}}

7.1.3.5 Multi-object retrieved attributes format

{{All of 7.1.3.5 is new. Text addition/deletion markups suspended for the remainder of 7.1.3.5.}}

The format shown in table x2 is used for indicating the attributes to be retrieved by:

a) A CREATE command (see 6.3) that creates more than one user object;
b) A GET MEMBER ATTRIBUTES command (see 6.13);

c) SET MEMBER ATTRIBUTES command (see 6.27);
d) A LIST command (see 6.14) with the LIST_ATTR bit set to one; and
e) A LIST COLLECTION command (see 6.15) with the LIST_ATTR bit set to one.

Table x2 — Multi-object retrieved attributes list format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			OBJECT ID				
7								(LSB)
8				OBJECT TYPE				
9				Reserved				
13								
14	(MSB)			ATTRIBUTES LIST LENGTH (n-15)				
15								(LSB)
				Attributes list entries				
16				Attributes list entry 0 (see 7.1.3.3)				
				⋮				
				Attributes list entry x (see 7.1.3.3)				
n								

The contents of the OBJECT ID field depend on the command that is retrieving the attributes as follows:

a) For a CREATE AND WRITE command, GET MEMBER ATTRIBUTES command, or SET MEMBER ATTRIBUTES command, the OBJECT ID field contains a User_Object_ID (see 4.6.5);
b) For a LIST command with the LIST_ATTR bit set to one, the OBJECT ID field contains a Partition_ID (see 4.6.4) or a User_Object_ID as described in 6.14.2; and
c) For a LIST COLLECTION command with the LIST_ATTR bit set to one, the OBJECT ID field contains a Collection_Object_ID (see 4.6.6) or a User_Object_ID as described in 6.15.

The OBJECT TYPE field indicates type of OSD object to which the attributes list entry applies using the code values shown in table 12 (see 4.9.2.2).

The ATTRIBUTES LIST LENGTH field indicates the number of bytes of attributes list entries that follow. If the parameter data is truncated due to insufficient allocation length, the ATTRIBUTES LIST LENGTH field shall not be altered to reflect the truncation (i.e., the attributes list length indicates the number of bytes that would follow if the allocation length had been infinite).

Each attributes list entry has the format shown in 7.1.3.3 and contain information about one attribute.

Change 11 – Partition and Root Default Security Method Selection

Problem Description

The default security method is selected based on the capability ALLOWED PARTITION_ID field. This is correct for user objects and collections, but incorrect for partitions and the root object.

Proposed changes in OSD-2 r01 4.10.3

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

1) …
4) Set the capability SECURITY METHOD field as follows:
   A) Select a security method other than the partition default:
      …
   B) Use the partition default:
      a) If the application client requested a credential to be used in a SET KEY command (see 6.25) or a SET MASTER KEY command (see 6.26), set the capability SECURITY METHOD field to the value in the default security method attribute in the Root Policy/Security attributes page (see 7.1.2.20);
      b) If the capability OBJECT TYPE field contains ROOT, set the capability SECURITY METHOD field to the value in the default security method attribute in the Root Policy/Security attributes page;
      c) If the capability OBJECT TYPE field contains PARTITION, set the capability SECURITY METHOD field to the value in the default security method attribute in the Partition Policy/Security attributes page (see 7.1.2.21) for partition zero (see 4.6.4);
      d) ~~b)~~ Otherwise, set the capability SECURITY METHOD field to the value in the default security method attribute in the Partition Policy/Security attributes page ~~(see 7.1.2.21)~~ for the partition whose Partition ID is contained in the capability ALLOWED PARTITION_ID field;