Date: February 23, 2000

To: T10 Committee (SCSI)

From: Jim Hafner (IBM) (hafner@almaden.ibm.com)

Subject: A Detailed Proposal For Access Controls

ABSTRACT:

A SAN (storage area network) is a network environment where multiple hosts machines (clients/initiators) have access to a collection of storage devices (targets). Unless there is significant collaboration between the initiators, it is desirable in this environment, to partition, fence or otherwise restrict access to some storage devices by different hosts. The current SAN protocols (either at the transport layer or in the SCSI layer) are not well-suited to this purpose.

In this proposal, we detail new SCSI commands and target actions to implement access control management. Two new commands are proposed which allow configuration (Data-Out) and reporting (Data-In) of access control management functions at the target. The new commands and actions are not restricted to storage devices but are applicable to any target.

Earlier revisions reflected comments, questions and suggestions from folks at LSI Logic, Sun Microsystems, Adaptec, Compaq and others at IBM.

This revision is based on extensive discussions between Ralph Weber (ENDL Technologies), David Chambliss (IBM) and the author. The general framework of the model is joint work. But the author is responsible for the content of this document, particularly for inconsistencies, incompleteness, errors, or blunders.

In particular, this last draft attempts to merge the requirements presented in 00-123r0 (for what has been dubbed LUN Mapping) and the additional features of 99-245r4.

This revision has no specific provision for access controls on subcomponents.

1.0 Introduction

A SAN (storage area network) is a network environment where multiple hosts machines (clients/initiators) have access to a collection of storage devices (targets). Unless there is significant real-time collaboration between all the initiators, it is desirable in this environment, to partition, fence or otherwise restrict access to some storage devices by different hosts. The current SAN protocols are not well-suited to this purpose of access control management.

In our view, access controls should have the following properties:

a) they should be enforced at the target;

b) they should be granted to a host (i.e., at the OS-image or virtual machine level) and not to particular initiators (or ports or HBAs) within a host if at all possible;

c) they should be configured by some application client which is responsible for overseeing access controls over the entire SAN;

d) a configuration of access controls should not be associated with the particular initiator from which the configuration command was sent.

The last three points imply that SCSI reservations are inadequate to the task unless there is a single (real-time) application client coordinating reservations for *all* initiators in the SAN simultaneously. Such an application in a complex, multi-OS, multi-initiator environment would be expensive and difficult to manage.

To enable the protection required for access to devices in a simpler and easier to manage way, we propose a new SCSI-based protocol for access controls. This protocol is independent of the transport layer and is suited for any SAN environment whose higher level protocol is SCSI (e.g., FCP).

A general scenario is the following. A client application (what we call the Partition Access Manager or PAM[1]) has knowledge of all the initiators and target devices on the SAN. PAM can instruct a given target device to restrict access to some or all of its logical units by all initiators except those from some small set. Such a set might be a single host. Within the set, data integrity, locking, etc., is coordinated by existing protocols (like reservations) via a separate application client operating within the scope of this group. One might say that such a set is a “shared access group”. Hosts outside this group are denied (most) access to the device. In particular, these hosts can not preempt a reservation, issue read/write commands and the like. (Provisions for quality of service or resource allocations within a “shared access group” are outside the scope of this proposal.)

Note the following features of this scenario. PAM need only have one in-band communication channel to the target devices. PAM does not need to have any active presence on all the initiators, because the configuration commands are initiator independent. Furthermore, access restrictions are enforced at the target devices. This means that new hosts added to the SAN have no access to restricted targets unless expressly added by PAM. Also, hosts need have no special application client running in order to “fence” them from target devices to which they should not access or to gain access to devices to which they have been granted access.

The proposal can be applicable to any kind of target, not just storage devices. Resource requirements at the target can vary so that even limited function devices such as disk drives themselves might be capable of implementing these functions. However, it is more likely that larger devices such as controllers, devices with an embedded controller, medium changers, intelligent bridges (e.g., FC to SCSI) and the like would implement these functions.

There are two new commands with different service actions proposed. There is a Data-In command to query various status information of the target with respect to access control functions and a Data-Out com-

1.PAM is not part of the proposed standard, nor is it necessarily a real application. Mainly it is a pseudonym for the management application overseeing access controls for the SAN. It can be instantiated by a real application or instantiated more generally by the use of the defined protocol by users.

mand to configure different kinds of access controls. These are detailed in later sections. However, use of configuration commands are limited with respect to application clients or initiators. Initiators with access to a device have the right to issue proxy rights to other third party initiators without PAM's direct intervention. On the other hand, PAM's configuration tools (MANAGE ACL service action) can only be used by an application client (namely PAM) which shares a key with the target. This key identifies PAM as the originator of the command independent of which initiator she uses for command delivery. The key is maintained as part of the access control information of the target and can be preserved through power cycles.

Override of the target's key (in the unlikely event that PAM forgets it) is not specified in this proposal. It is a subject of much debate and details about how such a mechanism might work require further study. Revision 4 of this document contained a template for such an override function. Within the template, vendors would be free to implement alternative methods. Other vendor specific methods such as jumpers, vendor-specific commands (which might include firmware download), etc. are beyond the scope of this proposal.

Hosts (or OS-images) can be identified by a new AccessID as defined in this proposal. The reasons for the new identifier are the following. First, the new AccessID is transport independent and so is applicable to all current and future transport protocols. Second, (as noted above) access rights are naturally associated with the host machine (or virtual machine), not the individual initiators (ports/HBAs) on that machine. Transport layer identifiers, either transitory (e.g., FC N_Port) or persistent world wide identifiers (e.g., FC World Wide Nodename) are cumbersome and inadequate. Because they are bound to the given HBA within a machine, they are portable. This would require PAM to maintain continual knowledge of host hardware configuration simply to manage access rights. However, for additional function, the design contains provisions for transport-layer as well as vendor-specific identifiers.

The intent of the AccessID is to assign a permanent identifier to a given host machine (actually OS-image) without regard to the number of ports/HBAs on that host or any actions which change the hardware configuration of the machine. This makes management by PAM of the target's access controls much simpler. But it also implies requirements on the part of target to maintain associations between the AccessID and a given hosts initiator port or ports. These requirements are similar to but in some cases less restrictive than those already required by reservations.

For Fibre Channel, the use of Process Associators allows multiple virtual machines to share the same hardware connection to the fabric. From the point of view of the target, however, each N_PortID/Process Associator pair appears as a separate SCSI initiator (in our understanding). Consequently, AccessIDs can enable finer grained access management than what is available by use of persistent transport identifiers such as WWNs. They don't require management by PAM of the specific assignment of Process Associators at the fabric layer and so further simplify PAM's job.

Though AccessIDs create a new identifier name space that PAM must manage, it is our opinion that the gains in simplicity, stability and transport independence outweigh this concern.

What follows this main section is a detailed description of the new commands and target requirements and constitutes the normative part of the proposal. Section 2.0 discusses the model, raises some design questions and issues and documents the revision history. Section 3.0 proposed changes to the glossary and acronyms clauses. Section 4.0 is proposed as an additional sub-clause in the model clause of SPC-2.

AUTHOR'S NOTE: *AUTHOR'S NOTEs are intended to generate small questions and expose small issues for possible further action. Ideally, later revisions of this document will have these issues addressed and the notes removed. In any case, they should not be included in the final editorial changes included in SPC-x. Larger issues are listed in the next section.*

2.0 Additional major issues or questions[1]

2.1 The new model and open issues

There are significant changes to the basic access controls model introduced in this draft. We summarize the new model in more detail in what follows, but begin by highlighting a few of the major differences with the old model.

The earlier drafts all shared the “visible but inaccessible” approach to access denial. That is: all logical units are visible to all initiators (meaning that they are discoverable under INQUIRY and REPORT LUNS) but non-privileged initiators are denied service (in particular I/O service) by specific CHECK CONDITIONS. This new draft replaces the “visible but inaccessible” model with what has been dubbed LUN Mapping. This has two features. First, it “hides” logical units which are not accessible to a given initiator. So, to such an initiator INQUIRY to some logical units will report “no device present” and REPORT LUNS will only show a set of LUN values representing a subset of the complete set of logical units on the target device. The second feature of LUN Mapping is that the LUN values reported in REPORT LUNs are initiator-specific. That is, for each initiator a given LUN will only be a pointer to an specific logical unit and that pointer is a function of the initiator. Thus, the same (shared) logical unit may be addressed by one initiator at LUN1 and by another initiator at LUN2. A consequence is that LUN values are no longer global addresses for specific logical units within a target. This complicates a number of shared functions (such as third party copy operations). This “mapping” function however is seen as a functional requirement.

Because of the changes required for LUN Mapping, the proxy model has changed significantly. This is discussed in more detail in 2.1.2.

The new model begins with the following assumptions and requirements:

a) the target knows (better than PAM) what the LUN Map should look like for a given initiator, particularly, if the map can change dynamically because of initiator AccessID enrollment;

b) some hosts can’t handle gaps in the LUN list (this requirement may not be as hard as we first suspected);

c) host resources for never-accessible logical units should not be wasted;

d) some hosts require access to a specific logical unit at LUN0 (for boot?);

e) facilitate third-party operations in a simple way (proxy);

f) need an interlock with hosts to assist them with enrollment requirements;

g) LUN Map changes should be minimized and managed so that data integrity is protected and host efforts to recover from LUN Map changes are minimized.

With these in mind, the model has the following characteristics.

First, the target owns the actual mapping of logical units to LUN values. PAM tells the target to grant access to a particular logical unit (in other words, PAM instructs the target to create a LUN Map entry) but doesn’t specify exactly what the corresponding LUN value will be. The only exception to this rule is that PAM can instruct that a particular logical unit map to LUN0 for a specific initiator identified only by TransportID.

An initiator can be identified to the target by either a TransportID (available at connection time, say, FC-login) or by AccessID (available only after an enrollment action by the initiator). Since the latter can only come after the connection, the target will always map TransportID-accessible logical units before mapping AccessID-accessible logical units. Furthermore, an initiator (such as a copy manager) may get access (have its LUN Map changed) through proxy functions. These are always numbered after (if applicable) AccessID enrollment.

1.This section is primarily to discuss the model, raise discussion points and to log changes to the various revisions; it is not part of the proposed standard.

We require the target to build its LUN Map for a specific initiator as shown in Table 1.

TABLE 1. LUN Mapping model.

LUN value	Reason
0	LU access granted by TransporID, specifically mapped to LUN0
0 m	LU access granted by TransportID, no specific LUN0 requirement
m+1 n	LU access granted by AccessID after enrollment
>n	LU access granted via proxy

The order of logical units in the mapping to LUN values respects the iLUN (internal) ordering, with the exception of the LUN0-specific request. This guarantees no gaps in the mapping (except perhaps in the proxy range) and also guarantees that the map is consistent after target resets.

The TransportID and AccessID ranges have the following properties:

a) LUNs in these ranges normally appear during system (host) startup and remain unchanged from one boot to the next;

b) more than one LUN can appear in either of these ranges as a result of a single action on the part of the initiator (e.g., PLOGI or enrollment);

c) automated loss of access (e.g., LIP) can be corrected by repeating the PLOGI or enrollment.

The Proxy range differs as follows:

a) LUNs in this range do not appear normally during system startup and may change dynamically during the life of the system;

b) exactly one new LUN appears in this region as the result of a single action by the initiator (ASSIGN PROXY LUN service action);

c) automated loss of access will not be corrected by the same mechanisms that work for TransportIDs and AccessIDs.

d) multiple LUN values assigned under proxy may reference the same logical unit.

Note that a logical unit may be addressable by a given initiator at a single LUN value in the TransportID or AccessID range and/or via multiple LUN values in the proxy range.

The TransportID portion of a LUN Map for an initiator after a target reset will be restored. The AccessID portion may not (in an implementation-specific manner) be restored. But such an initiator will easily detect the failure, recognize the potential reasons and take the necessary action (enroll) to restore access.

We assume three possible states for an initiator with respect to a target:

a) “enrolled” - in this state, LUN Map contains entries for logical units to which the enrolled AccessID has been granted access;

b) “de-enrolled” - in this state, which is the transition state from enrolled because of automated loss of access (e.g., LIP) or PAM-initiated event (e.g., Flush in MANAGE ACL service action), the AccessID accessible logical units stay in the LUN Map, but are “inaccessible” with status and sense indicating that the enrollment was invalidated;

c) “not-enrolled” - in this state, the target has no association of an AccessID for the initiator; the initiator’s LUN map contains only references to logical units for that initiator’s TransportID.

Simply, the first enrollment action by an initiator appends to the LUN Map entries for the enrolled AccessID and puts the initiator in the “enrolled” state. Events (outside of the initiator’s control) may cause the initiator to go into the de-enrolled state. The LUN Map doesn't change, but commands to the affected logical units are failed with sense data sufficient to trigger the initiator to re-enroll. This would put the initiator back into the enrolled state. The initiator switches between these two states under normal operation.

An initiator can go into the not-enrolled state under two events. The first is by its own actions (CANCEL ENROLLMENT). This should be used by an initiator prior to shutdown so the target can free up enrollment resources. This is not required, however. The second event is an action by PAM which causes a change in the LUN Map for that initiator. Such events are (or should be) rare but they have the potential to adversely affect the host and its data. This scenario is described in 2.1.1.

We propose the existence of a new entity in a device (more precisely, in an SMU). We call this the "access controls coordinator"[1]. This entity is the repository of the access controls data, the coordinator of access rights, the handler of enrollments and the builder of LUN maps. This entity spans multiple logical units as well as spans all the ports in a multi-ported device. In effect, it creates an interface between the ports and the logical units for the purpose of restricting access, and managing the LUN Maps for initiators. This entity is addressable only (or "should" be only) via LUN0. [We've allowed for access control commands to go to any LUN, to facilitate the difficulties of host/OSs which do LUN offsetting or LUN mapping internally.]

2.1.1 Host/target/PAM interlock for LUN Map changes

Actions by PAM to the ACLs for a given TransportID or AccessID may cause a change in the LUN Map for some initiators. If not coordinated carefully with the affected initiators, this can have undesirable effects on the user data. E.g., if a host has a string of I/Os enqueued (at the host) addressed to a LUN value and intended for a particular logical unit, and PAM changes the LUN Map so that this LUN value no longer addresses the same logical unit, the enqueued I/Os will (without a specific interlock) go to the wrong logical unit, thereby both corrupting the data on the newly addressed logical unit and the hosts view of that data consistency.

Note that this issue arises only if a LUN value "moves" to a new logical unit; not if the LUN value is "added" to the LUN Map or if the LUN value is "deleted" from the LUN Map. For the added case, there would be no active IOs, even though the initiator will need to take some action to "discover" the new logical unit. For the "deleted" case, all IOs will fail with "logical unit not supported", and no data is transferred.

To address this problem, we first postulate that PAM will never take this sort of action unless she has sufficient knowledge that such risks to data integrity have been minimized. This might mean making sure that the affected hosts are shutdown (or quiesced) and that sufficient rediscovery of the LUN Map by the host will correct the problem.

Second, we note that this heavy interlock between PAM and the host is actually only required for that portion of the LUN Map in the TransportID range. The AccessID range can be handled by relying on the enrollment process itself (see below). TransportIDs will (should) only be used in a couple of cases. First, if the host does not participate in the access controls protocol (e.g., a legacy system). In this case, PAM will need a direct interlock with the host. Second, for perhaps a very limited range of logical units (e.g., for boot devices) for hosts which do participate in the protocol; most of their LUN Map should be in the AccessID range. In this case, there is less likelihood of an action by PAM affecting the LUN Map, so less need for direct PAM/host interlock.

For the AccessID range, we propose the following host/target interlock. If PAM initiates a change to the LUN Map for a given AccessID, the target immediately places that host in the not-enrolled state; all formerly addressable logical units become "not supported". This error message will be sufficient notification to the host that something dramatic has happened (either the map changed or the target reset). In either case, the host should suspend IO, re-enroll and redrive the LUN-discovery process (at the affected target) to clean up its internal references to logical units.

1.Alternatives to "coordinator" are "enforcer" or "manager" or "enforcement manager". Enforcer is an unpleasant word and manager might be confused with PAM, so we chose coordinator.

2.1.2 The proxy model

In the previous revision, an initiator would grant a third party (based on an identifier for that third party) proxy access to a logical unit to which the granting initiator already had access. This had a couple of limitations (e.g., a copy manager given proxy access couldn't farm out part of its job to another party). In the presence of LUN Mapping, however, the problem is more pronounced in that there is no longer global addressing of logical units (by LUN). Consequently, if initiator A wanted initiator B to have access, A would need to grant proxy right (at the target), somehow find out what LUN got generated for that logical unit in B's LUN Map and use this value when requesting services from B (e.g., in EXTENDED COPY target descriptors). In this revision, we change the proxy model significantly. An initiator, instead of granting access to a third party by initiator identifier, requests a proxy token from the target for a specific logical unit, passes the token to the third party, and that third party uses the token to request a LUN value for that logical unit.

This has the following advantages:

a) global addressing of logical units by LUN is no longer required in the presence of a LUN Map;
b) proxy tokens can be forwarded from one third party to another;
c) multiple proxy tokens can be used for the same logical unit;
d) each token held by a third party can be used to assign a separate LUN value (for the same logical unit); this allows the third party like a copy manager to separate the required tasks by LUN);
e) separation of proxies between initiators sharing a given logical unit; e.g., if initiator A and B have forwarded proxy tokens for a shared logical unit, they can invalidate their own tokens without affecting the other initiator.

We have proposed changes to the EXTENDED COPY target descriptors to include Proxy Tokens as handles or references to logical units.

We propose that proxy tokens can be invalidated by an initiator which knows the proxy token, by any initiator with PAM-granted access to the logical unit (to invalidate all proxy tokens for a given logical unit) or by PAM either by individual proxy token or for all proxy tokens at a target.

In the current proposal, we've mapped out two variants of a protocol for the third party to get a LUN value for a Proxy Token. The preferred method is that the third party initiator specifies a preferred LUN value when requesting the access to the logical unit (that is, the initiator asks "can I have LUN=x for Proxy Token=t?"). The alternative is the third party initiator requests the target generate a LUN value for the token (that is, "what LUN value can I have for Proxy Token=t?").

The preferred method is probably required under alternative design point of 2.1.3.1 (where PAM owns the LUN Map). However, it can open up the "twenty questions" problem. To alleviate this, we've specified that the target include an alternative LUN value embedded in the sense data of the failed request. See 6.1.7.

The alternative method is documented in the clauses below as "obsolete" and is included only for the record to elicit opinion that this choice is better.

2.1.3 Alternative design points

It is possible that some of the assumptions and requirements of the model described above are either not valid, not sufficient, or have such limited applicability that they can be safely ignored. In particular, it possible that our assumption that the "target knows best" is incorrect and that only PAM can know best.

2.1.3.1 LUN Map owner

If "PAM knows best" is a more accurate assumption (or requirement), then we would be open to modifying the current proposal to give PAM complete ownership of the LUN Map. Unfortunately, this has two consequences:

a) It is possible for PAM to specify one LUN Map for a TransportID and one for an AccessID which conflict (e.g., specific mapping of a LUN value to two different logical units or for two LUN values to point to the same logical unit). This “error” state cannot necessarily be detected by the target at configuration time (when processing the parameter data for the command) but only at enrollment of the affected AccessID and initiator. This means that some rules will be required which the target must use for conflict resolution. We might also need some additional mechanism for PAM to determine if conflict resolution was instantiated by the target.
b) It is unlikely in this case that PAM can maintain the property of no gaps in LUN Maps. This is particularly the case when mixing AccessID and TransportID maps. This no-gap assumption/requirement may not hold and so may not be a significant point of conflict.

On the other hand, this approach does lend itself more readily to PAM being able to reduce the likelihood of LUN values changing the logical unit they address. It is easier for PAM to add and delete entries and not disrupt existing values.

2.1.3.2 INQUIRY bit

It has been suggested that we take a field or two from the standard INQUIRY data to indicate one or more of the following:

a) there is an access control coordinator at this logical unit (that is, ACCESS CONTROL IN/OUT commands can be handled if addressed to this logical unit);
b) this logical unit is accessible because of explicit rights afforded that initiator (that is, this logical unit is in fact subject to access controls and the requesting initiator is privileged).

For now, we leave this on the table as a point of discussion, and make no specific recommendations.

2.1.3.3 Host/Target LUN Map change interlock

Is there a need for a tighter host/target interlock when the LUN Map changes for the TransportID range? One alternative is that the target put a CHECK CONDITION status on the affected (or all) logical units with sense of ACCESS DENIED - LUN MAP CHANGED. These would stay in place until some explicit action is taken by the initiator, e.g., CLEAR LUN MAP CHANGED CONDITION.

Another alternative is that the target refuses to implement a MANAGE ACL command from PAM if that would affect an initiator which is currently “connected” (e.g., logged in at the FC layer). This would mean that the host is probably shutdown (or otherwise completely disconnected from the target) and any change to its LUN Map won’t be a problem.

In any of these scenarios, should PAM be allowed (via some specific bit or field in parameter data) to override any interlock rule that the target would normally implement? PAM might use this bit if she’s sure there won’t be any problems with the affected initiators.

2.2 Override of Management Identifier Key

We’ve left the issue of how PAM (or any one) can override the Management Identifier Key of the access controls coordinator (in the unlikely event that PAM forgets it) to further discussion. In revision 4 and 5, we’ve included a vendor-specific flag in the DISABLE ACCESS CONTROLS service action of the ACCESS CONTROL OUT command. This would then be a standardized template for this purpose. It would allow implementors flexibility in choosing a methodology (e.g., just set the VS bit, or set the VS bit and send some specific parameter data) which met their specific implementation limitations and requirements

There was some resistance to this approach. As this issue is not yet resolved, we’ve left this design point unchanged but we are open to specific proposals which meet the various (conflicting) requests and requirements.

2.2.1 “Override” design alternatives

There are a number of design points floating around, each attempting to address specific requirements. We list a few here.

The simplest is an override command/service action with no validation.

The next choice is the one in this document, namely a template for a vendor-specific implementation of the override mechanism.

There is the “state machine” version. In this, the override function can only be instantiated by the appropriate command (without the Key) if the device is in a specific state, e.g., if the command is the first command received by the target after a target-reset.

Next, the command can only work if accompanied by “private” data which is only available to initiators with some access to the device. Suggested private data is the unit serial number which would be made private by suppressing this data in INQUIRY response to non-privileged initiators.

2.3 Access controls on sublogical unit entities

Revision 4 and 5 have no notion of access control granularity below the logical unit level, though earlier revisions did. There may be a reason in the future to extend access controls to sublogical unit entities. In one context (medium changer) this might be elements. In the up-and-coming Object-based Storage Device model, this might be for access controls on Object Groups (this would provide some simple access controls without the need for complex encryption and authentication protocols - admittedly, this is not a complete solution to the long term objectives of OSDs, but might provide an interim solution).

It would be possible to extend the current model to allow such finer grained controls. The fundamental question however is what direction that can take. There are two alternatives:

a) Access grant to a logical unit (either by PAM or by proxy token) grants access only to the higher level entity and not any addressable subentity, unless specified otherwise.
b) Access grant to a logical unit (either by PAM or by proxy token) grants complete access to all sublogical unit entities, unless specified otherwise.

In the first case, we would need service actions and/or parameter structures to extend rights to sublogical unit entities. For example, give initiator A and B access to the logical unit, but they can’t use “elements” X or Y within the logical unit. Then we expressly extend A’s rights to X and B’s rights to Y.

In the second case, we need service actions and/or parameter structures to limit rights. For example, give A and B access to the logical unit and they both can use X and Y. Then we expressly restrict A’s access to only X and B’s to only Y.

Revision 3 had a different, somewhat intermediate, model. In that version, explicit grant to a full logical unit gave rights to all subentities (similar to case two above); explicit grant to a subentity was limited to that subentity and the top level entity (similar to case one above). This doesn’t quite work in this case, mostly because such semantics don’t fit (in the author’s opinion) very well with LUN Mapping syntax. The proposed options above separate the LUN Mapping service action/parameter data syntax from the subentity extend or restrict syntax.

For example, suppose we adopt case two. Initiator A has access to some or all of a logical unit and wants B to do some copy services for it, but limited to only a subset of A’s access. A requests a proxy token from the target which is initially valid for the entire context of A’s rights. A follows that request with a specific request to the target to limit the validity of the proxy token to a specific subset of its accessible subentities. A then forwards the proxy token to B with the assurance that B can’t get access to that part of the logical unit not accessible to A and not included in the subset scoped by the proxy. (Similar syntax can be used by PAM, to first map a LUN value for a given logical unit and then restrict the range of validity of that access.)

The author has no particular preference for either model at this time.

2.4 Changes from previous revisions

2.4.1 Changes from revision 2

A TransportID is defined for SPI devices.

The language concerning the effects of changes on access controls to commands already in the task manager has been clarified and simplified. It is modeled on the language from PERSISTENT RESERVATIONS.

There is a new OUT service action, RESET AC, which provides a Management Identifier Key validated reset function and a template for vendor-specific reset functions (which might provide an override mechanism for the Key).

There are additions to Table 8 of SPC-2 defining the device server's actions in the presence of reservations when access control commands are issued.

The table of new ASC/ASCQs for access controls has been updated with specific values, consistent with the proposal 99-314r1.

The model for access controls on elements (or more generally on subcomponents of the logical unit) has be significantly redone. There is a definition for an "access controllable component (ACC)" and revised specification of an initiator's access rights when granted access only to such a component. Also, there is now only one ACL at the device server (not one per ACC and the logical unit) and so there is only one ACL Enabled or Disabled state for the device. This was done both to simplify the model (and hopefully clarify it) and to enable a simple evolution to the next revision where this model is deleted (so access controls are only defined at the full logical unit). At the moment there is no driving force behind having this finer granularity, which is why it will be excised from the next revision. We are archiving this revision with the revised model in the event that a future need arises for access controls at a granularity below the full logical unit. For example, the model described here could work with elements of medium changers as originally expected or it could be applicable to access controls on object-groups as might be defined in the Object-based Storage Device proposal currently under review (99-315r0).

This change in the "element" model removes a certain functionality, namely, of disabling access controls on specific ACCs within the device while still maintaining access controls at the full logical unit. There are two possible approaches to this. One is to have a configuration command which can change the classification of a subcomponent from an ACC to a non-ACC component (though this might be hard to define carefully). A second approach is to define a "universal AccessID" which all initiators are automatically enrolled under (so a sort of wild-card AccessID). Granting access to this universal AccessID would be functionally equivalent to disabling access controls at the specific ACC.

Other wording changes of an editorial nature are included here as well.

2.4.2 Changes for revision 4

The model for access controls on subcomponents of the logical unit has been removed.

2.4.3 Changes for revision 5

This is a major revision, and so has substantial changes.

There have been some name changes. E.g., Manage ACL Key is now called the Management Identifier Key. The RESET AC service action has been renamed to DISABLE ACCESS CONTROLS. Many service actions have been changed, added or removed. The parameter data for some commands has changed as well. The set of required ASC/ASCQs has also changed. These changes are detailed in the sections which follow.

There is an additional section (Appendix XXX) which proposes changes to the EXTENDED COPY command's target descriptors to accommodate proxy tokens.

The SCSI Address field in TransportID for SPI-x has a reference to the glossary for this name.

The TransportID for FCP-x has been changed. N_PortID has been removed and the fields rearranged a little. The use N_PortID was only required for proxy purposes. With the changes to that protocol, there is no further need for this addressing field.

We've removed from MANAGE ACL the ability to set the device back to the factory default unconstrained state. This is now only available in the DISABLE ACCESS CONTROLS service action.

PTPL (Persist Through Power-Loss) is now required. Our current feeling is that vendors will implement this feature in all cases, so having this optional only complicated the model unnecessarily.

3.0 Glossary and Acronyms

The following additions to the glossary and acronyms section of SPC-x are proposed.

AUTHOR'S NOTE: *it is quite possible that not all these entries need to be in the glossary. They might be better suited for the Access Controls model clause instead.*

3.1 Glossary

Access Controls: An optional feature that restricts initiator access to specific logical units and the information about logical units sent to initiators in the parameter data of INQUIRY and REPORT LUNS commands. (See 5.x.).

Access Control List: The data used by a target to provide access controls for initiators.

Access Controls Coordinator: The entity within an SMU which coordinates the management and enforcement of access controls for all logical units within the device. This is always addressable through LUN0.

AccessID: An identifier used for granting or revoking access rights to a logical unit. An initiator may enroll an AccessID with the ACCESS CONTROL OUT command and ACCESS ID ENROLL service action so that the device server is able to determine the access rights for that initiator.

enrolled: The state an initiator enters as a consequence of a successful completion of an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action.

de-enrolled: The state an initiator enters from the enrolled state as a consequence of certain events in the service delivery subsystem or by certain access control management actions (e.g., ACCESS CONTROL OUT with MANAGE ACL service action).

not-enrolled: The state of an initiator not in either the enrolled or de-enrolled state. This is the default state of an initiator in the absence of any ACCESS CONTROL OUT command with ACCESS ID ENROLL service action from that initiator. This is also the state after successful completion of the ACCESS CONTROL OUT command with ENROLLMENT CANCEL service action.

Internal Logical Unit Number: Used in conjunction with the Access Controls feature (see 3.1.xx) to identify logical units before they are mapped to logical unit numbers.

TransportID: A protocol or interconnect-defined identifier used for granting or revoking access rights to a logical unit.

LUN Map: an initiator-specific list of pairs consisting of a LUN value and an iLUN value together with its associated logical unit. The list of LUN values shall be the same as that returned by the REPORT LUNS command.

3.2 Acronyms

ACL: Access Control List

iLUN: Internal Logical Unit NUmber

4.0 Access Controls

Access controls are an optional target feature that application clients may use to allow only specified initiators or groups of initiators to access specified logical units. Access controls are handled at the target by an access controls coordinator. The access controls coordinator maps a logical unit to different logical unit numbers depending on which initiator accesses the device. Access to a logical unit affects whether the logical unit appears in the parameter data returned by a REPORT LUNS command and how the logical unit responds to INQUIRY commands. An initiator can be identified uniquely by an access identifier, called an AccessID, as defined in 4.3 or by a protocol-specific identifier, called a TransportID, as defined in the relevant protocol standards.

AUTHOR'S NOTE: *See Annexes A-D for the changes required in other standards documents.*

An application client may add or remove restrictions on an initiator using access control commands.

The methods of managing access controls are identified by the commands:

a) ACCESS CONTROL IN - used to query the access control information; and
b) ACCESS CONTROL OUT - used to create, change or revoke access controls.

The access control management commands are not subject to reservation conflicts.

AUTHOR'S NOTE: *See Annex D for the changes required to Table 8 of SPC-2 (rev 14) with respect to reservation conflicts.*

If a target supports the access control commands, the access controls coordinator shall be able to maintain at least one entry in its ACL for each of its logical unit. In this way, each logical unit can be dedicated to at least one initiator and so restrict access to competing initiators. The default ACL is empty.

Included in the ACL data, the access controls coordinator shall maintain a Management Identifier Key of 8 bytes (64 bit integer). The default value for the Management Identifier Key is zero.

The access controls model provides three states for an initiator with respect to a specific target. The states and key features of each state are as follows:

a) not-enrolled: the default state for an initiator and the state entered into by the initiator in response to an ACCESS CONTROL OUT command with CANCEL ENROLLMENT service action;
b) enrolled: the state an initiator enters as a consequence of a successful ACCESS CONTROL OUT command with ACCESS ID ENROLL service action;
c) de-enrolled: the state an initiator enters as a consequence of specific events in the service delivery subsystem or by specific service actions (and parameter settings) in the ACCESS CONTROL OUT command.

4.3.1 describes each state in detail and the mechanisms that produce transitions between the states.

4.1 LUN Mapping

The access controls model specifies that the access controls coordinator modify the device servers' response to INQUIRY commands to a logical unit and to REPORT LUNS commands in initiator-specific ways. The access controls coordinator maps LUN values to its logical units in a manner dependent on the requesting initiator and the ACL data. This feature is called LUN Mapping. The access controls coordinator creates a LUN Map for an initiator subject to the following rules:

1. LUN0 is mapped to a specific accessible logical unit, as specified in the ACL data according to a TransportID identifier of that initiator;
2. LUN values are assigned sequentially to according to the order of the iLUNs for those logical units accessible to an initiator via a TransportID or an enrolled AccessID, subject to the first condition;

3. LUN values are assigned to logical units accessible by virtue of a TransportID identifier of the initiator;
4. LUN values are assigned to those additional logical units (not already assigned a LUN value) accessible via an AccessID enrolled by an initiator (in either the enrolled or de-enrolled state);
5. LUN values are assigned beyond the last value previously assigned under TransportID or AccessID and independent of the order of the iLUNs to logical units accessible through a Proxy Token, by specific request of the initiator.

Note that an initiator's LUN Map may have multiple LUN values referencing the same logical unit; there will be at most one such LUN value assigned via TransportID or AccessID, but multiple values may be assigned under proxy tokens.

The parameter data returned in response to an INQUIRY command addressed to a LUN which is not mapped to an accessible logical unit shall set the Peripheral Device Type to 1Fh and Peripheral Qualifier to 011b (the device server is not capable of supporting a device at this logical unit).

The parameter data returned in response to a REPORT LUNS command addressed to LUN0 will return only the list of LUN values which are mapped to accessible logical units. If the initiator is in the enrolled or de-enrolled state, this list includes any LUN values mapped to logical units accessible by virtue of the AccessID enrolled by that initiator. If the initiator is in the not-enrolled state, then no such LUN values are included in the parameter data. If the initiator has access to any logical units by virtue of proxy tokens, the corresponding LUN values are included in the parameter data.

4.2 Establishment of Access Controls and other tasks

The time at which access controls are established or revoked with respect to other tasks being managed by the device server is vendor specific. Successful completion of an access control command (MANAGE ACL) indicates that a new access control state is established. Changes in the access control state may cause a change in the LUN Map for a specific initiator. Such changes shall not apply to any tasks queued before completion of the access control command; these commands shall be handled by the task manager of the logical unit to which they were addressed according to the LUN Map in place at the time the tasks were queued. The new LUN Map shall apply to all tasks received by the device server after successful completion of the access control command by the access controls coordinator. The execution of any access control command shall be performed as a single indivisible event.

Multiple access control commands (both ACCESS CONTROL IN and ACCESS CONTROL OUT) may be queued at the same time. The order of execution of such commands is defined by the tagged queuing restrictions, if any, but each is executed as a single indivisible command without any interleaving of actions that may be required by other access control commands.

4.2.1 Existing reservations and LUN Map changes

If a logical unit is reserved by one initiator and that logical unit is added to another initiator's LUN Map by any access control command, there shall be no changes in the reservation state of that logical unit.

If a logical unit is reserved by an initiator and that logical unit is removed from that initiator's LUN Map by any access control command or other event, there shall be no changes in the reservation. Existing mechanisms in the RESERVE/RELEASE and Persistent Reservations allow for other initiators with access to that logical unit to clear the reservation.

AUTHOR'S NOTE: *Is the above the right approach?*

4.3 Identifying initiators

Access rights are granted or revoked on the basis of either a TransportID (as defined in the protocol or interconnect standard) or an AccessID. An AccessID is enrolled with the access controls coordinator by an ACCESS ID ENROLL service action.

Access to logical units granted on the basis of a TransportID (and the related portion of the LUN Map) shall not be affected by events in the service delivery subsystem which might change non-persistent address identifiers for the initiator.

> NOTE: The requirement here is similar to that in the following requirement from FCP-2, rev 02, 5.3): “the relationship between address identifier of the initiator and a persistent reservation for a logical unit can be adjusted as defined in SPC-2 during those reconfiguration events that may change the S_ID of the initiator”. In other words, if the non-persistent service delivery subsystem address of an initiator changes because of events in the subsystem, the target is required to maintain the LUN Map and access rights of that initiator as determined by TransportID identifiers of that initiator.

> **AUTHOR’S NOTE**: *The above note is probably more for reviewers in this context but perhaps the sentiment of the note should be added to FCP-2 explicitly or at least to the addendum in this document.*

4.3.1 Initiator enrollment states

Initiators may enroll an AccessID with an access controls coordinator in order to gain access to logical units accessible via such an identifier. There are three states that an initiator can be in with respect to such an enrollment: not-enrolled, enrolled, de-enrolled. Each of these states and the mechanisms that cause transitions between these states are detailed in the following clauses.

4.3.1.1 Not-enrolled state

This is the default state for an initiator. An initiator stays in this state until it successfully completes the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action. See 4.3.1.2.

An initiator in the enrolled or de-enrolled state can transition to the not-enrolled state as follows:

a) by successful completion of the ACCESS CONTROL OUT command with CANCEL ENROLLMENT service action;

b) as a consequence of a change in the LUN Map effected by successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action from any initiator.

> NOTE: this requirement provides only a simple interlock to assist an initiator in detecting a change in its LUN Map which might be caused by events or actions not taken by that initiator. The use of the ACCESS CONTROL OUT with MANAGE ACL service action by the managing application client should be coordinated with the affected initiators to ensure proper data integrity. Such coordination is beyond the scope of this standard.

When in the not-enrolled state, the LUN Map for an initiator shall only contain LUN values which reference logical units accessible to that initiator via a TransportID or via proxy tokens.

4.3.1.2 Enrolled state

The initiator enters the enrolled state from either the not-enrolled or de-enrolled state by successful completion of the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action. This service action is successful only under the following conditions:

a) the initiator was in the not-enrolled state and the AccessID in the parameter data of the service action matches entries in the ACL data (so that this AccessID has rights to one or more logical units); in this case, the LUN Map is modified according to the rules of 4.1 for the logical units accessible via the enrolled AccessID;

b) the initiator was in the enrolled or de-enrolled state and the AccessID in the parameter data matches that of the current enrolled AccessID for that initiator; in this case, no changes to the LUN Map are effected, however commands to the affected logical units are handled according to the rules of 4.7.

The AccessID enrollment of an initiator (in either the enrolled or not-enrolled state) may be kept in non-volatile memory in an implementation dependent manner subject to the rules in 4.5.

Transitions out of the enrolled state are described in the subclauses for the not-enrolled and de-enrolled states.

> NOTE: This standard does not preclude implicit enrollments through mechanisms in the service delivery subsystem.

An initiator in the not-enrolled state may have entries in its LUN Map based on proxy tokens (that is, created by ASSIGN PROXY LUN service action) which would be affected by an enrollment. In this case, the LUN Map entries for the proxy tokens are implicitly released (as if the initiator sent the RELEASE PROXY LUN service action) and the LUN Map is then modified according to the rules of 4.1.

4.3.1.3 De-enrolled state

An initiator shall enter the de-enrolled state only from the enrolled state, and as a consequence of the following:

a) any event in the service delivery subsystem which causes the access controls coordinator to question whether the AccessID of an initiator in the enrolled state has changed (e.g., a PRLO or LOGO in FCP);

b) successful completion of an ACCESS CONTROLS OUT command (from any initiator) with MANAGE ACL service action and FLUSH bit set to one;

c) optionally after a target reset, as described in 4.5.

While in the de-enrolled state, the LUN Map for the initiator does not change; that is, the AccessID portion of the LUN Map is as if the initiator were in the enrolled state. However, commands to the affected logical units are handled according to the rules of 4.7.

4.3.2 Identifier Type and Initiator Identifier

Initiators are identified in parameter data with an IDENTIFIER TYPE code and INITIATOR IDENTIFIER field as defined in Table 2.

TABLE 2. IDENTIFIER TYPE and INITIATOR IDENTIFIER values.

Code	Description	Length
00h	AccessID	16
01h	TransportID	transport-specific
02h-7Fh	Reserved	n/a
80h-FFh	Vendor-specific	VS

Use of the TransportID is protocol and interconnect-specific. Each SCSI protocol standard shall specify the description of the TransportID structure.

Additionally, there is provision for vendor-specific initiator identification types.

AUTHOR'S NOTE: *The AccessID Length was chosen to allow an IPv6 style address to be used as an AccessID (this is NOT required). It was suggested that a longer or variable length AccessID be used to allow implementors a richer and more user-friendly name space. The decision here for fixed length of AccessIDs (what goes over the wire) still leaves open the user-interface side behavior. E.g., an implementation could simply create a pairing of user-friendly names and (say) a hash of that to 16 bytes for the over-the-wire AccessID. In other words, it seemed simpler to push this burden onto the application user-interface and not on the target.*

AUTHOR'S NOTE: *The author has no fundamental objection to fixing the length of all Initiator Identifiers to the largest necessary. The harder part is determining what that length is with enough forethought not to run into extensibility problems later.*

4.4 Granting and revoking access rights

The MANGE ACL service action of the ACCESS CONTROL OUT command is used to grant or revoke access to one or more logical units to an initiator based on the AccessID identifier or TransportID identifier. This same service action can selectively or universally revoke all proxy rights.

An initiators can gain access to a logical unit also via proxies. See 4.4.1.

An initiator's access right to a logical unit is the logical 'or' of all rights granted under both MANAGE ACL for any identifier which corresponds to that initiator and under proxy. For example, an initiator may have rights granted under MANAGE ACL action under both its enrolled AccessID and TransportID. Similarly, it may have multiple proxy rights granted under proxy tokens. Revocation of that initiator's access rights occurs only when all such access rights have been revoked; this means removal of all entries in the LUN Map for that initiator which referenced the logical unit.

When an initiator's access rights to a logical unit are changed (granted or revoked), the rules of 4.2 shall apply with respect to all commands from that initiator.

4.4.1 Proxy tokens and proxy access

An initiator with access to a logical unit on the basis of either a TransportID or AccessID may allow a third party temporary access to the same logical unit via the proxy mechanism.

The initiator requests from the access controls coordinator a Proxy Token for a specific logical unit via the ACCESS CONTROL IN command with REQUEST PROXY TOKEN service action. The access controls coordinator generates this Proxy Token in an implementation specific manner; it is recommended that all active Proxy Tokens for a specific logical unit be unique.

The initiator can then forward the Proxy Token to a third party (e.g., in a target descriptor in the parameter data of the EXTENDED COPY command; see Appendix D.2).

The third party uses the Proxy Token to request an entry be created in its LUN Map for the referenced logical unit via the ACCESS CONTROLS OUT command with ASSIGN PROXY LUN service action.

As long as the Proxy Token is valid, this entry in the third party's LUN Map is valid.

A Proxy Token shall be invalidated under the following events:

a) an initiator with access to the logical unit revokes the Proxy Token by the ACCESS CONTROL OUT command with the REVOKE PROXY TOKEN service action or with the REVOKE ALL PROXY TOKENS service action;

b) an application client issues the ACCESS CONTROL OUT command with MANAGE ACL service action and appropriate Revoke Proxy Token or Revoke All Proxy Tokens parameter pages;

c) a target reset or power-cycle.

A proxy LUN value for the third party (entry in the LUN Map based on a Proxy Token) shall be valid unless one of the following occurs:

a) the third party releases the LUN value with the ACCESS CONTROL OUT command and RELEASE PROXY LUN service action;

b) the Proxy Token is invalidated, as above;

c) an event in the service delivery subsystem causes the access controls coordinator to question whether the third party initiator that created the LUN value has changed (and may no longer be in possession of the Proxy Token).

In the latter two cases, the third party can reissue the ASSIGN PROXY LUN in an attempt to re-establish its proxy rights. In case (b), the access controls coordinator shall fail the request.

4.5 Preserving access control information (power-cycles and target resets)

AUTHOR'S NOTE: *we are requiring that the ACL data and state be preserved through power cycle (PTPL) in this draft. This design point is open for discussion, however.*

The access controls coordinator is required to maintain in non-volatile form the entire ACL data and state established by the MANAGE ACL service action of the ACCESS CONTROL OUT command.

If the device's non-volatile memory is not ready (to read the ACL data), the device server shall return on all addressed logical units a CHECK CONDITION status, a sense key of NOT READY and additional sense data as defined in the TEST UNIT READY command (see SPC-2, rev 14, 7.27) for all commands except INQUIRY.

All Proxy Tokens are revoked by a power cycle or target reset.

It is implementation specific what effects a power cycle or target reset may have on the AccessID enrollment and enrolled state of initiators. An access controls coordinator may preserve the enrollment; if it does, the following holds after the reset is complete:

a) all initiators formerly in the enrolled or de-enrolled state enter the de-enrolled state until changed by an initiator ACCESS ID ENROLL service action;

b) all entries in an initiator's LUN Map corresponding to an enrolled AccessID are restored.

If the access controls coordinator does not preserve enrollment across a power cycle, the following holds after the reset is complete:

a) all initiators enter the not-enrolled state until changed by an initiator ACCESS ID ENROLL service action;

b) all entries in an initiator's LUN Map shall correspond only to access rights granted under a TransportID.

4.6 Reporting access control information

There are two ways to request a report from the access controls coordinator about its ACL data.

A service action (REPORT ACL) shall report all ACL data for the access controls coordinator independent of the requesting initiator.

In this case, the information includes the following:

a) the list of initiator identifiers and their access rights currently in effect;

b) the list of proxies currently in effect.

The REPORT LUN MAP service action may be used by an initiator to get summary information about its access rights.

4.7 Verifying access rights for initiators

When the access controls coordinator has access controls enabled, access rights from a given initiator are validated in the following manner.

If the initiator has access to a logical unit by virtue of a TransportID or a valid proxy token, then all commands are handled in the normal fashion.

If the initiator has access to a logical unit by virtue of an AccessID enrolled by that initiator and the initiator is in the enrolled state, then all commands are handled in the normal fashion.

If the initiator has access to a logical unit by virtue of an AccessID enrolled by that initiator and the initiator is in the de-enrolled state, then commands are handled according to the following:

a) INQUIRY, ACCESS CONTROL OUT and ACCESS CONTROL IN are handled in the normal fashion;
b) all other commands receive a CHECK CONDITION status with sense key ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INITIATOR DE-ENROLLED and no data is transferred.

This last case may cause the initiator to issue the ACCESS ID ENROLL service action and then retry the failed command.

In all other cases, the commands are rejected with CHECK CONDITION status, ILLEGAL REQUEST and ASC/ASCQ set to LOGICAL UNIT NOT SUPPORTED.

4.8 Access Control Service Actions

Table 3 gives a summary list of the access control service actions.

TABLE 3. Access Control Commands and Service Actions

Code	Name	Type	Section
ACCESS CONTROL IN (OPCODE 86h)			
00h	REPORT ACL	M	5.1.1
01h	REPORT LU DESCRIPTIONS	M	5.1.2
02h	REPORT LUN MAP	O	5.1.3
03h	REQUEST PROXY TOKEN	O	5.1.4
04h	OPEN PROXY LUN	OBS	5.1.5
05h-0Fh	Reserved		
10h-1Fh	Vendor-specific	V	
ACCESS CONTROL OUT (OPCODE 87h)			
00h	MANAGE ACL	M	6.1.1
01h	DISABLE ACCESS CONTROLS	M	6.1.2
02h	ACCESS ID ENROLL	M	6.1.3
03h	CANCEL ENROLLMENT	M	6.1.4
04h	REVOKE PROXY TOKEN	O	6.1.5
05h	REVOKE ALL PROXY TOKENS	O	6.1.6
06h	ASSIGN PROXY LUN	O	6.1.7
07h	RELEASE PROXY LUN	O	6.1.8
08h	CLOSE PROXY LUN	OBS	6.1.9
08h-0Fh	Reserved		
10h-1Fh	Vendor-specific	V	

AUTHOR'S NOTE: *we have included two "Obsolete" service actions here (OPEN PROXY LUN and CLOSE PROXY LUN). See 2.1.2.*

4.9 Access Control Additional Sense Codes

Table 4 contains a list of the Additional Sense Code and Additional Sense Code Qualifiers relevant to access controls.

TABLE 4. Access Control Additional Sense Codes and Qualifiers

ASC	ASCQ	Name	Function
20h	01h	ACCESS DENIED - ENROLLMENT CONFLICT	An initiator in the enrolled or de-enrolled state issues an ACCESS ID ENROLL service action under a different AccessID.
20h	02h	ACCESS DENIED - INITIATOR DE-ENROLLED	An initiator in the de-enrolled state sends a restricted command to a logical unit accessible under the enrolled AccessID.
20h	03h	ACCESS DENIED - NO ACCESS RIGHTS	An initiator in the not-enrolled state sends an ACCESS ID ENROLL service action and the given AccessID has no access rights in the ACL data.
20h	04h	ACCESS DENIED - INVALID MGMT ID KEY	The Management Identifier Key value is does not match the value maintained by the access controls coordinator.
20h	05h	ACCESS DENIED - INVALID LU IDENTIFIER	The LUN or iLUN does not correspond to an accessible logical unit.
20h	05h	ACCESS DENIED - INVALID PROXY TOKEN	The Proxy Token is not valid; it does not correspond to a logical unit.
55h	05h	INSUFFICIENT ACCESS CONTROL RESOURCES	The access controls coordinator has exhausted its resources for the requested access controls action.

5.0 ACCESS CONTROL IN command

The ACCESS CONTROL IN command (see Table 5) is used to obtain information about the access controls that are active within the access controls coordinator and to facilitate proxy functions. The command shall be used in conjunction with the ACCESS CONTROL OUT command. It shall not be affected by reservations, persistent reservations or access controls.

This command should only be sent to LUN0 (in the SAM-2 hierarchical addressing scheme) and processed by the access controls coordinator. It should be rejected by the device server if addressed to any other LUN with CHECK CONDITION status, sense key of ILLEGAL REQUEST and additional sense code of INVALID OPCODE.

TABLE 5. ACCESS CONTROL IN command

Byte	Bit 7	6	5	4	3	2	1	0
0	OPERATION CODE (86h)							
1	RESERVED			SERVICE ACTION				
2	MSB							
9		SERVICE ACTION-SPECIFIC DATA						LSB
10	MSB							
13		ALLOCATION LENGTH						LSB
14	RESERVED							ILUN
15	CONTROL							

The actual length of the ACCESS CONTROL IN parameter list is available in a parameter list field. The ALLOCATION LENGTH field in the CDB indicates how much space has been reserved for the returned parameter list.

The SERVICE ACTION-SPECIFIC DATA field and ILUN field are described in the appropriate subclause for each service action.

The ALLOCATION LENGTH shall conform to the requirements of clause 4.2.5 (of SPC-2 revision 15) unless otherwise specified in the clause for a service action.

5.1 ACCESS CONTROL IN Service Actions

5.1.1 REPORT ACL service action (Mandatory)

The REPORT ACL service action of the ACCESS CONTROL IN command is used by an application client to query the complete ACL data currently maintained by the access controls coordinator.

If the access controls coordinator is in the default state where there is no ACL data and no Management Identifier Key, the device server shall respond with GOOD status and return no data, regardless of the value of any other field in the CDB.

If the access controls coordinator has a non-empty ACL data (including an established Management Identifier Key), the SERVICE ACTION-SPECIFIC DATA field in the CDB shall contain the current Management Identifier Key maintained by the access controls coordinator. If this is not the case, the device server shall return no data and respond with CHECK CONDITION, sense key ILLEGAL REQUEST, additional sense data set to ACCESS DENIED - INVALID MGMT ID KEY.

The ILUN field is reserved for this service action.

The ALLOCATION LENGTH shall be at least four (4), sufficient for the header information. If the Allocation Length is less than four (4), then the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST and additional sense code of INVALID FIELD IN CDB.

The format of the returned data shall conform to the specification in 5.1.1.1.

5.1.1.1 REPORT ACL parameter data format

The format of the parameter data provided in response to an ACCESS CONTROL IN command with REPORT ACL service actions is shown in Table 6. The ACL ENTRY PAGE(S) are described in 5.1.1.1.1 and 5.1.1.1.2.

TABLE 6. REPORT ACL parameter data format

Byte	Bit 7	6	5	4	3	2	1	0
0 3	MSB ADDITIONAL LENGTH (*n*-3)							LSB
3 *n*	ACL ENTRY PAGE(S)							

The ADDITIONAL LENGTH field shall contain a count of the number of bytes in the remaining parameter data. The value in this field shall contain the actual number of bytes available without consideration for insufficient ALLOCATION LENGTH in the requesting CDB.

The ACL ENTRY PAGE(S) shall contain a description of the ACL maintained by the access controls coordinator. Each ACL Entry Page is identified by a Page Code. The list of Page Codes and their definitions is given in Table 7 and the detailed description of the pages are in subsequent subclauses.

TABLE 7. ACL Entry PAGE CODE definitions for REPORT ACL service action

Page Code	Description	Clause
`00h`	Granted	5.1.1.1.1
`01h`	Granted All	5.1.1.1.1
`02h`	Proxy Token	5.1.1.1.2
`03b-FFh`	Reserved	

5.1.1.1.1 REPORT ACL parameter data Granted and Granted All page formats

The Granted and Granted All page formats for the REPORT ACL service action is specified in Table 8.

TABLE 8. Granted and Granted All page formats

Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (00h-01h)							
1	RESERVED							LUN0
2 3	PAGE LENGTH (*n*-3)							
4	RESERVED							
5	IDENTIFIER TYPE							
6 7	IDENTIFIER LENGTH (*n*-7)							
8 *n*	MSB INITIATOR IDENTIFIER							LSB
n+1 *m*	iLUN LIST							

The PAGE LENGTH field shall indicate the number of additional bytes required for this page.

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields are specified in 4.3.2. The IDENTIFIER LENGTH field indicates the number of bytes following taken up by the INITIATOR IDENTIFIER.

The iLUN LIST shall contain a list of iLUN values (eight bytes each) to which are accessible to the specific initiator identifier.

If the INITIATOR IDENTIFIER (only of type TransportID) has an explicit ACL entry to map a particular iLUN to LUN0, then the access controls coordinator shall set the LUN0 field in the Granted page to one and the first entry in the iLUN LIST shall be the iLUN mapped to LUN0.

For the Granted All page, the iLUN LIST is empty and the LUN0 field shall be set to zero.

If the INITIATOR IDENTIFIER does not have access to all logical units at the device, or if the INITIATOR IDENTIFIER (only of type TransportID) has a particular iLUN mapped to LUN0, then the access controls coordinator shall include one Granted page for that identifier and shall include in this page the complete list of iLUNs accessible to that initiator or those initiators.

If the INITIATOR IDENTIFIER has access to all logical units at the device and LUN0 is not expressly mapped to a particular iLUN, the access controls coordinator may include one Granted page for that identifier and shall include in this page the complete list of valid iLUNs of the device. Alternatively, the access controls coordinator may include one Granted All page for that identifier.

One and only one Granted or Granted All page shall be returned for a given INITIATOR IDENTIFIER.

5.1.1.1.2 REPORT ACL parameter data Proxy Tokens page format

The Proxy Tokens page format for the REPORT ACL service action is specified in Table 9.

TABLE 9. Proxy Tokens page format

Byte	Bit 7	6	5	4	3	2	1	0
0	PAGE CODE (02h)							
1	RESERVED							
2 3	PAGE LENGTH (*n*-3)							
4 *m*	PROXY TOKEN/ILUN LIST							

The PAGE LENGTH field shall indicate the number of additional bytes required for this page.

If there are no active Proxy Tokens at the access controls coordinator, the access controls coordinator may either not include the Proxy Tokens page in the parameter data or may include one such page with an empty PROXY TOKEN/ILUN LIST.

At most one Proxy Token page shall be included in the parameter data.

The PROXY TOKEN/ILUN LIST shall contain a list of pairs of Proxy Tokens (each eight bytes) followed by the iLUN (also eight bytes) to which it is associated.

5.1.2 REPORT LU DESCRIPTIONS service action (Mandatory)

TBD -- this should contain a list of all the logical units in the device, together with enough inventory information for the PAM application client to perform her partitioning/allocation responsibilities. For example, it should include iLUN numbers, INQUIRY, IDENTIFIER, and READ CAPACITY data for disks devices, etc.

5.1.3 REPORT LUN MAP service action (Optional)

The REPORT LUN MAP service action of the ACCESS CONTROL IN command is used by the initiator to request the access controls coordinator send a summary of its own access rights. Support for this service action is optional. If the access controls coordinator does not support this service action, the device server shall respond with CHECK CONDITION, sense key ILLEGAL REQUEST, and additional sense code INVALID FIELD IN CDB.

The SERVICE ACTION-SPECIFIC DATA and ILUN field in the CDB for the REPORT LUN MAP service action are reserved.

The ALLOCATION LENGTH shall be at least four (4), sufficient for the header information. If the Allocation Length is less than four (4), then device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST and additional sense code of INVALID FIELD IN CDB.

The format of the returned data shall conform to the specification in 5.1.3.1.

The device server shall respond with a summary of the initiator's LUN Map as determined by the access controls coordinator based on any TransportID corresponding to that initiator, an AccessID enrolled by that initiator in either the enrolled or de-enrolled state and any proxy tokens used by that initiator.

> NOTE: it is the initiator's responsibility to ensure that it has an enrolled AccessID prior to issuing this service action in order to get an accurate report.

5.1.3.1 REPORT LUN MAP parameter data format

The format of the parameter data provided in response to an ACCESS CONTROL IN command with REPORT LUN MAP service action is shown in Table 10.

TABLE 10. REPORT LUN MAP parameter data format

Byte	Bit 7	6	5	4	3	2	1	0
0	RESERVED							
1	RESERVED							
2	MSB							
3	ADDITIONAL LENGTH (*n*-3)							LSB
4 *n*	LUN MAP ENTRIES							

The ADDITIONAL LENGTH field shall contain a count of the number of bytes in available in the remaining parameter data without consideration for insufficient ALLOCATION LENGTH in the requesting CDB (see 5.0).

The LUN MAP ENTRIES contain a description of the LUN Map for this initiator. The format for these entries is found in Table 11.

TABLE 11. LUN MAP ENTRIES data format

Byte	Bit 7	6	5	4	3	2	1	0
0 2	RESERVED							
3	RESERVED						ACCESS TYPE	
4	MSB							
11	LUN VALUE							LSB
12	MSB							
19	ILUN VALUE							LSB

The ACCESS TYPE field indicates the nature of the access rights granted to that initiator as specified in Table 12.

TABLE 12. ACCESS TYPE Codes summary

Code	Description
`00b`	The LUN Map entry was established for this logical unit under a TransportID.
`01b`	The LUN Map entry was established for this logical unit under an enrolled AccessID and not under a TransportID.
`10b`	The LUN Map entry was established for this logical unit under a proxy token.
`11b`	Reserved

The ACCESS TYPE shall be set to `00b` if the LUN Map entry was established on the basis of the initiator's TransportID.

The ACCESS TYPE shall be set to `01b` if the LUN Map entry was established on the basis of an AccessID enrolled by that initiator in either the enrolled or de-enrolled state.

The ACCESS TYPE shall be set to `10b` if the LUN Map entry was established under a proxy token.

The LUN VALUE field indicates the value which would be reported for this logical unit in response to a REPORT LUNS command. The iLUN VALUE field indicates the Internal Logical Unit Number for that logical unit.

AUTHOR'S NOTE: *Should we add a field (maybe in the header of the entire parameter data) which indicates if PAM required that LUN0 be mapped explicitly to the specific logical unit? We could have the same effect if we use up the last Access Type code to indicate that this was mapped to LUN0 under TransportID (this would be valid only if the LUN field was set to zero). Also, do we need this "page" to be on a 4 byte boundary (we could remove the first two bytes at no loss in information)?*

5.1.4 REQUEST PROXY TOKEN service action (Optional)

The REQUEST PROXY TOKEN service action of the ACCESS CONTROL IN command is used by an initiator to obtain from the access controls coordinator a Proxy Token that it can use to grant a third-party temporary access to a logical unit to which it already has non-proxy access. This is used in conjunction with the other PROXY service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

The ALLOCATION LENGTH in the CDB shall be at least eight (8), sufficient for a Proxy Token. If the ALLOCATION LENGTH is less than eight (8), then device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST and additional sense code of INVALID FIELD IN CDB.

If the iLUN bit is set to zero in the CDB for this service action, then the SERVICE ACTION-SPECIFIC DATA field shall contain a Logical Unit Number as might be reported in the REPORT LUNS command. If the iLUN bit is set to one, then the SERVICE ACTION-SPECIFIC DATA field shall contain an Internal Logical Unit Number.

If the Logical Unit Number or the Internal Logical Unit Number corresponds to a logical unit accessible through either a TransportID or an enrolled AccessID for an initiator in the enrolled state, and the access controls coordinator has sufficient resources, then the device server shall respond with GOOD status and return in the parameter data an eight (8) byte Proxy Token. This token (while valid, see 4.4.1) can be used by any initiator to gain temporary access to the associated logical unit via a ASSIGN PROXY LUN service action.

If the Logical Unit Number (iLUN field equal zero) or if the Internal Logical Unit Number (iLUN field equal one) does not correspond to an accessible logical unit as indicated above, then the following rules apply:

a) if the Internal Logical Unit Number is not valid at the access controls coordinator or corresponds to a logical unit not accessible to the requesting initiator, then the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID LU IDENTIFIER;

b) if the Logical Unit Number does not correspond to an accessible logical unit, then the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID LU IDENTIFIER;

c) f the Logical Unit Number or Internal Logical Unit Number corresponds to a logical unit accessible only through a proxy token, then the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID LU IDENTIFIER;

d) if the Logical Unit Number or the Internal Logical Unit Number corresponds to a logical unit accessible only through an enrolled AccessID for that initiator and the initiator is in the de-enrolled state, then the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INITIATOR DE-ENROLLED.

In these cases, no parameter data is returned.

If the access controls coordinator does not have enough resources to create and manage a new Proxy Token, the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to INSUFFICIENT ACCESS CONTROL RESOURCES.

AUTHOR'S NOTE: *do we want to return the iLUN value along with the Proxy Token? There's no real need for that because the initiator can get that information from the REPORT LUN Map command, but it might be useful here as well*.

5.1.5 OPEN PROXY LUN service action (Optional)

This service action is OBSOLETE. It is replaced by the ASSIGN PROXY LUN service action of the OUT command.

The OPEN PROXY LUN service action of the ACCESS CONTROL IN command is used by an initiator to obtain from the access controls coordinator a Logical Unit Number which it can use to address the logical unit associated with the Proxy Token. This is used in conjunction with the other PROXY service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

The ALLOCATION LENGTH in the CDB shall be at least sixteen (16), sufficient for a Logical Unit Number and Internal Logical Unit Number. If the ALLOCATION LENGTH is less than sixteen (16), then device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST and additional sense code of INVALID FIELD IN CDB.

The SERVICE ACTION-SPECIFIC DATA field shall contain a valid Proxy Token associated to a specific logical unit managed by the access controls coordinator. If the Proxy Token is not valid, then the device server shall return CHECK CONDITION status, sense key set to ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID PROXY TOKEN.

If the Proxy Token is valid and the access controls coordinator has sufficient resources, then the device server shall return in the parameter data an eight (8) byte LUN value and the eight (8) byte Internal Logical Unit Number of the associated logical unit.

Regardless any entries in the initiator's LUN Map which reference the logical unit associated with the Proxy Token, the access controls coordinator shall generate a new LUN value (one not in the current LUN Map for that initiator) and the device server shall return that in the parameter data.

If the access controls coordinator does not have enough resources to create and manage a new LUN for that initiator and logical unit, the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to INSUFFICIENT ACCESS CONTROL RESOURCES.

The ILUN bit in the CDB for this service action is reserved.

6.0 ACCESS CONTROL OUT Command

The ACCESS CONTROL OUT command (see Table 13) is used to request service actions by the access controls coordinator to limit or grant access to the logical units to initiators. The command shall be used in conjunction with the ACCESS CONTROL IN command. This command shall not be affected by reservations or persistent reservations.

This command should only be sent to LUN0 (in the SAM-2 hierarchical addressing scheme) and processed by the access controls coordinator. It should be rejected by the device server if addressed to any other LUN with CHECK CONDITION status, sense key of ILLEGAL REQUEST and additional sense code of INVALID OPCODE.

TABLE 13. ACCESS CONTROL OUT command

Byte	Bit 7	6	5	4	3	2	1	0
0	OPERATION CODE (87h)							
1	RESERVED			SERVICE ACTION				
2 9	RESERVED							
10 13	MSB PARAMETER LIST LENGTH							LSB
14	RESERVED							
15	CONTROL							

Fields in the ACCESS CONTROL OUT parameter list specify the information required to perform a particular access control service action.

A description of the additional fields in this command are found in the subclause for each service action.

6.1 ACCESS CONTROL OUT Service Actions

6.1.1 MANAGE ACL service action (Mandatory)

The MANAGE ACL version of the ACCESS CONTROL OUT command is used by an application client to authorize access or revoke access to a logical unit or logical units by initiators. This service action adds, changes or removes an entry or multiple entries in the access controls coordinator's ACL. This service action is mandatory if the ACCESS CONTROL OUT command is supported.

The PARAMETER LIST LENGTH field indicates the amount of data which the initiator shall send to the access controls coordinator in the Data-Out buffer. The structure of the data is as described in 6.1.1.1. If this value is zero, then no data shall be transferred. This is not an error condition and shall result in no changes to the access controls coordinator's state.

Any of the following conditions in any parameter page or header require the device server to respond with CHECK CONDITION, sense key ILLEGAL REQUEST, and additional sense code INVALID FIELD IN PARAMETER DATA and also make no changes to the access controls coordinator's state:

a) the INITIATOR TYPE field indicates an unsupported value;
b) the INITIATOR TYPE=01h (TransportID) and the INITIATOR IDENTIFIER field is invalid as specified in the relevant protocol standard;
c) a Grant LUN0 page is invalid (see 6.1.1.1.1).

If the access controls coordinator cannot complete the command because it has insufficient resources to implement the command, the device server shall return a CHECK CONDITION with sense key ILLEGAL

REQUEST and additional sense data of INSUFFICIENT ACCESS CONTROL RESOURCES. In this case, no changes shall be made to the access controls coordinator's state.

6.1.1.1 MANAGE ACL parameter list format

The format of the parameter list provided for an ACCESS CONTROL OUT command with MANAGE ACL service action is shown in Table 14. The ACL ENTRY PAGE(S) are described in 6.1.1.1.1 and 6.1.1.1.2.

TABLE 14. MANAGE ACL parameter list format

Byte	Bit 7	6	5	4	3	2	1	0
0	MSB							
7	MANAGEMENT IDENTIFIER KEY							LSB
8	MSB							
15	NEW MANAGEMENT IDENTIFIER KEY							LSB
16	RESERVED							
17	FLUSH	RESERVED						
18	RESERVED							
19	RESERVED							
20								
n	ACL ENTRY PAGES(S)							

The MANAGEMENT IDENTIFIER KEY is used to compare with the current Management Identifier Key maintained by the access controls coordinator. If the MANAGEMENT IDENTIFIER KEY in the parameter list does not match the access controls coordinator's current Management Identifier Key, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, additional sense data set to ACCESS DENIED - INVALID MGMT ID KEY and take no other action. If the access controls coordinator successfully implements the requested service action, the access controls coordinator resets its Management Identifier Key to the value specified in the NEW MANAGEMENT IDENTIFIER KEY field.

The FLUSH bit of one instructs the access controls coordinator to move any initiator in the enrolled state into the de-enrolled state.

The ACL ENTRY PAGE(S) that may follow in the parameter list provide additional changes to the ACL data.

Implementation of changes to the access control state of the device follow these rules:

a) no change to the access control state of the device shall occur if the command cannot be processed with status GOOD;

b) if the command can result in status GOOD, the following shall be instantiated as a single indivisible event:

1. changes dictated in the fields in the header of the parameter list are implemented;
2. changes dictated by ACL Entry Pages are implemented;
3. multiple ACL Entry Pages are implemented sequentially.

The structure of ACL Entry pages and the action to be taken is determined by a PAGE CODE field as defined in Table 15. Details of the contents of each page are described in subsequent subclauses.

TABLE 15. ACL Entry PAGE CODE definitions

Page Code	Action	Clause
00h	Grant LUN0	6.1.1.1.1
01h	Grant	6.1.1.1.1
02h	Revoke	6.1.1.1.1
03h	Grant All	6.1.1.1.1
04h	Revoke All	6.1.1.1.1
05h	Revoke Proxy Token	6.1.1.1.2
06h	Revoke All Proxy Tokens	6.1.1.1.2
07b-FFh	Reserved	

6.1.1.1.1 MANAGE ACL parameter data Grant LUN0, Grant, Revoke, Grant All, Revoke All page formats

The Grant LUN0, Grant, Revoke, Grant All and Revoke All page formats for the MANAGE ACL service action is given in Table 16.

TABLE 16. Grant LUN0, Grant, Revoke, Grant All, Revoke All page formats

Byte	Bit 7	6	5	4	3	2	1	0
0	PAGE CODE (00h-04h)							
1	RESERVED							
2 3	PAGE LENGTH (*m*-3)							
4	RESERVED							
5	IDENTIFIER TYPE							
6 7	IDENTIFIER LENGTH (*n*-7)							
8 *n*	MSB INITIATOR IDENTIFIER							LSB
n+1 *m*	ILUN LIST							

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields are described in 4.3.2. The IDENTIFIER LENGTH is described in 5.1.1.1.1.

The PAGE LENGTH field shall indicate the number of additional bytes required for this page.

The ILUN LIST shall contain a set of iLUN values (eight (8) bytes each), in any order. If any iLUN value is not valid at the access controls coordinator, the device server shall fail the command with CHECK CONDITION status, sense key of ILLEGAL REQUEST and additional sense code of ACCESS DENIED - INVALID LU IDENTIFIER.

The Grant LUN0 page has the following restrictions:

a) the IDENTIFIER TYPE field shall be set to indicate a TransportID (01h);
b) no other Grant LUN0 page with the same INITIATOR IDENTIFIER occurs in the parameter data;
c) the ILUN LIST shall contain only one iLUN.

If these conditions are not met, the Grant LUN0 is invalid.

The Grant LUN0 page instructs the access controls coordinator to allow access to the specified iLUN to the indicated initiator and that the LUN Map entry for this logical unit shall have LUN value set to zero.

The Grant page instructs the access controls coordinator to allow access to the listed iLUNs to the initiator or initiators identified by the specified INITIATOR IDENTIFIER. The target shall modify the initiator's LUN Map by adding entries for the specified iLUNs and adjusting the LUN Map according to the rules of 4.1.

The Revoke page instructs the access controls coordinator to disallow access to the listed iLUNs by the initiator or initiators identified by the specified INITIATOR IDENTIFIER. The access controls coordinator shall modify the initiator's LUN Map by removing entries for the specified iLUNs and adjusting the LUN Map according to the rules of 4.1. It is not an error condition if the iLUN references a valid logical unit, but that logical unit is not accessible to the specified initiator(s).

The Grant All and Revoke All pages shall contain an empty iLUN list. That is, there shall be no data in this page after the last byte of the INITIATOR IDENTIFIER field. The Grant All pages allows access to all logical units at the access controls coordinator to the specified initiator(s). The Revoke All pages removes access to all logical units by the specified initiator(s). These are implemented in the same manner as if the corresponding Grant or Revoke page contained the complete list of valid iLUNs.

Multiple pages which identify the same INITIATOR IDENTIFIER are processed sequentially, with the exception of the Grant LUN0 page as noted above.

6.1.1.1.2 MANAGE ACL parameter data Revoke Proxy Token and Revoke All Proxy Tokens page formats

The Revoke Proxy Token and Revoke All Proxy Tokens page formats for the MANAGE ACL service action is given in Table 17.

TABLE 17. Revoke Proxy Token and Revoke All Proxy Tokens page formats

Byte	Bit 7	6	5	4	3	2	1	0
0	PAGE CODE (04h-05h)							
1	RESERVED							
2 3	PAGE LENGTH (*m*-3)							
4 *m*	PROXY TOKEN LIST							

The PAGE LENGTH field shall indicate the number of additional bytes required for this page.

For the Revoke Proxy Token page, the PROXY TOKEN LIST shall contain a list of Proxy Tokens (eight (8) bytes each). This instructs the access controls coordinator to revoke each of the listed Proxy Tokens. It is not an error condition if a Proxy Token specified in this page is not currently valid. In this case, no action is taken by the access controls coordinator with respect to this token.

For the Revoke All Proxy Tokens page, the PROXY TOKEN LIST shall be empty. This instructs the access controls coordinator to revoke all existing Proxy Tokens.

Multiple Revoke Proxy Token and Revoke All Proxy Tokens pages may be included in the parameter data. They are processed sequentially.

6.1.2 DISABLE ACCESS CONTROLS service action (Optional)

The DISABLE ACCESS CONTROLS service action of the ACCESS CONTROL OUT command is used by an application client to return the access controls coordinator to its default state where there are no access controls or ACL data.

For the DISABLE ACCESS CONTROLS service action, the parameter list is described in 6.1.2.1. The PARAMETER LIST LENGTH field in the CDB shall be set to at least four (4) to contain the header. If not, the device server shall return CHECK CONDITION, sense key ILLEGAL REQUEST and additional sense code INVALID FIELD IN CDB.

Successful completion of the service action requires the access controls coordinator to clear all access restrictions for all logical units, change all initiator LUN Maps to the default map (where LUN equals iLUN for all logical units), change all initiators into the not-enrolled state, set the Management Identifier Key to zero, and remove any ACL data from persistent memory.

6.1.2.1 DISABLE ACCESS CONTROLS parameter list format

The format of the parameter list for an ACCESS CONTROL OUT command with DISABLE ACCESS CONTROLS service action is shown in Table 18.

TABLE 18. DISABLE ACCESS CONTROLS parameter list format

Byte	7	6	5	4	3	2	1	0
0	RESERVED							
1	RESERVED							VS
2	MSB							
3	ADDITIONAL LENGTH (*n*-3)							LSB
4								
n	ADDITIONAL PARAMETER DATA							

If the VS field is set to zero, then the ADDITIONAL LENGTH field shall be set to eight (8) and the ADDITIONAL PARAMETER DATA shall have the form specified in Table 19. If the ADDITIONAL LENGTH is not set to eight (8), then the device server shall return CHECK CONDITION, with sense key ILLEGAL REQUEST and additional sense data of INVALID FIELD IN PARAMETER DATA.

TABLE 19. ADDITIONAL PARAMETER DATA when VS=0

Byte	7	6	5	4	3	2	1	0
0	MSB							
7	MANAGEMENT IDENTIFIER KEY							LSB

The MANAGEMENT IDENTIFIER KEY is used to compare with the current Management Identifier Key maintained by the access controls coordinator. If the MANAGEMENT IDENTIFIER KEY in the parameter list does not match the access controls coordinator's current Management Identifier Key, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, additional sense data of ACCESS DENIED - INVALID MGMT ID KEY and take no other action.

Support for the VS field set to one is optional, even if the service action is supported. If the VS field is set to one and the access controls coordinator does not support the VS set to one, then the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST and additional sense data of INVALID FIELD IN PARAMETER DATA. If the VS field is set to one, the contents of the remaining parameter data are vendor-specific.

NOTE: The VS field set to one provides a mechanism for vendors to implement alternatives to a check on the Management Identifier Key for resetting the device to the unconfigured AC state and overriding the current Management Identifier Key.

6.1.3 ACCESS ID ENROLL service action (Mandatory)

The ACCESS ID ENROLL service action of the ACCESS CONTROL OUT command is used by an initiator to enroll an AccessID with the access controls coordinator. The access controls coordinator shall use this information to possibly modify that initiator's LUN Map and associated access rights and change the enrolled state of the initiator. This service action is mandatory if the ACCESS CONTROL OUT command is supported.

The parameter list contains the AccessID. The PARAMETER LIST LENGTH field shall be sixteen (16). If not, then the device server shall return CHECK CONDITION, sense key ILLEGAL REQUEST, and additional sense code INVALID FIELD IN CDB.

If the initiator is in the enrolled or de-enrolled state under a given AccessID and the parameter data contains a different AccessID, then the device server shall respond with CHECK CONDITION status, with sense key set to ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - ENROLLMENT CONFLICT.

If the initiator is in the enrolled or de-enrolled state under given AccessID and the parameter data contains a matching AccessID, then the device server shall respond with GOOD status, and the access controls coordinator shall place the initiator in the enrolled state and make no change to the LUN Map for that initiator.

If the initiator is in the not-enrolled state, and the AccessID in the parameter data has any access rights associated with it in the ACL data, the device server shall respond with GOOD status, and the access controls coordinator shall place the initiator into the enrolled state and modify the LUN Map according to 4.1.

If the AccessID in the parameter data has no access rights associated with it, then the initiator stays in the not-enrolled state and the device server responds with CHECK CONDITION status, sense key of ILLEGAL REQUEST and additional sense code set to ACCESS DENIED - NO ACCESS RIGHTS.

6.1.4 CANCEL ENROLLMENT service action (Mandatory)

The CANCEL ENROLLMENT service action of the ACCESS CONTROL OUT command is used by an initiator to remove its enrollment with the access controls coordinator. Successful completion of this command changes the state of the initiator to the not-enrolled state.

This command shall always return status GOOD regardless of the enrolled state of the initiator. The affect of this command is to remove from that initiator's LUN Map any entries which were included as a result of an enrollment by that initiator. Any subsequent commands addressed to the logical units no longer accessible are handled according to the rules of 4.7.

This command should be used by an initiator prior to any period where use of its accessible logical units will be suspended for an extensive period of time (e.g., if the host is preparing to shutdown). This allows the access controls coordinator to free any resources allocated to manage the enrollment for that initiator.

There is no parameter data for this command. The PARAMETER LIST LENGTH in the CDB for this service action shall be set to zero.

AUTHOR'S NOTE: *instead, should the parameter data be the AccessID with failure response if the "cancelling" AccessID doesn't match the enrolled AccessID? Or should this service action move to the ACCESS CONTROL IN with no parameter data?*

6.1.5 REVOKE PROXY TOKEN service action (Optional)

The REVOKE PROXY TOKEN service action of the ACCESS CONTROL OUT command is used by an initiator to cancel all proxy access rights to a logical unit which were granted to third parties under the specified Proxy Token. This is used in conjunction with the other PROXY-related service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

The parameter data for this command shall be eight (8) bytes long. It shall contain a Proxy Token.

This command shall always return status GOOD.

If the Proxy Token is not valid, that is, not associated with any logical unit at the access controls coordinator, then no further action is taken by the access controls coordinator.

If the Proxy Token is valid, that is, associated with a logical unit at the access controls coordinator, then the access controls coordinator shall take the following additional actions:

a) invalidate the Proxy Token;

b) deny access to that logical unit by any initiator whose rights were granted under that Proxy Token by a REQUEST PROXY LUN service action; that is, remove from all such initiator's LUN Map all proxy entries for this logical unit.

6.1.6 REVOKE ALL PROXY TOKENS service action (Optional)

The REVOKE ALL PROXY TOKENS service action of the ACCESS CONTROL OUT command is used by an initiator to cancel all proxy access rights to a logical unit which were granted to third parties under all Proxy Tokens. This is used in conjunction with the other PROXY-related service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

The parameter data for this command shall be eight (8) bytes long. It shall contain a LUN value.

This command shall always return status GOOD.

If the LUN value is not in the LUN Map for the requesting initiator, then no further action is taken by the access controls coordinator.

If the LUN value is in the LUN Map for the requesting initiator but this entry was established on the basis of a Proxy Token, then no further action is taken by the access controls coordinator.

If the LUN value is in the LUN Map for the requesting initiator and was established on the basis of a non-proxy access right, then the access controls coordinator shall take the following additional actions:

a) invalidate all Proxy Tokens associated to the logical unit referenced by the LUN value in the requesting initiators LUN Map;

b) deny access to that logical unit by any initiator whose rights were granted under any Proxy Token by a REQUEST PROXY LUN service action; that is, remove from all such initiator's LUN Map all proxy entries for this logical unit.

6.1.7 ASSIGN PROXY LUN service action (Optional)

The ASSIGN PROXY LUN service action of the ACCESS CONTROL OUT command is used by an initiator to request the access controls coordinator grant access to a logical unit under the rights of a Proxy Token and to assign that logical unit a particular LUN value in that initiator's LUN Map. This is used in conjunction with the other PROXY-related service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

The PARAMETER LIST LENGTH field in the CDB shall be set to sixteen (16). The parameter data shall contain the eight (8) byte Proxy Token associated with a logical unit followed by an eight (8) byte LUN value.

If the Proxy Token is not valid, then the device server shall return CHECK CONDITION status, sense key set to ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID PROXY TOKEN.

If the Proxy Token is valid but the access controls coordinator cannot assign the requested LUN value to the associated logical unit (that is, modify the initiator's LUN Map with an entry with this LUN referencing the logical unit), then the device server shall return CHECK CONDITION status, sense key set to ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID LU IDENTIFIER. Furthermore, the sense data shall be modified as follows. The SENSE-KEY SPECIFIC bit shall be set as described in 7.22.1 (of SPC-2 revision 15) with the FIELD POINTER field indicating the first byte of the requested LUN (as counted within the full parameter data) which differs from a value which can be supported by the access controls coordinator. Additionally, the next 8 bytes (if available) beyond the last byte of the FIELD POINTER may include a LUN value which the access controls coordinator can support for this proxy token. In this case, the LUN Map for the initiator shall not be changed.

> NOTE: Such a failure scenario can happen only in two rare cases. First, the LUN is already assigned in the initiator's LUN Map. The initiator can know this in advance, however, and should not be making this request. Second, if the LUN value, for any reason, cannot be supported by the access controls coordinator.

If the Proxy is valid but the access controls coordinator has insufficient resources to perform the requested action, then the device server shall respond with CHECK CONDITION status, sense key of ILLEGAL REQUEST and additional sense data of INSUFFICIENT ACCESS CONTROL RESOURCES.

If the Proxy is valid and the access controls coordinator has sufficient resources, then the device server shall return status GOOD and modify the LUN Map for that initiator as requested.

6.1.8 RELEASE PROXY LUN service action (Optional)

The RELEASE PROXY LUN service action of the ACCESS CONTROL OUT command is used by an initiator to remove a LUN from its LUN Map which was created with a Proxy Token and the ASSIGN PROXY LUN service action. This is used in conjunction with the other PROXY-related service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

This command should be used by an initiator when its access to that logical unit is no longer required under its proxy rights (e.g., when a copy server has completed a specific third party copy service under the proxy). This allows the access controls coordinator to free any resources allocated to manage the proxy for that initiator.

The PARAMETER LIST LENGTH field in the CDB shall be set to sixteen (16). The parameter data shall contain the eight (8) byte Proxy Token associated with a logical unit followed by an eight (8) byte LUN value as was used in the ASSIGN PROXY LUN service action.

AUTHOR'S NOTE: *This needn't contain both the Proxy Token and the LUN. It really only needs one of them and the more natural one is LUN (since that is what is being released by the initiator). Specifying either just the Proxy Token or both the Proxy Token and the LUN means additional language (not currently included here) to deal with error conditions (invalid token, token not associated with LUN, etc.). It is spec'ed here as both for consistency with the parameter data of the ASSIGN service action.*

6.1.9 CLOSE PROXY LUN service action (Optional)

This service action is OBSOLETE. It is replaced by the RELEASE PROXY LUN.

The CLOSE PROXY LUN service action of the ACCESS CONTROL OUT command is used by an initiator to remove a LUN from its LUN Map which was created with a Proxy Token and the OPEN PROXY LUN service action. This is used in conjunction with the other PROXY-related service actions of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands.

This command should be used by an initiator when its access to that logical unit is no longer required under its proxy rights (e.g., when a copy server has completed a specific third party copy service under the proxy). This allows the access controls coordinator to free any resources allocated to manage the proxy for that initiator.

The parameter data for this command shall be sixteen (16) bytes long. It shall contain a LUN value and an iLUN value (as might have been returned under the OPEN PROXY LUN service action).

If the LUN and iLUN combination are associated with a valid logical unit under which access was granted by a Proxy Token, then the LUN Map entry for this LUN and logical unit is removed from the LUN Map and the device server responds with GOOD status.

If the LUN and iLUN combination do not associate with a valid logical unit accessible under a Proxy Token, the device server responds with CHECK CONDITION status, sense key of ILLEGAL REQUEST and ASC/ASCQ set to ACCESS DENIED - INVALID LU IDENTIFIER. No additional action is taken by the access controls coordinator.

A. Changes required in SAM-x.

This section contains some changes required in SAM-x to deal with Task Management in the presence of access control. (Section numbers correspond to SAM-2, rev. 10).

A.1. Changes for the end of section 6.0.

The device server response to task management requests is subject to the access control state of the access controls coordinator (as instantiated by ACCESS CONTROL OUT commands) as follows:

a) a task management request of ABORT TASK, ABORT TASK SET or CLEAR ACA shall be unaffected by the presence of access restrictions;

b) a task management request of CLEAR TASK SET or LOGICAL UNIT RESET received from an initiator that is denied access to the logical unit shall cause no change to the logical unit, but shall receive a response of FUNCTION COMPLETE.

c) a TARGET RESET task management request shall initiate a logical unit reset as described in 5.6.7 for all logical units to which the initiator has access, and shall cause no change to any logical units to which the initiator is denied access. A response of FUNCTION COMPLETE shall be returned in the absence of any other error condition.

A.2 Additions for section 5.6.6

While the device server response to task management requests is subject to the access rights of the requesting initiator, a target hard reset in response to a reset event within the service delivery subsystem shall be unaffected by access control.

B. Changes required in FCP-x.

This section contains the changes required in FCP-x. This includes the description of the TransportID. (Section numbers correspond to FCP-2, rev. 02).

B.1. Specification of the TransportID

The TransportID structure is 24 bytes long and is described in Table 20.

TABLE 20. TransportID for FCP.

Byte	Bit 7	6	5	4	3	2	1	0
0	RESERVED					PA_VAL	PN_VAL	NN_VAL
1 3	RESERVED							
4 7	MSB	PROCESS ASSOCIATOR						LSB
8 16	MSB	WWPORTNAME						LSB
17 24	MSB	WWNODENAME						LSB

A PA_VAL bit of one indicates that the PROCESS ASSOCIATOR field is valid. Similarly, the PN_VAL and NN_VAL bits of one indicate that the corresponding WWPORTNAME and WWNODENAME fields, respectively, are valid. A value of zero for any of these bits indicate that the corresponding field is invalid and shall be ignored. At least one of these validity bits must be set to one. If not, then the TransportID is invalid.

If any of the valid fields are inconsistent, that is, they do not correspond to a device in the fabric, then the TransportID is invalid.

B.2. Changes to 6.3

CHANGE:

All tasks, reservations, mode page parameters ...that are logged out are not affected.

TO:

All tasks, reservations, mode page parameters, AccessID enrollment states, and status for image pairs removed by the PRLO operation are set to the state they would have after a SCSI hard reset or power on reset. Only the specified image pairs are logged out. Open exchanges for logged out image pairs shall be terminated by a recovery abort operation. (See 8.1.2.2.) Tasks, reservations, mode page parameters, AccessID enrollment states, and status for image pairs other than those that are logged out are not affected.

B.3. Additional rows required in Table 4:

TABLE 21. Changes to Table 4 of FCP-2: Clearing effects of SCSI Initiator Actions

	POWER	RESETLIP	LOGO, PLOGI	ABTS	PRLI, PRLO	TPRLO	TGTRESET	CLEAR	ABORT	LURESET
ACL and Management Identifier Key	N	N	N	N	N	N	N	N	N	N
AccessID enrollment state to de-enrolled state										
For all SCSI initiators in enrolled state	Y[a]	Y	Y[b]	N	Y[b]	Y[c]	Y[a]	N	N	N
Only for SCSI initiator port initiating action in enrolled state	-	-	Y[6]	N	Y	-	N	N	N	N

a. Transition is to de-enrolled or not-enrolled state in implementation dependent manner

b. For PRLO only and for explicit or implicit LOGO only

c. Only for the initiator attached to the port in the third party logout page.

C. Changes required in SPI-x.

This section contains the changes required in SPI-x. This includes the description of the TransportID. (Section numbers correspond to SPI-3, rev. 10).

C.1. Specification of the TransportID

The TransportID structure is 4 bytes long and is described in Table 22.

TABLE 22. TransportID for SPI.

Byte	Bit 7	6	5	4	3	2	1	0
0	RESERVED							
1								
2	MSB							
3		SCSI ADDRESS						LSB

The SCSI ADDRESS field indicates the SCSI address of the initiator. (See the glossary in SPI-3, 3.1.77.)

C.1. Volatility of the AccessID enrollments

AccessID enrollment state of initiators (established initially with the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action) shall be transitioned by the following events or states:

a) power cycle of the device server;

b) hard reset bus condition.

The transition state is implementation dependent subject to the rules of 4.5 (of this proposal).

D Changes to SPC-2 (rev 15)

D.1 Changes to Table 8 of SPC-2 (rev 14)

The following additional line(s) need to be added to Table 8 of SPC-2 (rev 14).

TABLE 23. Additional rows for Table 8, SPC-2 (rev 14)

Command	Addressed LU is reserved by another initiator [A]	Addressed LU has this type of persistent reservation held by another initiator [B]				
		From any initiator		From registered initiator (RO all types)	From initiator not registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access - RO
ACCESS CONTROL IN/OUT	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed

D.2 Changes to EXTENDED COPY

In the target descriptors of clauses 7.4.5.1-7.4.5.4, make the following changes. Addition of a 2bit field called LU ID TYPE in byte 3 (bits 0-1) of the target descriptor which can be used to define the interpretation of the LOGICAL UNIT NUMBER field in bytes 4-11. Change the name of this field to LU IDENTIFIER. Its contents would be interpreted according to the value of the LU ID TYPE as defined in the Table 24.

TABLE 24. LU ID TYPE and LU IDENTIFIER description

LU ID TYPE	LU IDENTIFIER description
00b	Logical Unit Number
01b	Reserved
10b	Proxy Token
11b	Reserved