ENDL
TEXAS

Date: 22 September 2002
To: T10 Technical Committee & SNIA OSD Working Group
From: Ralph O. Weber
Subject: Restoring and updating security key management to OSD

Motivation

OSD revision 5 included a 'master key' and 'device key' as attributes of the root object. These attributes were lost in the attributes rewrite that was reflected in OSD revision 6. Since it was always agreed that more attributes would be added as the need for their definition was recognized, this proposal restores those two key attributes.

In addition, all OSD revisions since revision 3 have included a model subclause describing a security key hierarchy. This proposal instantiates the full hierarchy as root and group object attributes. Note that no OSD revision has described user object keys and thus there is no user object attribute page for keys in this proposal.

The final area addressed by this proposal is the nomenclature around security keys. For example, most OSD revisions make reference to a SET KEYS command but no OSD revision defines that command. This proposal eliminates the SET KEYS command, providing for keys to be set with the SET ATTRIBUTES command.

Revision Information

r0 Initial proposal prepared for review by the SNIA OSD group

Proposed OSD Changes

All PDF page, clause, and table numbers are with respect to OSD revision 6:

ftp://ftp.t10.org/t10/drafts/osd/osd-r06.pdf

In table 55, add two new attributes pages, while maintaining common page numbering for similar pages:

G+4h	Reserved
G+5h	Group Security Keys
R+5h	Root Security Keys

Update table 57 to reflect the new attributes pages:

G+5h	INCITS T10 Group Security Keys
R+5h	INCITS T10 Root Security Keys

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

Add new clauses that define the new attributes pages as follows:

7.1.2.16 Root Security Keys attributes page

The Root Security Keys attributes page (number R+5h) shall contain the attributes listed in table 1.

Table 1 — Root Security Keys attributes page contents

Attribute Number	Bytes	Attribute	May be set	OSD Provided
0h	40	Page identification	No	Yes
1h	16	Master key	Yes	Yes
2h	16	Device key	Yes	No
3h - FFFF FFFFh		Reserved	No	

The page identification attribute (number 0h) shall have the format described in 7.1.2.2 with the VENDOR IDENTIFICATION field containing INCITS and the ATTRIBUTE PAGE IDENTIFICATION field containing T10 Root Security Keys.

The master key attribute (number 1h) shall contain the OSD device master key value (see 4.4.5.2). The master key attribute shall be set when the OSD device is manufactured and the value of the master key shall be communicated to the purchaser of the OSD device in a vendor specific manner.

If a set attributes parameter list contains an entry specifying the master key attribute, the capabilities parameters in the CDB (see 5.1.2.1) shall have the following characteristics, in addition to allowing all other aspects of the service action specified by the CDB and parameter data:

a) The KEY IDENTIFIER field shall contain 0h (i.e., the capabilities are signed with the master key); and
b) The RTKYCH bit shall be set to one.

If either of these requirements are not met, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

The device key attribute (number 2h) shall contain the OSD device key value (see 4.4.5.2).

If a set attributes parameter list contains an entry specifying the device key attribute, the capabilities parameters in the CDB shall have the following characteristics, in addition to allowing all other aspects of the service action specified by the CDB and parameter data:

a) The KEY IDENTIFIER field shall contain 0h or 1h (i.e., the capabilities are signed with the master key or the device key); and
b) The RTKYCH bit shall be set to one.

If either of these requirements are not met, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

If a get attributes parameter list contains an entry specifying the master key or device key attribute, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If a set attributes parameter list contains an entry specifying the number of an attribute that table 1 states may not be set, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST. If a CDB set attributes field specifies the number of an attribute that table 1 states may not be set, the command shall be terminated with a

CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

No page format is specified for the Root Security Keys attributes page. If a CDB get or set attributes field specifies the page number of the Root Security Keys attributes page, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

7.1.2.17 Group Security Keys attributes page

The Group Security Keys attributes page (number G+5h) shall contain the attributes listed in table 2.

Table 2 — Group Security Keys attributes page contents

Attribute Number	Bytes	Attribute	May be set	OSD Provided
0h	40	Page identification	No	Yes
1h	16	Object group key	Yes	Yes
2h	16	Object group working key A	Yes	No
3h	16	Object group working key B	Yes	No
4h - FFFF FFFFh		Reserved	No	

The page identification attribute (number 0h) shall have the format described in 7.1.2.2 with the VENDOR IDENTIFICATION field containing INCITS and the ATTRIBUTE PAGE IDENTIFICATION field containing T10 Group Security Keys.

The object group key attribute (number 1h) shall contain the object group key value (see 4.4.5.2). The object group key value shall be set by a CREATE OBJECT GROUP service action (see 6.7).

If a set attributes parameter list contains an entry specifying the object group key attribute, the capabilities parameters in the CDB (see 5.1.2.1) shall have the following characteristics, in addition to allowing all other aspects of the service action specified by the CDB and parameter data:

a) The KEY IDENTIFIER field shall contain 1h or 2h (i.e., the capabilities are signed with the device key or the object group key); and
b) The GPKYCH bit shall be set to one.

If either of these requirements are not met, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

The object group working key A attribute (number 2h) shall contain the object group working key A value (see 4.4.5.2).

The object group working key B attribute (number 3h) shall contain the object group working key B value.

If a set attributes parameter list contains an entry specifying the object group working key A or object group working key B attribute, the capabilities parameters in the CDB shall have the following characteristics, in addition to allowing all other aspects of the service action specified by the CDB and parameter data:

a) The KEY IDENTIFIER field shall contain 2h (i.e., the capabilities are signed with the object group key); and
b) The GPKYABCH bit shall be set to one.

If either of these requirements are not met, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

If a get attributes parameter list contains an entry specifying the object group key, object group working key A, or object group working key B attribute, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

If a set attributes parameter list contains an entry specifying the number of an attribute that table 2 states may not be set, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN PARAMETER LIST. If a CDB set attributes field specifies the number of an attribute that table 2 states may not be set, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

No page format is specified for the Group Security Keys attributes page. If a CDB get or set attributes field specifies the page number of the Group Security Keys attributes page, the command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

Capabilities parameters changes

To allow capabilities parameters to restrict security changes new bit definitions are required. Also, it seems advisable to allow capabilities to restrict setting attributes and changes to provide this based on associated object type are also proposed here.

In capabilities parameters byte 128, define the follows bits (leaving any undefined bits reserved):

Bit 7: If the Set Root Attributes (STRTATTR) bit is set to one, the capabilities allow service actions to set attributes in pages associated with the root object. If the STRTATTR set to zero and a command requests the setting of an attribute in a page associated with the root object, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

Bit 6: If the Set Group Attributes (STGPATTR) bit is set to one, the capabilities allow service actions to set attributes in pages associated with an object group. If the STGPATTR set to zero and a command requests the setting of an attribute in a page associated with an object group, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

Bit 5: If the Set User Attributes (STUATTR) bit is set to one, the capabilities allow service actions to set attributes in pages associated with a user object. If the STUATTR set to zero and a command requests the setting of an attribute in a page associated with a user object, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

Bit 3: If the Root Key Changes (RTKYCH) bit is set to one, the capabilities allow service actions to set attributes in the Root Security Keys attributes page (see 7.1.2.16). If the RTKYCH set to zero and a command requests the setting of an attribute in the Root Security Keys attributes page, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

Bit 2: If the Group Key Changes (GPKYCH) bit is set to one, the capabilities allow service actions to set the object group key attribute in the Group Security Keys attributes page (see 7.1.2.16). If the GPKYCH set to zero and a command requests the setting of an attribute in the object group key attribute in the Group Security Keys attributes page, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

Bit 1: If the Group working Key A/B Changes (GPKYABCH) bit is set to one, the capabilities allow service actions to set the object group working key A and object group working key B attributes in the Group Security Keys attributes page. If the GPKYABCH set to zero and a command requests the setting of an attribute in the object group working

key A and object group working key B attributes in the Group Security Keys attributes page, the command shall be terminated with a CHECK CONDITION status and the sense key shall be set to ILLEGAL REQUEST.

GET/SET ATTRIBUTES changes

Any service action has the potential to attempt retrieval of security key values, meaning that the best place to specify the prohibition against doing so is in the attribute page descriptions as is done above. The above specified requirements mean that singling out the GET ATTRIBUTES service action for statement of the requirement is not useful. So in the GET ATTRIBUTES service action definition, delete the following sentence:

> Security keys shall not be returned by the GET ATTRIBUTES service action.

For similar reasons, remove the following sentences from the SET ATTRIBUTES service action definition:

> Security keys (see 5.3.2.1) are the only attributes able to be set via the SET ATTRIBUTES service action that the GET ATTRIBUTES service action is prohibited from returning.

Modest changes are required in the security model section to address:

a) The fact that there is no SET KEYS service action;
b) The reliance on how the capabilities are signed for limiting key changes; and
c) Other changes in key mechanics contained in this proposal.

For convenient review, the entire security model clause is included here with additions in red text and deletions in red strike through text.

4.4.5 Security

4.4.5.1 Security Introduction

A complete and useful OSD environment should:

a) Authenticate application client identities,
b) Test each requested operation against the environment's database of authorized transformations,
c) Ensure that the integrity of each operation is not tampered with during transmission,
d) Protect the privacy of data, access patterns and application clients,
e) Identify the source of inappropriate activity, and
f) Survive inappropriate activity with minimal degradation of service qualities.

Proper operation of some of these security features requires support from the SCSI protocol and ~~as such~~ is beyond the scope of this standard.

Security is the confident state of mind of users of a OSD environment that exhibits compelling instances of these properties. While it is possible that application clients can be trusted to accurately state their identities and authorizations, and it is possible for the service deliver subsystem and other initiators to be trusted not to interfere with the integrity and privacy of all issued requests so that inappropriate activity never occurs, this standard is designed not to depend upon such high degrees of trustworthiness.

To avoid specialization to any particular file system or server application, OSD security mechanisms are expected to enforce a wide range of possible authentication and access policies whose details may not be determined until OSD systems are available to the software designers, users and administrators of a specific application system. Thus, the details of access control policy as well as user authentication and authorization are beyond the scope of this standard.

OSD security defines ~~encryption mechanisms enabling OSDs to protect privacy and~~ a digital signature mechanisms enabling OSDs to enforce authentication, authorization, integrity, auditing and isolation policies set by customer decisions through ~~p~~Policy/~~s~~Storage managers without unnecessary limitation on the flexibility of those policies.

4.4.5.2 OSD security key hierarchy

A basic theme in cryptographic security is the controlled sharing of secrets. Fundamentally, to perform an OSD access, an application client demonstrates that either it shares a secret with the OSD device, or that the access was previously authorized by a Policy/Storage Manager who shares a secret with the OSD device. Client level key management is a critical security function, but since it is inherently a system wide function, it is properly a function of a higher level than the storage system. Client level keys should be distributed privately to application clients by Policy/Storage Managers, valid for a lifetime that is inversely proportional to their frequency of use, and protected against disclosure by a device's owner and users.

Policy/Storage Managers, some owning OSDs and allocating them to other Policy/Storage Managers (e.g. ~~for example~~, a backup system may own devices and allocate them to file systems or database systems), need to share secrets with OSDs. To protect against any specific key being overused and exposed to yet-to-be-created key discovery algorithms, keys should have a primary and secondary version. Secondary keys should more commonly used than primary keys and should be changed frequently while primary keys should be rarely used to enable their lifetime to be much longer.

~~Since OSD resources are shared among non-cooperating Policy/Storage Managers by granting each an object group which behaves like a virtual OSD, each object group needs its own set of keys. Finally, to avoid the simultaneous revocation of all signatures made by a working (secondary) key that is due to change, working keys come in pairs, both valid, but only one being used to generate new signatures. This way old signatures are not necessarily revoked by the first working key change; until the second working key change these signatures may still be valid.~~

~~Editors Note 1 - ROW: This appears to be saying that the group object has been overloaded with a security function that more properly should be defined as a fourth object type.~~

The hierarchy of security keys is shown in table 3.

Table 3 — OSD security key hierarchy

Name	Description
Master key	OSD owner's key~~; preferably used only at installation~~
Device key	Frequently changed online proxy for owner's key
Object group key	Less frequently changed online per object group key
Object group working key A	Frequently changing, commonly used per object group key A
Object group working key B	Frequently changing, commonly used per object group key B

Initially an OSD device has a single key preinstalled. This is known as the master key. The master key may only be modified by a command whose capabilities are signed ~~encrypted~~ with the previous value of the master key. While an OSD has no control over the environment into which its owner modifies its master keys, it is recommended that this key only be changed when the OSD device is attached to a physically secured private network without any gateway to application client systems (i.e., on a two-port network on the desk of an administrator).

The device key is set infrequently using capabilities that are signed with the master key or the previous value of the device key. This key exists to provide a more frequently used variant of the master key. With this key, the master key need only be used in operations that modify this key. This helps protect the master key from attack.

Each object group has a unique object group key. An object group key is set at object group creation time, and is modified using capabilities that are signed with the device key or the previous value of the object group key.

~~In this manner~~ Because each object group has a unique object group key, two file system managers, database managers, or Policy/Storage managers utilizing different ~~OGs~~ object groups on the same OSD may be provided ~~separate~~ different object group keys, and thus be totally denied access to one anothers' ~~OGs~~ object groups.

Each object group also has two object group working keys, A and B, that are set using capabilities that are signed with the object group key. ~~As with the master and device keys, t~~The working keys are intended to limit the use of the object group key.

Most operations that require the use of a key in the key hierarchy should use object group working keys.~~, and~~ Policy/Storage Managers should change the object group working keys regularly (e.g., daily). Two object group working keys are provided in each object group because OSD capabilities, authenticated by a working key signature, are reusable over an extended (e.g., 12 hour) time period. By having a second object group working key to use in future capability generation, the first of these object group working keys may be changed without invalidating capabilities signed with the other. This allows capabilities generated shortly before a working key is changed to continue to work until they gracefully timeout ~~according to their expiration time~~.

4.4.5.3 Digital signatures

No changes are proposed in this clause.

4.4.5.4 OSD capabilities

Application clients, except for Policy/Storage Managers, should never possess one of the keys in the OSD key hierarchy. Instead, they should be issued individual capabilities by Policy/Storage Managers that are digitally signed (see 4.4.5.3) using an appropriate key from the hierarchy (see 4.4.5.2). A capability describes the accesses that application clients are allowed to perform by proving that they have been given that capability.

Capabilities are validated by means of a digital signature ~~(see 4.4.5.2)~~. A central feature of this OSD capability system is that these ~~signed hashes~~ digital signatures may themselves be treated as a key, provided that the Policy/Storage Manager has distributed them securely. By using the ~~signed hash~~ digital signature distributed to it as a secret in the computation of a derived ~~signed hash~~ digital signature, an application client is capable of proving to an OSD device that it holds the first ~~signed hash~~ digital signature. Hence we call these ~~signed hash~~ digital signature capability key-signatures to emphasize their roles as keys. A signed capability is the union of a capability and its capability key-signature.

The information contained in a capability (e.g., ~~which~~ what OSD service actions are permitted by the capability) is detailed in 5.1.2.1.1.