Date: December 16, 1999

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 device server 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 device server. The new commands and actions are not restricted to storage devices but are applicable (or extendable) to any device server.

This draft reflects comments, questions and suggestions from folks at LSI Logic, Sun Microsystems, Adaptec, Compaq and others at IBM.

This revision has a redefined subcomponent model (e.g., for elements) from the previous draft. The new model is a bit simpler than the previous one and sets the stage for the next revision which has this finer granularity of access controls removed. We're submitting this version more for the historical record; if the need for this finer granularity arises in the future (e.g., for object-group access controls without encryption in Object-based Storage Devices), a starting point has already been documented.

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 device server;

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;

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 inadaquate 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 itself 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 device server, not just storage devices. Resource requirements at the device server 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 device server with respect to access control functions and a

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.

Data-Out command 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 device server's key (in the unlikely event that PAM forgets it) is not specified in this proposal, though the proposal does include a template for such an override function. Vendors are free to implement alternative methods as well such as jumpers, other vendor-specific commands (which might include firmware download), etc.

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 inadaquate. 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 device server access controls much simpler. But it also implies requirements on the part of device server 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. 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 outway this concern.

What follows this main section is a detailed description of the new commands and device server requirements and constitutes the normative part of the proposal. 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-2 (or SPC-3). Large issues are listed in the next section.*

2.0 Additional major issues, questions and revision history[1]

2.1 The proxy model

The model for proxies is relatively simple and it's not clear if it is sufficient.

In the current model, all initiators with access rights granted by PAM are equivalent. If two or more different initiators issue a proxy for the same third party initiator identifier/ACC combination, it is as if a single initiator repeated the request (that is, the second request is redundant). Furthermore, the device server is not required to maintain a record of which initiator issued a proxy. Explicit revocation of a proxy can be handled by any initiator with PAM-granted rights. This can have some consequences. For example, if two different initiators granted a proxy to some third party copy server for their own purposes, revocation of the proxy by the first initiator when his copy operation is complete might interrupt the second initiator's copy operation.

If these issues are a concern, the proxy model would have to be enriched. One choice is that the target manages a complete record of which initiator granted which proxy. Then revocation would only be allowed by implicit actions (like PAM's Clearing) or by the granting initiator. But the grantor might have had rights under multiple different identifiers. Some identifiers (like AccessID) are associated to a common host and it is fundamentally that host which is granting the proxy. In this case, the host, through any of its initiators, should be able to revoke the proxy. But the target can't tell that since it can't tell whether the grantor had its initial rights because of the AccessID or some flavor of TransportID or some other mechanism. A second choice is that the target simply maintain a counter of the number of proxies requested for each initiator identifier/ACC combination and each revocation decrements the counter until it is zero. This is feasible but it's not clear if it adds any necessary function. (It would also require a small change to the REPORT ACL parameter data for proxies, namely, the counter should be returned as well).

An additional issue is the following. If an initiator which had rights grants a proxy to a third party and then has his own rights revoked by PAM, the third party's rights are still in place. Initially, it might be a good design point that the the third party's rights are also revoked, but again this requires the target to maintain a record of the grantor and to be able to tell under what initiator identifier it used its proxy (and, as above, it's not clear if this can be done). PAM does have the mechanism (albeit a hammer of sorts with the Clear function) to remove the third party's rights. Also, other initiators with rights can also expressly revoke the proxy.

To avoid the complexities mentioned above, we've chosen the simpler and easier to manage model that all initiators with PAM-granted access are equivalent, but is that sufficient?

Is there a reason to add some language to Section 4.0 to make this model for proxies more clear? Is the language of REPORT ACL and REPORT INITIATOR ACCESS with respect to report proxy access rights clear enough?

2.2 Override of Manage ACL Key

We've left the issue of how PAM can override the Manage ACL Key of a device server (in the unlikely event that PAM forgets it) to a vendor-specific flag in the RESET AC service action of the ACCESS CONTROL OUT command. This at least allows for a template for some flavor of standardization, but allows implementors a lot of flexibility (e.g., just set the VS bit, or set the VS bit and send some specific parameter data).

1.This section is primarily to raise discussion points for the teleconference; it should be removed by next revision.

2.3 Changes from previous revisions

2.3.1 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 Manage ACL 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 maintaing 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.

3.0 Glossary and Acronyms

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

3.1 Glossary

access controllable component: An addressable physical or logical subcomponent of a logical unit on which access controls can be maintained. Subcomponents of the logical unit such as elements of a medium changer can be access controllable components.

Access Control List: The list of initiator identifiers and their access rights to a logical unit or ACCs within the logical unit.

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 devicer server is able to determine the access rights for that initiator.

ACL Disabled: The state of a device server in which the Access Control List is ignored during command processing for commands which are subject to access controls.

ACL Enabled: The state of a device server in which the Access Control List is checked during command processing for commands which are subject to access controls.

enrolled: The condition that exists for an initiator from a successful completion of an ACCESS CONTROL OUT command with ACCESS ID ENROLL service action until the initiator enrollment is removed.

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

unconfigured AC: The state of the device server when the device server is in the ACL Disabled state and PTPL disabled state, the Manage ACL Key is zero and the ACL is empty.

3.2 Acronyms

ACL: Access Control List

ACC: access controllable component

4.0 Access Controls

Access controls may be used to restrict the commands that an initiator can execute at a device server. An initiator can be identified uniquely by an access identifier, called an AccessID, as defined in 4.2 or by a protocol-specific identifier, called a TransportID, as defined in the relevant protocol standards.

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

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

There are only two types of access rights for an initiator:

a) unrestricted access - all commands are handled in their normal fashion, and

b) restricted access - only certain commands are accepted and access-restricted commands are rejected with an error condition.

The scope of an access control shall be a logical unit. However, a device which contains ACCs can be configured to allow more restricted access to such ACCs.

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 device server supports the access control commands, it shall be able to maintain at least one entry in its ACL. In this way, the logical unit can be dedicated to at least one initiator and so restrict access to competing initiators. The default ACL is empty.

Along with the ACL, the device server shall maintain a Manage ACL Key of 8 bytes (64 bit integer). The default value for the Manage ACL Key is zero.

Additionally, the device server shall maintain an Access Restricted state bit which shall be preserved across power cycles. This state bit shall be zero if the device server is in the ACL Disabled state and shall be one if the device server is in the ACL Enabled state. The default value for this state bit is zero.

The device shall be in the unconfigured AC state as shipped from the factory prior to the first successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action. It can also change to this state after successful completion of the ACCESS CONTROL OUT command with RESET AC service action.

The ACCESS CONTROL OUT command with MANAGE ACL service action can selectively change the ACL Enabled/Disabled state or PTPL state for the device, add or delete entries from the ACL, or change the value of the Manage ACL Key.

For each command, this standard or a related command standard for the particular device type defines the conditions under which the command from a particular initiator is or is not subject to access control. The SPC-commands not subject to access control are:

a) INQUIRY;

b) LOG SENSE;

c) PREVENT/ALLOW, PREVENT equal zero;

d) REPORT DEVICE IDENTIFIER;

e) REPORT LUNS;

f) REQUEST SENSE;

g) RELEASE(6), RELEASE(10);

h) PERSISTENT RESERVE OUT, with RELEASE service action;

i) ACCESS CONTROL IN;

j) ACCESS CONTROL OUT, except the PROXY ACCESS service action.

AUTHOR'S NOTE: *I would include MODE SENSE in the above list as it only requests information about the device and not the user data. It is left out of the list above because it is subject to reservation conflicts. Does anyone think it ought to be added to the list above?*

For SBC devices, the following additional commands are not subject to access control:

a) READ CAPACITY;

b) START/STOP UNIT, START equal one, POWER CONDITION equal zero;

c) SET LIMITS(10), SET LIMITS(12).

For SMC devices, the following additional commands are not subject to access control:

a) READ ELEMENT STATUS, CURDATA equal one;

b) READ ELEMENT STATUS ATTACHED, CURDATA equal one.

AUTHOR'S NOTE: *In spite of a suggestion to the contrary, I've left the above as separate paragraphs as I feel that reads better (easier to parse for an implementor). The final wording of this text into SPC-x is however up to the editor of that document.*

AUTHOR'S NOTE: *Are there other device types which need device-specific commands added to this list?*

4.1 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 or PROXY ACL service actions) indicates that a new access control state is established. Changes in the access control state may apply to some or all tasks queued before completion of the access control command. The new access control state shall apply to all tasks received by the device server after successful completion of the access control command. 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 Identifying initiators

Access rights at a device server 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 device server by an ACCESS ID ENROLL service action.

If an initiator issues the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action, the device server shall first remove any existing AccessID enrollment for that initiator and then perform the following functions:

a) If the AccessID has been granted some access rights to the device server, the device server shall maintain in volatile memory the association of this AccessID with the initiator. Subsequent com-

mands from that initiator can then be handled subject to the access rights of the associated AccessID.

b) If the AccessID has no specific access rights to the device, the device server shall maintain a volatile record that an AccessID was enrolled by that initiator. Optionally, the device server can maintain the initiator/AccessID association in this case.

NOTE1: When a MANAGE ACL service action to a device server grants access rights to an AccessID that previously had none, a host that has enrolled that AccessID through some initiator may need to re-enroll in order to make use of the granted access. This happens if the device server does not retain the initiator/AccessID associations for AccessIDs that have no access rights. In this case, the sense data from the device server would not indicate to the host the need to re-enroll. Re-enrollment can be triggered, if necessary, by setting the parameters in the configuration command to invoke the FLUSH action of initiator/AccessID associations list and enrolled initiators list at the device server.

The initiator/AccessID association list and the enrolled initiators list shall be stored in volatile memory; it shall be invalidated after a power-cycle of the device server.

If any event in the service delivery subsystem causes the device server to question whether the AccessID of an enrolled initiator has changed, it shall remove this initiator/AccessID association or remove the enrollment, as appropriate. The initiator should detect this change of state at the next access restricted command failure and may then reissue the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action followed by a retry of the failed command.

If an access right is granted on the basis of a TransportID that contains non-persistent components (e.g., in FCP, an identifier which includes the N_Port identifier) which might be affected by such events in the service delivery subsystem, then such events shall always revoke that access right.

If an access right is granted on the basis of a TransportID that contains only persistent components (such as WWPortName or WWNodeName in FCP) which are not affected by events in the service delivery subsystem, the device server shall reestablish the initiator/TransportID association and thereby restore the access right.

NOTE2: The detection process required here is similar to that in the following requirement from FCP-2, rv 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 FCP, the initiator port identifier is the S_ID.) However, in this proposal, the device server does not maintain this association but severs it until a new ACCESS ID ENROLL service action is received. This has two purposes. First, it prevents physical reconfiguration of initiator ports (say, moving an HBA from one initiator machine to another) from exposing protected data. Second, it associates the access protection with initiator machines and not with the port or node (or HBA). In particular, in Fibre Channel, a PRLO or LOGO (either explicit or implicit) shall invalidate initiator/AccessID associations.

On the other hand, for those access rights based on persistent WWNs, it is exactly this detection and “relationship between address identifier of the initiator... reconfiguration events that may change the S_ID of the initiator” which enables the device server to autonomously reestablish the initiator/WWN/access relationship.

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.*

Additionally, there is provision for vendor-specific initiator identification placeholders (see 4.2.1).

4.2.1 Identifier Type and Initiator Identifier

Initiators are identified in parameter data with an IDENTIFIER TYPE code and INITIATOR IDENTIFIER field as defined in Table 1.

TABLE 1. 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.

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.*

4.3 Granting and revoking access rights

There are two service actions defined for configuring access rights and controls at a device server.

The MANGE ACL service action can grant or revoke access rights for the logical unit to an initiator based on the AccessID identifier or TransportID identifier. This same service action can change the ACL Enabled/Disabled state for the device. It can request the access control information persist through power loss, or disable this function.

The PROXY ACCESS service action can grant or revoke access rights for the logical unit to a third party initiator. This service action is valid only if the requesting initiator already has access rights to the logical unit consistent with the requesting proxy. That is, an initiator can extend its own existing rights to another initiator. This allows an initiator to autonomously create an access right for a third party to facilitate additional services such as third party copy. See 4.6.1 for additional information.

An initiator's access right to a logical unit or an ACC within the logical unit is the logical 'or' of all rights granted under both MANAGE ACL and PROXY ACCESS for any identifier which corresponds to that initiator. 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 by other initiators under either the same or different identifiers which correspond to the requesting initiator. Revocation of that initiator's access rights occurs only when all such access rights have been revoked.

When an initiator's access rights to a logical unit or ACC are changed (granted or revoked), the rules of 4.1 shall apply with respect to all commands from that initiator.

4.4 Preserving access control information

A device server is required to maintain in non-volatile form the Access Restricted state bit which indicates whether the device is unconstrained (ACL Disabled) or constrained (ACL Enabled). This state flag shall persist through power cycles.

The application client may request that the device server preserve the ACL and the Manage ACL Key across power cycles by requesting the Persist Through Power Loss (PTPL) capability. This is done by setting the PTPL bit to one in the MANAGE ACL service action parameter list. Support for this feature is optional.

AUTHOR'S NOTE: *We use PTPL instead of APTPL to avoid confusion with the APTPL of PERSISTENT RESERVATIONS.*

If the device server does not support the PTPL feature or if the PTPL feature is disabled, it shall perform the following functions after a power off period until new access control information is transmitted to the device server using the ACCESS CONTROL OUT command with MANAGE ACL service action:

a) if the Access Restricted state bit is one prior to power off, the device server shall set the ACL Enabled state;

b) if the Access Restricted state bit is zero prior to the power off, the device server shall set the ACL Disabled state;

c) reset the Manage ACL Key to zero.

If the device server's non-volatile memory is not ready (either to read the Access Restricted state bit or the ACL under the PTPL state), the device server shall return CHECK CONDITION, 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 access-restricted commands.

If the device server is in the PTPL state only those access rights granted via a MANAGE ACL service action and the Manage ACL Key shall persist across a power cycle; proxy access rights shall not.

A device server's ACL, the initiator/AccessID association list and the enrolled initiator list shall not be directly affected by task management functions such as TARGET RESET. Protocol- and interconnect-specific reset events, however, may cause these to be cleared. In particular, an event that shall cause Persistent Reservations (with APTPL not set) to be cleared shall cause the initiator/AccessID and enrolled initiator lists to be cleared, and shall also cause the ACL and Manage ACL Key to be cleared when the PTPL state is not set.

4.5 Reporting access control information

There are two ways to request a report from the device server about its access control information.

A service action (REPORT ACL) shall report all access control information for the entire device server independent of the requesting initiator.

In this case, the information includes the following:

a) the number of ACL entries currently managed at the device server;

b) the ACL Enabled/Disabled state of the device server;

c) the PTPL state of the device server;

d) the list of initiator identifiers and their access rights currently in effect; this includes proxy entries.

A related service action (REPORT INITIATOR ACCESS) shall return summary information relevant to the requesting initiator.

4.6 Verifying access rights for initiators

Access-restricted commands from a given initiator are validated in the following manner.

If the device server is in the ACL Disabled state or any of the the following conditions hold with respect to all ACC referenced by the command, the command is handled in the usual way:

a) the initiator has a non-proxy access right granted under either an enrolled AccessID or a TransportID;

b) the initiator has a proxy access right granted under either an enrolled AccessID or a TransportID and the command is not an ACCESS CONTROL OUT with PROXY ACCESS service action;

If none of these conditions hold, the device server shall transfer no data and shall respond with CHECK CONDITION, sense key ILLEGAL REQUEST and additional sense data set according the following:

1. if the initiator has enrolled an AccessID and neither the AccessID nor any TransportID for that initiator has access rights, then the ASC/ASCQ is set to ACCESS DENIED - INITIATOR NOT AUTHORIZED;
2. if the initiator has not enrolled an AccessID, then the ASC/ASCQ is set to ACCESS DENIED - INITIATOR NOT ENROLLED.

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

4.6.1 Access rights to ACCs

For a device server with ACCs, the following rules apply.

a) An access right granted an initiator to the logical unit implicitly grants access to all ACCs within the logical unit.

b) An explicit access right granted an initiator to an ACC or ACCs within a logical unit (but not granted to the full logical unit) also implicitly grants that initiator access to the logical unit for any command which does not reference any other ACC.

c) Revoking access rights to an ACC of a logical unit revokes access to the logical unit unless other explicit or implicit rights are granted.

d) A non-proxy access right granted an initiator to the full logical unit can be proxied to another initiator either for the full logical unit or for any ACC or ACCs within the logical unit.

e) A non-proxy access right granted an initiator to a specific ACC (and not to the full logical unit) can be proxied to another initiator for that ACC.

4.7 Access Control Service Actions

Table 2 gives a summary list of the access control service actions.

TABLE 2. Access Control Commands and Service Actions

Code	Name	Section
ACCESS CONTROL IN (OPCODE 86h)		
00h	REPORT ACL	5.1.1
01h	REPORT INITIATOR ACCESS	5.1.2
02h-0Fh	Reserved	
10h-1Fh	Vendor-specific	
ACCESS CONTROL OUT (OPCODE 87h)		
00h	ACCESS ID ENROLL	6.1.1
01h	MANAGE ACL	6.1.2
02h	PROXY ACCESS	6.1.3
03h	RESET AC	6.1.4
04h-0Fh	Reserved	
10h-1Fh	Vendor-specific	

4.8 Access Control Additional Sense Codes

Table 3 contains a list of the Additional Sense Code and Additional Sense Code Qualifiers relevant to access control.

TABLE 3. Access Control Additional Sense Codes and Qualifiers

ASC	ASCQ	Name	Function
20h	01h	ACCESS DENIED - INITIATOR NOT ENROLLED	Initiator has not sent an ACCESS ID ENROLL service action.
20h	02h	ACCESS DENIED - INITIATOR NOT AUTHORIZED	A enrolled initiator has access permissions insufficient for the requested command
20h	03h	ACCESS DENIED - INVALID MANAGE ACL KEY	The Manage ACL Key value is does not match the Manage ACL Key maintained at the device server
55h	05h	INSUFFICIENT ACCESS CONTROL RESOURCES	The device server has exhausted its resources for access control

5.0 ACCESS CONTROL IN command

The ACCESS CONTROL IN command (see Table 4) is used to obtain information about the access controls that are active within a device server. The command shall be used in conjunction with the ACCESS CONTROL OUT command. This command shall not be affected by reservations, persistent reservations or access controls (with the exception noted in 5.1.1).

TABLE 4. 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	MANAGE ACL KEY							LSB
10	MSB							
13	ALLOCATION LENGTH							LSB
14	RESERVED							
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 MANAGE ACL KEY field is described in the appropriate subclause for each service action.

The ALLOCATION LENGTH shall be at least eight (8), sufficient for the header information. 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 ALLOCATION LENGTH is not sufficient to contain the available data, the first portion of the data shall be returned. This shall not be considered an error. If the remainder of the data is required, the application client may send a new ACCESS CONTROL IN command with a ALLOCATION LENGTH field large enough to contain the entire available data.

AUTHOR'S NOTE: *Is the above paragraph necessary considering the section on Allocation Length in SPC-2?*

5.1 ACCESS CONTROL IN Service Actions

The ACCESS CONTROL IN command service actions are defined in Table 5.

TABLE 5. ACCESS CONTROL IN service actions

Code	Name	Description
00h	REPORT ACL	Used by a client application to query the device server's current ACL. See 5.1.1
01h	REPORT INITIATOR ACCESS	Used by an initiator to get a summary of his access rights at the device server. See 5.1.2
02h-0Fh	Reserved	Reserved
10h-1Fh	Vendor-specific	Vendor-specific

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 currently maintained on the device server.

For a REPORT ACL service action, if the MANAGE ACL KEY field in the CDB does not match the Manage ACL Key maintained at the device server, the device server shall return no data and respond with CHECK CONDITION, sense key ILLEGAL REQUEST, additional sense data set to ACCESS DENIED - INVALID MANAGE ACL KEY.

The format of the returned data shall conform to the specification in 5.1.1.1. Active third party Proxy access rights pages shall also be returned by the device server (see 6.1.3).

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.

TABLE 6. REPORT ACL parameter data format

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	RESERVED							
1	PTPL	RESERVED						ENBLD
2	MSB							
3		RESOURCE UTILIZATION						LSB
4	MSB							
7		ADDITIONAL LENGTH (*n*-7)						LSB
8								
n	ACL ENTRY PAGE(S)							

The PTPL bit of one indicates that the Persist Through Power Loss state is enabled at the device server. A value of zero indicates that this state is disabled.

The ENBLD bit of one indicates that the device server is in the ACL Enabled state. A value of zero indicates that the device server is in the ACL Disabled state.

The RESOURCE UTILIZATION field shall indicate the total number of entries in the device server's ACL currently used. If the field size is insufficient to contain the actual value, then the field shall set to FFFFh.

The ADDITIONAL LENGTH field contains 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 (see 5.0).

The ACL ENTRY PAGES contain a description of the ACL maintained by the device server. The format for these pages is

5.1.1.1.1 REPORT ACL parameter data ACL Entry page format

The ACL Entry page format for the REPORT ACL service action is specified in Table 7.

TABLE 7. REPORT ACL parameter data ACL Entry page format

Byte	Bit 7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	PAGE LENGTH (*n*-1)							
2	RESERVED							
3	ACC TYPE				RESERVED			PROXY
4	MSB							
7	ACC ADDRESS							LSB
8	RESERVED							
9	RESERVED							
10	IDENTIFIER TYPE							
11	IDENTIFER LENGTH (*n*-11)							
12	MSB							
n	INITIATOR IDENTIFIER							LSB

The ACC TYPE and the ACC ADDRESS fields specify either the logical unit or an ACC to which the specified initiator identifier has been granted access. The valid ACC TYPE codes and specifications for the associated ACC ADDRESS are defined in Table 8.

TABLE 8. ACC TYPE Codes

Code	Name	ACC Address
0h	logical unit	zero
1h	reserved	zero
2h	element	element address, zero filled in the most significant bytes to fit the field
3h-Bh	reserved	zero
Ch-Fh	VS	VS

A PROXY value of zero indicates that the access right was created by an ACCESS CONTROL OUT command with MANAGE ACL service action. The PROXY value of one indicates that the access right to the indicated ACC was created by an ACCESS CONTROL OUT command with PROXY ACCESS service action by some other initiator.

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields are specified in 4.2.1. The IDENTIFIER LENGTH field indicates the number of bytes following taken up by the INITIATOR IDENTIFIER.

One Entry page with PROXY bit equal to zero shall be returned for a given ACC TYPE/ACC ADDRESS and INITIATOR IDENTIFIER combination.

One Entry page with PROXY bit equal to one shall be returned for a given ACC TYPE/ACC ADDRESS and INITIATOR IDENTIFIER combination for all third party proxies granted under the PROXY ACCESS service action, regardless of which or how many initiators granted the access right.

5.1.2 REPORT INITIATOR ACCESS service action (Optional)

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

The MANAGE ACL KEY field in the CDB for the REPORT INITIATOR ACCESS service action is reserved.

The format of the returned data shall conform to the specification in 5.1.2.1.

The device server shall respond with a summary of the initiator's access rights as determined by the device server based on any enrolled AccessID or any TransportID corresponding to that initiator.

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

5.1.2.1 REPORT INITIATOR ACCESS parameter data format

The format of the parameter data provided in response to an ACCESS CONTROL IN command with REPORT INITIATOR ACCESS service action is shown in Table 9. The ACC PAGE(S) are described in 5.1.2.1.1.

TABLE 9. REPORT INITIATOR ACCESS parameter data format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	RESERVED							
1	RESERVED						ACCESS TYPE	
2	RESERVED							
3	RESERVED							
4	MSB							
7	ADDITIONAL LENGTH (*n*-7)							LSB
8								
n	ACC PAGE(S)							

The ACCESS TYPE field indicates the nature of the access rights granted to that initiator as specified in Table 10 .

TABLE 10. ACCESS TYPE Codes summary

Code	Description	Additional Length	ACC Pages
00b	No access to the logical unit	zero	empty
01b	Access to the logical unit was established only by a PROXY ACCESS service action.	zero	empty
10b	Access to the logical unit was established by a MANAGE ACL service action.	zero	empty
11b	Access to the logical unit is limited to specific ACCs.	variable	see 5.1.2.1.1

The ACCESS TYPE shall be set to 00b if the initiator has no access rights to the logical unit either explicitly or implicitly. In this case, the ADDITIONAL LENGTH field shall be set to zero and no ACC PAGES shall be returned.

The ACCESS TYPE shall be set to 01b if the initiator has access rights to the logical unit which were granted only by a PROXY ACCESS service action. That is, there are no access rights for this initiator granted under the MANAGE ACL service action. In this case, the ADDITIONAL LENGTH field shall be set to zero and no ACC PAGES shall be returned, whether or not explicit rights have been granted that initiator to any ACC.

The ACCESS TYPE shall be set to 10b if the initiator has access rights to the logical unit granted under any MANAGE ACL service action whether or not there are any rights granted under a PROXY ACCESS service action. This includes the case where the device server is in the ACL Disabled state. In this case, the ADDITIONAL LENGTH field shall be set to zero and no ACC PAGES shall be returned, whether or not explicit rights have been granted that initiator to any ACC.

The ACCESS TYPE shall be set to 11b if the initiator has no explicit proxy or non-proxy access rights to the logical unit but has access rights to specific ACCs. In this case, ACC PAGES (see 5.1.2.1.1) for each such ACC shall be returned. 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 (see 5.0).

AUTHOR'S NOTE: *This data is intended to be a very brief summary of that initiator's access rights, not a detailed listing.*

5.1.2.1.1 REPORT INITIATOR ACCESS parameter data ACC Page format

The REPORT INITIATOR ACCESS service action parameter data ACC Page format is specified in Table 11.

TABLE 11. REPORT INITIATOR ACCESS parameter data ACC Page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PAGE CODE (01h)							
1	PAGE LENGTH (06h)							
2	RESERVED							
3	ACC TYPE				RESERVED			PROXY
4	MSB							
7	ACC ADDRESS							LSB

All fields in this parameter page are as defined in 5.1.1.1.1. For this page, the ACC TYPE field shall not indicate the logical unit (value zero). That is, this page is only to be used for reporting access rights to specific ACCs of the logical unit.

At most one ACC Page with PROXY bit equal to zero shall be returned for a given ACC, whether or not that initiator has access rights granted under different INITIATOR IDENTIFIERS corresponding to the requesting initiator.

At most one ACC Page with PROXY bit equal to one shall be returned for a given ACC, whether or not that initiator has access rights granted under mulitple proxies by different initiators or granted under different INITIATOR IDENTIFIERS corresponding to the requesting Initiator.

6.0 ACCESS CONTROL OUT Command

The ACCESS CONTROL OUT command (see Table 12) is used to request service actions at a device server to limit or grant access to the device server 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.

TABLE 12. ACCESS CONTROL OUT command

Byte	Bit							
	7	6	5	4	3	2	1	0
0	OPERATION CODE (XXh)							
1	RESERVED			SERVICE ACTION				
2 9	RESERVED							
10	MSB							
13	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 fields in this command are found in the subclause for each service action.

6.1 ACCESS CONTROL OUT Service Actions

The ACCESS CONTROL OUT command service actions are defined in Table 13..

TABLE 13. ACCESS CONTROL OUT service actions

Code	Name	Description
00h	ACCESS ID ENROLL	Used by an initiator to enroll an AccessID at the device server. See 6.1.1
01h	MANAGE ACL	Used by an application client to add, change, remove entries in the device server's ACL. See 6.1.2
02h	PROXY ACCESS	Used by an initiator to grant or revoke a third party access to the logical unit or an ACC to which the requesting initiator already has access. See 6.1.3
03h	RESET AC	Used by an application client to reset the device server to the unconfigured AC state. See 6.1.4
04h-0Fh	Reserved	Reserved
10h-1Fh	Vendor-specific	Vendor-specific

6.1.1 ACCESS ID ENROLL service action (Mandatory)

The ACCESS ID ENROLL service action of the ACCESS CONTROL OUT command is used by an initiator to inform a device server of its AccessID. The device server shall use this information to maintain an association between the initiator and the AccessID. In this way commands coming from a given initiator can be

referred to the correct entry (or entries) in the ACL. 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.

The device server is required to maintain a temporary (volatile) record that this command was successful from the issuing initiator. That is, the initiator has the enrolled state. If the AccessID has any access rights at the device server, then the initiator and AccessID association shall also be maintained (in volatile memory). In this way, the device server can respond with correct sense information (INITIATOR NOT ENROLLED or INITIATOR NOT AUTHORIZED) to subsequent access restricted commands.

6.1.2 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 device server by initiators. This service action adds, changes or removes an entry or multiple entries in the device server's ACL. This service action can also be used to change the ACL Enabled/Disabled state or PTPL state of the device server and to otherwise manage the access control state of the device server. 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 device server in the Data-Out buffer. The structure of the data is as described in 6.1.2.1. 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 device server's access control 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 device server's access control state:

a) an ACC specification (ACC TYPE and ACC ADDRESS) is not valid at the device;

b) the PROXY field is one;

c) the INITIATOR TYPE field indicates an unsupported value;

d) the INITIATOR TYPE=01h (TransportID) and the INITIATOR IDENTIFIER field is invalid as specified in the relevant protocol standard;

e) the PTPL bit is one and the device server does not support non-volatile access control information.

If the device server cannot complete the command because it has insufficient resources to implement the command, it 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 device server's access control state.

6.1.2.1 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 6.1.2.1.1.

TABLE 14. MANAGE ACL parameter list format

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

The MANAGE ACL KEY is used to compare with the current Manage ACL Key maintained at the device server. If the MANAGE ACL KEY in the parameter list does not match the device server's current Manage ACL Key, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, additional sense data set to ACCESS DENIED - INVALID MANAGE ACL KEY and take no other action. If the device server successfully implements the requested service action, the device server resets its Manage ACL Key to the value specified in the NEW MANAGE ACL KEY field.

The PTPL (Persist Through Power Loss) bit of one instructs the device server to place all non-proxy access control information after successfully completing the current service action in non-volatile memory so that it can be restored after power cycles. This includes the Manage ACL Key. If this feature is not supported by the device server, it shall respond with CHECK CONDITION, sense key ILLEGAL REQUEST and additional sense code INVALID FIELD IN PARAMETER DATA and no changes to the current access control state are instantiated.

If the PTPL bit is zero, the device server shall only maintain in non-volatile memory the Access Restricted state bit (see 4.4).

The FLUSH bit of one instructs the device server to flush its current initiator/AccessID association list and its enrolled initiators list.

The CLEAR bit of one instructs the device server to completely clear its entire ACL (including proxies). The CLEAR bit of one shall also implicitly force a FLUSH action as specified above.

The ACL Enabled/Disabled state of the device server is set according to the value of the ENABLEDISABLE field, as described in Table 15.

TABLE 15. Global ENABLEDISABLE Codes.

Code	Description
00b	Leave unchanged the existing ACL Enabled/Disabled state of the device servert.
01b	Set the device server to the ACL Enabled state.
10b	Set the device server to the ACL Disabled state.
11b	Reserved

The ACL ENTRY PAGES that may follow in the parameter list provide additional changes to the access control state.

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 Entry Pages are implemented; if multiple ACL Entry Pages occur with conflicting instructions, the last such page shall take precedence.

6.1.2.1.1 ACCESS CONTROL OUT command ACL Entry page format

The ACL Entry page format for the MANAGE ACL and PROXY ACCESS service actions is given in Table 16.

TABLE 16. MANAGE ACL parameter list ACL Entry page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	PAGE LENGTH (*n*-1)							
2	RESERVED							REVOKE
3	ACC TYPE				RESERVED			PROXY
4	MSB							
7	ACC ADDRESS							LSB
8	RESERVED							
9	RESERVED							
10	IDENTIFIER TYPE							
11	IDENTIFIER LENGTH (*n*-11)							
12	MSB							
n	INITIATOR IDENTIFIER							LSB

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields are described in 4.2.1. The IDENTIFIER LENGTH is described in 5.1.1.1.1.

The ACC TYPE and ACC ADDRESS fields are described in 5.1.1.1.1.

The REVOKE bit of zero directs the device server to allow unrestricted access to the indicated ACC TYPE/ACC ADDRESS by the indicated initiator(s).

A REVOKE bit of one directs the device server to remove any matching ACL entries (as created by a previously successful MANAGE ACL service action). It is not an error condition if there are no matching ACL entries.

The PROXY bit shall be set to zero for the MANAGE ACL service action. The PROXY bit shall be set to one for the PROXY ACCESS service action.

In the case of an access right granted on the basis of a TransportID which might be invalid as a result of events in the service delivery subsystem, the following shall hold. Any such event which causes the device server to logout or otherwise determine that the TransportID may no longer be associated with the original initiator, shall also cause the device server to revoke that access right for that Initiator Identifier.

6.1.3 PROXY ACCESS service action (Optional)

The PROXY ACCESS service action of the ACCESS CONTROL OUT command is used by an initiator to grant or remove a third party access to logical unit or ACC within a logical unit to which the requesting initiator already has access.

Support for this service action is optional. If the device server does not support this service action, the server responds with CHECK CONDITION, sense key ILLEGAL REQUEST, and additional sense code INVALID FIELD IN CDB.

The parameter list shall contain a list of ACL Entry pages as described in 6.1.2.1.1 with the PROXY bit set to one. There is no header section of the parameter list for this service action.

Any of the following conditions in any parameter page 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 device server's ACL:

a) the ACC specification (ACC TYPE and ACC ADDRESS) is not valid at the device ;

b) the PROXY field is zero;

c) the INITIATOR TYPE field indicates an unsupported value;

d) the INITIATOR TYPE=01h (TransportID) and the INITIATOR IDENTIFIER field is invalid as specified in the relevant protocol standard.

If the initiator has no explicit or implicit non-proxy access rights to the ACC TYPE/ACC ADDRESS specified in any parameter page, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, and additional sense data set to ACCESS DENIED - INITIATOR NOT AUTHORIZED for enrolled initiators and set to ACCESS DENIED - INITIATOR NOT ENROLLED for all other initiators. Furthermore, the device server shall also make no changes to the device server's ACL.

Device servers shall treat proxy entries in a manner consistent with the MANAGE ACL service action, with the following exception. Proxy entries shall be maintained in volatile memory even if the PTPL state of the device server is active.

If the device server has no more resource available to instantiate the proxies, it shall return CHECK CONDITION, sense key ILLEGAL REQUEST with additional sense data of INSUFFICIENT ACCESS CONTROL RESOURCES and the ACL is restored to the state prior to receiving this command and service action.

6.1.4 RESET AC service action (Optional)

The RESET AC service action of the ACCESS CONTROL OUT command is used by an applicaiton client to return the device server to its unconfigured AC state where it is in the ACL Disabled state and PTPL disabled state, the Manage ACL Key is zero and the ACL is empty. Support for this service action is optional.

> NOTE: An alternative method to restore the device server to the unconfigured AC state is to use the MANAGE ACL service action with parameter list containing only the header with the following fields set: FLUSH=1, CLEAR=1, PTPL=0, ENABLEDISABLE=10b, NEW MANAGE ACL KEY=0 and MANAGE ACL KEY set to the current value.

For the RESET AC service action, the parameter list is described in 6.1.4.1. 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 device server to set the ACL Disabled state, disable the PTPL state, set the Manage ACL Key to zero, clear the ACL and flush its current initiator/AccessID association list and its enrolled initiator list.

6.1.4.1 RESET AC parameter list format

The format of the parameter list for an ACCESS CONTROL OUT command with RESET AC service action is shown in Table 17.

TABLE 17. RESET AC parameter list format

Byte	Bit							
	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 18. 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 18. ADDITIONAL PARAMETER DATA when VS=0

Byte	Bit							
	7	6	5	4	3	2	1	0
0	MSB							
7		MANAGE ACL KEY						LSB

The MANAGE ACL KEY is used to compare with the current Manage ACL Key maintained at the device server. If the MANAGE ACL KEY in the parameter list does not match the device server's current Manage ACL Key, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, additional sense data of ACCESS DENIED - INVALID MANAGE ACL 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 device server does not support the VS set to one, then the 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 Manage ACL Key for resetting the device to the unconfigured AC state and overriding the current Manage ACL Key.

A. Changes required in SAM-x.

This section contains some changes required in SAM-x to deal with Task Management in the presense 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 device server (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 presense 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 absense of any other error condition.

A.2 Additions for section 5.6.6

While the target 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 28 bytes long and is described in Table 19.

TABLE 19. TransportID for FCP.

Byte	Bit 7	6	5	4	3	2	1	0
0	RESERVED				PT_VAL	PA_VAL	PN_VAL	NN_VAL
1	MSB							
3	N_PORTID							LSB
4	RESERVED							
7								
8	MSB							
11	PROCESS ASSOCIATOR							LSB
12	MSB							
19	WWPORTNAME							LSB
20	MSB							
28	WWNODENAME							LSB

A PT_VAL bit of one indicates that the N_PORTID field is valid. Similarly, the PA_VAL, PN_VAL and NN_VAL bits of one indicate that the corresponding PROCESS ASSOCIATOR, 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. In the case of a private loop, the N_PORTID shall consist only of the AL_PA in the LSB byte of the field (the other bytes shall be zero). At least one of these validity bits must be set to one. If not, then the TransportID is invalid.

For the ACCESS CONTROL OUT command, the following apply. In the MANAGE ACL service action, the PT_VAL and PA_VAL bits must be zero. In the PROXY ACCESS service action, there are no such restrictions.

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 associations and enrollments, 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 associations and enrollments, and status for image pairs other than those that are logged out are not affected.

B.3. Additional rows required in Table 4:

TABLE 20.

	POWER	RESETLIP	LOGO, PLOGI	ABTS	PRLI, PRLO	TPRLO	TGTRESET	CLEAR	ABORT	LURESET
ACL and Manage ACL Key	Y[a]	N	N	N	N	N	N	N	N	N
Initiator/AccessID association/enrollment										
For all SCSI initiators	Y	Y	N	N	N	Y[b]	N	N	N	N
Only for SCSI initiator port initiating action	-	-	Y[6]	N	Y	-	N	N	N	N

a. When the most recent PTPL value received by the device server is zero.

b. 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 21.

TABLE 21. 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.

C.1. Volatility of the AccessID enrollments

AccessID enrollments (via the ACCESS CONTROL OUT command with ACCESS ID ENROLL service action) shall be invalidated by the following events or states:

a) power cycle of the device server;

b) hard reset bus condition.

D. 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 22. 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