Date: November 11, 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.

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. Vendors are free to implement any method they deem reasonable, but some options include hardware solutions like jumpers, 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 or questions[1]

2.1 Vendor-specific identifiers

An earlier draft had device server requirements when vendor specific initiator identifier types where used. I've left in the placeholders for such id types, but removed any specific mention of the device server responsibilities (e.g., there was language in 4.5 about initiators identified by VS identifiers and in REPORT ACL service action). Is this the right approach or should I put back some of that language?

2.2 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/access controllable component 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.

To address this problem, 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/access controllable component 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 ACL with respect to report proxy access rights clear enough?

2.3 Why not modified INQUIRY peripheral qualifier and block all other commands?

This question has been raised and in some respects has the advantage of simplifying a lot of the target's job. But there are some interesting consequences of doing this.

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

First, the target would need to maintain different INQUIRY data (albeit only a few bits) for different initiators. My impression is that this is unconventional as far as the standards are concerned.

Second, a single "access denied" type qualifier doesn't convey enough information for the initiator to determine if he needs to enroll (ACCESS ID ENROLL) or not. There are really three states for each initiator: access granted, not enrolled, enrolled but access denied. There's no way to convey this with simple PQs in INQUIRY data.

Third, to truly support this approach, an initiator during device discovery would first have to enroll his AccessID and then follow that with specific INQUIRY requests (or do the INQUIRY, check the PQ, and if needed, enroll and retry the INQUIRY). This would require fundamental modifications to device discovery code in host OS's driver stack. It might be possible to infiltrate this additional function in the HBA vendor's driver code, but the natural place for it is higher up in the driver discovery stack. This means getting OS vendors to implement this function.

Fourth, what is the behavior when an initiator has his rights revoked (UA with INQUIRY DATA HAS CHANGED?)?

Finally, this approach doesn't really work well with element-level access controls. An initiator might have rights to the logical unit, but not to some element, and it will be confusing to the initiator in this case (unless we change the behavior of READ ELEMENT STATUS and such).

Of course, removing AccessIDs and only using TransportIDs simplifies this problem, but complicates PAM's management job as mentioned in the introduction. We believe the current design achieves a reasonable balance between what PAM, each host and each target are required to do.

2.4 Override of Manage ACL Key

We've chosen to ignore the issue of how PAM can override the Manage ACL Key of a device server in the (hopefully) unlikely event that PAM forgets it (see the introduction). Is this the right approach or should we add languate to describe alternatives or should we mandate a specific override method (e.g., download firmware)?

2.5 Do we need additional UAs?

It was suggested that there might be a need for additional Unit Attention conditions.

Should the target set up UAs of the "I'VE LOST MY ACL" type after a power cycle when the PTPL feature is disabled? The author's feeling is that this is not necessary. First, this UA would have to get filtered back to PAM by some mechanism. It's not clear that even a filter driver of sorts would be able to see this UA after handling by lower driver levels. This information is of no use to the initiator itself, only to PAM. The initiator is already going to get power-on reset UA, so we've just added additional UAs with no real additional information. The initiator's commands will start to fail only if there were some access restrictions in place before the reset, so that should be sufficient information for the initiator (at a higher level in the driver stack) to generate some event to PAM to intervene.

Do we need an "ACCESS RIGHTS REVOKED" UA when a revocation occurs, either at the full logical unit level or for some element? Do we need an "ACCESS RIGHTS GRANTED" UA either for the full logical unit or if the initiator gets rights to an element? This might be a way to help trigger the initiator to enroll, but the question remains whether an initiator who thought he had no access to a logical unit would ever send a command in order to receive the UA.

Are there other conditions which might justify UAs of some sort or another?

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 component of a logical unit on which access control can be independently managed. The logical unit itself and 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 access controllable component combinations which indicate access rights at a logical unit.

AccessID: An identifier used for granting or revoking access rights to an access controllable component of 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 an access controllable component of a logical unit in which the Access Control List is ignored during command processing for commands which reference that access controllable component.

ACL Enabled: The state of an access controllable component of a logical unit in which the Access Control List is checked during command processing for commands which reference that access controllable component.

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 an access controllable component of a logical unit.

3.2 Acronyms

ACL: Access Control List

4.0 Access Controls

Access controls may be used to limit the set of initiators which can execute certain commands at a device server. The device server shall reject certain commands from initiators outside the specified set. Initiators can be identified uniquely by an access identifier, called an AccessID, as defined in this proposal or by protocol-specific identifiers defined in the relevant protocol addendums.

An application client may add or remove initiators from the selected set 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 an access controllable component. These are one of the following:

a) **logical unit** - a logical unit access control restricts access to the entire logical unit; and

b) **element -** an element access control restricts access to a specified element within a medium changer.

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 commands are not subject to reservation conflicts.

AUTHOR'S NOTE: *Does the above statement require additional changes to the SPC Model Reservations subclause? This is repeated in the description of the ACCESS CONTROL IN and ACCESS CONTROL OUT commands - is it still needed there as well?*

The default state of the device is to allow unrestricted access to the logical unit by all initiators (subject to the existing protocols). That is, every access controllable component of the logical unit is in the ACL Disabled state. This remains the case until the first successful completion of an ACCESS CONTROL OUT command with MANAGE ACL service action. This same service action can selectively change the ACL Enabled/Disabled state for any access controllable component of the logical unit.

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.

Along with the ACL, the device server shall maintain a Manage ACL Key of 8 bytes (64 bit integer). The default state for this value is zero. The value is managed by ACCESS CONTROL OUT commands with MANAGE ACL service actions.

Additionally, the device server shall maintain an Access Restricted state bit. This state bit shall be zero if all access controllable components of the logical unit are in the ACL Disabled state. This state bit shall be one if any access controllable component of the logical unit is in the ACL Enabled state. This Access Restricted state bit shall be preserved across power cycles. The default value for this state bit is zero.

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.

4.1 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 commands 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 will 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 not be stored in non-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 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 5.1.1.1.3).

4.2 Granting and revoking access rights

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

A service action (MANGE ACL) can grant or revoke access rights for any specific access controllable component of the logical unit to an initiator based on the AccessID identifier or TransportID identifier (if applicable). This same service action can selectively change the ACL Enabled/Disabled state for any access controllable component of the logical unit. It can request the access control information persist through power loss, or disable this function.

A service action (PROXY ACCESS) can grant or revoke access rights for any specific access controllable component of the logical unit to a third party initiator. This service action is valid only if the requesting initiator already has access rights to the specified access controllable component. 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.

An initiator’s access right to an access controllable component of 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 are revoked, the following cleaning actions shall be applied:

a) all tasks from that initiator on that logical unit shall be aborted;

b) any CA or ACA state for which this is the faulting initiator shall be cleared.

4.3 Preserving access control information

A device server is required to maintain in non-volatile form the Access Restricted state bit which indicates whether the logical unit is unconstrained (ACL Disabled for all access controllable components) or constrained in any way (ACL Enabled for any access controllable component). These state flags 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:

a) if the Access Restricted state bit is one prior to power off, the device server shall set the logical unit to the ACL Enabled state and set all other access controllable components to the ACL Disabled state until new access control information is transmitted to the device server using the ACCESS CONTROL OUT command with MANAGE ACL service action;

b) if the Access Restricted state bit is zero prior to the power off, the device server shall set the logical unit and all other access controllable components to 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 will 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 11, 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. Specifically, an event that will 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.

4.4 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 list of access controllable components in the ACL Enabled state;

c) the list of access controllable component/initiator identifier combinations for which access is currently granted; this includes proxy entries.

If applicable, the requesting application client may compare the results of this service action with the results of a READ ELEMENT STATUS command to determine all access controllable components for which access control is currently disabled.

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

4.5 Verifying access rights for initiators

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

If any of the these conditions hold with respect to all access controllable components referenced by the command, the command is handled in the usual way:

a) access control is disabled;

b) the initiator has enrolled an AccessID and the AccessID has access rights (proxy or non-proxy);

c) the initiator has a TransportID (if applicable) which has access rights (proxy or non-proxy);

d) the initiator has an active proxy access right 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 code of ACCESS DENIED. The additional sense code qualifier is 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 ASCQ is set to INITIATOR NOT AUTHORIZED;
2. if the initiator has not enrolled an AccessID, then the ASCQ is set to INITIATOR NOT ENROLLED.

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

4.5.1 Access rights to elements

For a device server with elements, the device server may optionally allow for access control at the element level within the logical unit. Note that access control can be disabled either at the full logical unit and/or at each element within the logical unit.

For a device server which supports element-based access control, the following rules apply.

a) An access right granted an initiator to an element within a logical unit also implicitly grants that initiator access to the logical unit.

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

c) An initiator request to a logical unit which does not reference any specific elements within the logical unit (e.g., READ BUFFER, INITIALIZE ELEMENTS) is restricted only by access control at the logical unit.

d) An initiator request to a logical unit which references one or more particular elements is restricted by the access control at all of the elements referenced.

4.6 Access Control Service Actions

Table 1 gives a summary list of the access control service actions.

TABLE 1. Access Control Commands and Service Actions

Code	Name	Section
ACCESS CONTROL IN (OPCODE 86h)		
00h	REPORT ACL	5.1.1
01h	REPORT INITIATOR ACL	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-0Fh	Reserved	
10h-1Fh	Vendor-specific	

4.7 Access Control Additional Sense Codes

Table 2 contains a list of the Additional Sense Code and Additional Sense Code Qualifiers relevant to access control.

TABLE 2. Access Control Additional Sense Codes and Qualifiers

ASC	ASCQ	Name	Function
XXh	00h	ACCESS DENIED	Initiator is not sufficiently authorized to make the request.
XXh	XXh	ACCESS DENIED - INITIATOR NOT ENROLLED	Initiator has not sent an ACCESS ID ENROLL service action.
XXh	XXh	ACCESS DENIED - INITIATOR NOT AUTHORIZED	A enrolled initiator has access permissions insufficient for the requested command
XXh	XXh	ACCESS DENIED - INVALID MANAGE ACL KEY	The Manage ACL Key value is does not match the Manage ACL Key maintained at the device server
XXh	XXh	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 3) 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 3. 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 4.

TABLE 4. 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 ACL	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 code of ACCESS DENIED and additional sense code qualifier of 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 5. The ENABLED AND ENTRY PAGE(S) are described in 5.1.1.1.1 and 5.1.1.1.2.

TABLE 5. REPORT ACL parameter data format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	RESERVED							
1	RESERVED							PTPL
2	MSB							
3	RESOURCE UTILIZATION							LSB
4	MSB							
7	ADDITIONAL LENGTH (*n*-7)							LSB
8								
n	ENABLED AND 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 RESOURCE UTILIZATION field shall indicate the total number of entries in the device server's ACL currently used (this is the same as the number of Entry pages available to be returned in the parameter data). 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).

5.1.1.1.1 REPORT ACL parameter data ACL Enabled page format

The ACL Enabled page format for the REPORT ACL service action is specified in Table 6.

TABLE 6. REPORT ACL parameter data ACL Enabled page format

Byte	Bit 7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	PAGE LENGTH (06h)							
2	RESERVED							
3	SCOPE				RESERVED			
4	MSB							
7	SCOPE-SPECIFIC ADDRESS							LSB

One such page shall be returned for each access controllable component within the logical unit that is in the ACL Enabled state.

The SCOPE and the SCOPE-SPECIFIC ADDRESS fields specify the access controllable component within the logical unit to which this page refers. The valid SCOPE codes are defined in the PERSISTENT RESERVATION IN clause of SPC-2, rev 11, 7.12.3.1. If the SCOPE field indicates the full logical unit (value 0h), then the SCOPE-SPECIFIC ADDRESS field shall be zero. If the SCOPE field indicates a subcomponent of the logical unit (that is, SCOPE not equal to 0h), then the SCOPE-SPECIFIC ADDRESS field is as specified in the aforementioned clause of PERSISTENT RESERVATION IN.

5.1.1.1.2 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 (01h)							
1	PAGE LENGTH (*n*-1)							
2	RESERVED							
3	SCOPE				RESERVED			PROXY
4	MSB							
7	SCOPE-SPECIFIC ADDRESS							LSB
8	RESERVED							
9	RESERVED							
10	IDENTIFIER TYPE							
11	IDENTIFER LENGTH (*n*-11)							
12	MSB							
n	INITIATOR IDENTIFIER							LSB

The SCOPE and the SCOPE-SPECIFIC ADDRESS fields are described in 5.1.1.1.1 and specify the access controllable component to which the specified initiator identifier has been granted access.

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 access controllable component 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 5.1.1.1.3. 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 access controllable component and INITIATOR IDENTIFIER combination.

One Entry page with PROXY bit equal to one shall be returned for a given access controllable component 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.

5.1.1.1.3 Identifier Type and Initiator Identifier

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields in an ACL Entry parameter page for an ACCESS CONTROL IN command with service action REPORT ACL or an ACCESS CONTROL OUT command with service action MANAGE ACL or PROXY ACCESS have the meaning described in Table 8.

TABLE 8. 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.*

5.1.2 REPORT INITIATOR ACL service action (Optional)

The REPORT INITIATOR ACL 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 ACL service action is reserved.

The format of the returned data shall conform to the specification in 5.1.2.1. Also, the device server will not return any initiator-specific entries for an access controllable component in the ACL Disabled state.

The device server shall respond with a summary of the initiator's relevant ACL entries for any access controllable component in the ACL Enabled state. This includes the following:

a) any entries involving the AccessID (IDENTIFIER TYPE=00h) corresponding to the requesting initiator as enrolled by an ACCESS CONTROL OUT command with service action ACCESS ID ENROLL;

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.

b) any entries involving TransportIDs corresponding to the requesting initiator (IDENTIFIER TYPE=01h);

The returned data shall not return any such entries for logical unit access controllable components to which the initiator has rights derived implicitly from rights granted to an access controllable subcomponent (see 4.5.1).

If the initiator has no access rights at the device server, then the device server shall return only the required header information.

AUTHOR'S NOTE: *The returned list does not reflect any access controllable component for which access controls are disabled (and so for which the initiator has access). It's unclear to me how exactly to handle this. We could include the Enabled pages as for REPORT ACL (so he knows he has access to the access controllable components NOT listed) or the converse, namely the pages for access controllable components in the ACL Disabled state.*

5.1.2.1 REPORT INITIATOR ACL parameter data format

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

TABLE 9. REPORT INITIATOR ACL parameter data format

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

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).

5.1.2.1.1 REPORT INITIATOR ACL parameter data ACL Initiator Entry page format

The REPORT INITIATOR ACL service action ACL Initiator Entry parameter data page format is specified in Table 10.

TABLE 10. REPORT INITIATOR ACL parameter data Initiator Entry page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PAGE CODE (02h)							
1	PAGE LENGTH (06h)							
2	RESERVED							
3	SCOPE				RESERVED			PROXY
4	MSB							
7	SCOPE-SPECIFIC ADDRESS							LSB

All fields in this parameter page are as defined in 5.1.1.1.1 and 5.1.1.1.2.

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

At most one Entry page with PROXY bit equal to one shall be returned for a given access controllable component, 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.

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

6.0 ACCESS CONTROL OUT Command

The ACCESS CONTROL OUT command (see Table 11) is used to request service actions at a device server to limit or grant access to the device server by 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 11. 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.

6.1 ACCESS CONTROL OUT Service Actions

The ACCESS CONTROL OUT command service actions are defined in Table 12..

TABLE 12. 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 an access controllable component to which the requesting initiator already has access. See 6.1.3
03h-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 will 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 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 disable or reenable access control at a specific access controllable component 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 will 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 will be transferred. This is not an error condition and shall result in no changes to the device servers 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) the access controllable component specification (SCOPE and SCOPE-SPECIFIC ADDRESS) is not valid at the device or the device server does not support access control on this access controllable component;

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 code of INSUFFICIENT ACCESS CONTROL RESOURCES. In this case, the device server shall restore its access control state to the state prior to receiving this command.

If the device server is currently in its default state (namely, every access controllable component of the logical unit is in the ACL Disabled state and the Manage ACL Key is zero), receipt of this command shall first set the logical unit to the ACL Enabled state and all other access controllable components shall be set to the ACL Disabled state, then instantiate access control as specified in the parameter list.

If a MANAGE ACL service action causes the complete revocation of access rights from an initiator, the actions listed under 4.2 shall be applied to that initiator.

6.1.2.1 MANAGE ACL parameter list format

The format of the parameter list provided for an ACCESS CONTROL OUT command is shown in Table 13. The ENABLE/DISABLE AND ENTRY PAGE(S) are described in 6.1.2.1.1 and 6.1.2.1.2.

TABLE 13. 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	RESERVED							PTPL
18	RESERVED				FLUSH	CLEAR	ENABLEDISABLE	
19	RESERVED							
20 *n*	ENABLE/DISABLE AND 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 code ACCESS DENIED and additional sense code qualifier set to 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 4.3).

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 ACL Enabled/Disabled state for each access controllable component of the logical unit are set according to the value of the ENABLEDISABLE field, as described in Table 14. The ENABLE/DISABLE AND ENTRY PAGES that fol-

low in the parameter list provide a new access control state. The CLEAR bit of one will also implicitly force a FLUSH action as specified above.

TABLE 14. Global ENABLEDISABLE Codes.

Code	Description
00b	Leave unchanged the existing ACL Enabled/Disabled state for each access controllable component of the logical unit.
01b	Set the logical unit to the ACL Enabled state and set each subcomponent access controllable component to the ACL Disabled state (if applicable).
10b	Set each access controllable component of the logical unit to the ACL Disabled state.
11b	Reserved

NOTE: The device server can easily be restored to its default unconstrained state by a MANAGE ACL service action with parameter list containing only the header with the following fields set: FLUSH=1, CLEAR=1, ENABLEDISABLE=10b, NEW MANAGE ACL KEY=0.

6.1.2.1.1 MANAGE ACL parameter list ACL Enable/Disable page format

The ACL Enable/Disable page format for the MANAGE ACL service action is specified in Table 15.

TABLE 15. MANAGE ACL parameter list ACL Enable/Disable page format

	Bit							
Byte	7	6	5	4	3	2	1	0
0	PAGE CODE (00h)							
1	PAGE LENGTH (06h)							
2	RESERVED					CLEAR	ENABLEDISABLE	
3	SCOPE				RESERVED			
4	MSB							
7	SCOPE-SPECIFIC ADDRESS							LSB

The SCOPE and SCOPE-SPECIFIC ADDRESS fields are defined in 5.1.1.1.1.

The CLEAR bit of one instructs the device server to completely clear its ACL for all access rights (including proxies) for the specified access controllable component. After this clear action, the value of the ENABLEDISABLE field dictates the enable/disable controls state for the specified access controllable component as described in Table 16.

TABLE 16. ENABLEDISABLE Codes

Code	Description
00b	Leave unchanged the existing ACL Enabled/Disabled state for the specified access controllable component.
01b	Set the access controllable component to the ACL Enabled state.
10b	Set the access controllable component to the ACL Disabled state.
11b	Reserved

If the parameter list contains two or more ACL Enable/Disable pages with conflicting instructions, the last such page shall take precedence. This is not an error condition.

6.1.2.1.2 MANAGE ACL parameter list ACL Entry page format

The ACL Entry page format for the MANAGE ACL service action is given in Table 17.

TABLE 17. MANAGE ACL parameter list ACL Entry page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PAGE CODE (01h)							
1	PAGE LENGTH (*n*-1)							
2	RESERVED							REVOKE
3	SCOPE				RESERVED			
4	MSB							
7	SCOPE-SPECIFIC 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 5.1.1.1.3. The IDENTIFIER LENGTH is described in 5.1.1.1.2.

The SCOPE and SCOPE-SPECIFIC ADDRESS fields are described in 5.1.1.1.1.

The REVOKE bit of zero directs the device server to allow access to the indicated access controllable component by the indicated initiator(s) without access control restrictions.

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.

In the case of an access right granted on the bases 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.

If the parameter list contains two or more Entry pages with conflicting instructions, the last such page shall take precedence. This is not an error condition.

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 an access controllable component of a device server 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 will contain a list of ACL Proxy Entry pages as described in 6.1.3.1. 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 access controllable component specification (SCOPE and SCOPE-SPECIFIC ADDRESS) is not valid at the device or the device server does not support access control on this access controllable component;

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 access rights to the access controllable component specified in any parameter page, the device server shall return CHECK CONDITION with sense key ILLEGAL REQUEST, additional sense code of ACCESS DENIED and qualifier set to INITIATOR NOT ENROLLED for unenrolled initiators and set to INITIATOR NOT AUTHORIZED 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 will not be maintained in non-volatile memory even if PTPL state of the device servicer 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 code of INSUFFICIENT ACCESS CONTROL RESOURCES and the ACL is restored to the state prior to receiving this command and service action.

If the PROXY ACCESS service action causes the complete revocation of access rights from an initiator, the actions listed under 4.2 shall be applied to that initiator.

6.1.3.1 PROXY ACCESS parameter list ACL Proxy Entry page format

The ACL Proxy Entry page format for the PROXY ACCESS service action is given in Table 18.

TABLE 18. PROXY ACCESS parameter list ACL Proxy Entry page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PAGE CODE (02h)							
1	PAGE LENGTH (*n*-1)							
2	RESERVED							REVOKE
3	SCOPE				RESERVED			PROXY
4	MSB							
7	SCOPE-SPECIFIC ADDRESS							LSB
8	RESERVED							
9	RESERVED							
10	IDENTIFIER TYPE							
11	IDENTIFIER LENGTH (*n*-11)							
12	MSB							
n	INITIATOR IDENTIFIER							LSB

The PROXY bit shall be one in this page for the PROXY ACCESS service action.

The IDENTIFIER TYPE and INITIATOR IDENTIFIER fields are described in 5.1.1.1.3. The IDENTIFIER LENGTH is described in 5.1.1.1.2.

The SCOPE and SCOPE-SPECIFIC ADDRESS fields are described in 5.1.1.1.1.

Revocation of a proxy occurs when the REVOKE bit is one. It is not an error condition to receive the revocation if there is no existing proxy for the indicated third party by the requesting initiator.

In the case of a proxy access right granted on the bases 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 third party initiator, shall also cause the device server to revoke all existing proxies for that third party initiator.

Any access-restricted command by the third party after revocation of his proxy will be treated as if no proxy had ever been in place.

If the parameter list contains two or more Proxy Entry pages with conflicting instructions, the last such page shall take precedence. This is not an error condition.

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 will 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 24 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	MSB							
7	PROCESS ASSOCIATOR							LSB
8	MSB							
15	WWPORTNAME							LSB
16	MSB							
23	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	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.