**Date:** August 30, 2006

**To:** T10 Committee (SCSI)

**From:** Roger Cummings (Symantec), Krithivas Ramamurthy (Intel), Drue Reeves (Dell) on behalf of the SNIA Management Protocol (MP) Technical Working Group (TWG)

**Subject:** T10/06-xxxr0 Request for Additional Security Protocol Value Allocation

**Revision History**

| | | |
|---|---|---|
| | (August 15, 2006) | 1st SNIA Revision |
| | (August 17, 2006) | 2nd SNIA Revision - added details of proxies in the overview section |
| | (August 21, 2006) | 3rd SNIA Revision - add full details of the suggested changes. Amended transfer sizes to work with ATA as well as SCSI. Added more discovery requirements. |
| | (August 22, 2006) | 4th SNIA Revision - Changed overview for better flow, corrected typos. Revision submitted for SNIA Ballot. |
| 06-392r0 | (August 29, 2006) | 1st T10 version - Changes made as a result of SNIA ballot comments. |

**Background**

This proposal is the result of experience with two generations of implementations of the Storage Management protocol (SMI-S) that is defined in ANSI/INCITS 388-2004. It has been developed in the Storage Network Industry Association (SNIA) by a number of organizations that participate in the SNIA Management Protocol Technical Working Group (TWG), and has been formally approved by that group.

**Overview**

Two architectures predominate for supporting SMI-S in the storage area today, as follows:

- Proxy-based -- in which the Common Information Model (CIM) management software required by SMI-S is hosted on a separate server from the storage being managed;
- Direct or native -- in which the CIM management software is contained within the storage device.

Both architectures have advantages and disadvantages, and both can be optimized by further standardization work as described below.

In the proxy architecture, the CIM management software stack resides on a server. The management application sends the management commands to the “proxy server”, which in-turn invokes a vendor unique “plug-ins” that translates management commands in both directions between the protocols defined by SMI-S and legacy, proprietary or other standard interconnects. A specific example of the last of these is given in SMI-S as a disk array controller connected via SCSI. Creating these vendor unique plug-ins has proven difficult in many cases, and has acted as a significant inhibitor to the adoption of SMI-S.

Experience gained in SNIA plugfests and in the field has also found these proxy servers to be less than ideal. They have been identified as the source of several performance and interoperability problems, and these are mostly believed to stem from a lack of detailed knowledge of the devices being managed on the part of the designers of the proxy implementations.

There are a number of different architectures for CIM software, each with their own plugin architecture, and thus a storage device manufacturer will often have to create multiple types of plugin to ensure that their device can be supported by all of the types of proxy server that will be found in the field. This is a sizable design task that is often contracted out by the storage manufacturer, thus compounding the knowledge problem described above.

A much better situation would be one where the CIM software implementer was able to create a single plug-in that was capable of enabling management of any storage device, regardless of manufacturer. But in order for this to be possible, the interface and protocol between the plugin and the storage device has to be standardized.

This document proposes that a standard method of transporting the CIM-XML protocol (defined by the DMTF) over SCSI be created for this purpose. With this approach, the plugin becomes a become a "pass through" of CIM data to the storage device, and thus multiple storage device manufacturers will be able to share use of a single plugin instance.

In a direct or native architecture, the CIM software and "plug-in" are embedded in the storage device and are not distinguishable as separate entities. In fact, a subset of required functionality can be implemented by the storage device.

Conversations with device manufacturers have identified two specific impediments to implementing the direct model for SMI-S in both legacy devices and some classes of devices under development. These impediments are:

a) The need to provide an Ethernet port for connection to a management station;
b) The need to implement a complete TCP/IP stack (often within a unique embedded environment) within a device.

Both of these impediments could be overcome if a method of transporting the existing CIM-XML protocol defined by the DMTF across a SCSI infrastructure could be created. The existing XML structures would essentially be transported unchanged, but a simple additional XML "wrapper" might be needed to handle sequencing etc.

Note that while it would be possible to use existing products to use "Internet Protocol over Fibre Channel" (IP over FC), as defined in IETF RFC to carry SMI-S traffic over FC, this would only address the first impediment identified above, and not the second. Thus it is believed a new solution is needed that addresses both impediments in order to convince device manufacturers that supporting SMI-S is feasible.

In current SMI-S implementations, the CIM-XML protocol is transported over HTTP, and then over TCP/IP. SMI-S defines use of HTTP authentication, and Secure Sockets Layer (SSL) and Transport Layer Security (TLS) for various security features. It is expected that the SNIA definition for CIM-XML over SCSI would require use of one or more of the protocols supports by the SCSI SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands to provide equivalent or better authentication, and other security features.

**Proposal**

The document proposes that a method be defined in SPC-4 of allowing CIM-XML to be transported across a SCSI infrastructure. The specific requirements are:

a) Provide a method of sending up to 16 megabytes of arbitrary data from an Initiator to a Target;
b) Provide a method of sending up to 16 megabytes of arbitrary data from a Target to an Initiator;
c) Provide a method of allowing a Target to notify an Initiator of the completion of an "event";
d) Provide a method by which an Initiator can discover that a Target supports a) thru c) above using the facilities provided by the FC-HBA API (INCITS 386:2004) or the later SM-HBA API.

After discussing the above with several interested parties in T10, and despite the obvious clash with the command names, this proposal proposes that the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands be used to fulfil the above requirements. This approach has been followed because:

a) It will avoid consuming other code points (such as new Service Actions for the MAINTENANCE IN and MAINTENACE OUT commands) to define functionality this is very similar to that provided by the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands.
b) It will provide a simple path to allowing the same functionality to be defined in ATA.

Thus this proposal specifically proposes:

1) A contiguous range of codes in the SECURITY PROTOCOL field in Tables 172 and 177 be described as "Defined by SNIA". Two such codes are required by this proposal. One code will be used for all transport operations, and the second code used for the protocol that monitors the Target for completed Event Notifications.
2) A new MANAGEMENT well known logical unit value be assigned. SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands containing the codes in the SECURITY PROTOCOL field identified in 1) above will be defined as being addressed to this logical unit.

The specific code points identified below are suggestions only, and are of course at the discretion of T10 to assign. SNIA will be pleased to utilize any method of meeting the four requirements identified at the beginning of this paragraph.

**Suggested Changes**

## 3.1 Definitions

**3.1.x Storage Networking Industry Association (SNIA)**: An organization that develops and promotes standards for storage management and other storage-related functions, and which publishes standards via the INCITS Fast Track process. See http://www.snia.org.

### 6.27.1 SECURITY PROTOCOL IN command description

The SECURITY PROTOCOL IN command (see table 172) is used to retrieve security protocol information (see 6.27.2) or the results of one or more SECURITY PROTOCOL OUT commands (see 6.28).

**Table 172 — SECURITY PROTOCOL IN command**

| Bit / Byte | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | OPERATION CODE (A2h) | | | | | | | |
| 1 | SECURITY PROTOCOL | | | | | | | |
| 2 | SECURITY PROTOCOL SPECIFIC | | | | | | | |
| 3 | | | | | | | | |
| 4 | INC_512 | Reserved | | | | | | |
| 5 | Reserved | | | | | | | |
| 6 | (MSB) | ALLOCATION LENGTH | | | | | | |
| 9 | | | | | | | | (LSB) |
| 10 | Reserved | | | | | | | |
| 11 | CONTROL | | | | | | | |

The SECURITY PROTOCOL field (see table 173) specifies which security protocol is being used.

**Table 173 — SECURITY PROTOCOL field in SECURITY PROTOCOL IN command**

| Code | Description | Reference |
|---|---|---|
| 00h | Security protocol information | 6.27.2 |
| 01h - 06h | Defined by the TCG | 3.1.122 |
| 07h - 1Fh | Reserved | |
| 20h | Tape Data Encryption | SSC-3 |
| 21h - 3Fh | Reserved | |
| 40h - 45h | Defined by the SNIA | 3.1.x |
| 46h - EDh | Reserved | |
| EEh | Authentication in Host Attachments of Transient Storage Devices | IEEE P1667 |
| EFh | ATA Device Server Password Security | TBD |
| F0h - FFh | Vendor Specific | |

Editors Note 1 - ROW: SECURITY PROTOCOL field code values 21h – 2Fh are tentatively reserved for SSC-x uses

## 6.28 SECURITY PROTOCOL OUT command

The SECURITY PROTOCOL OUT command (see table 177) is used to send data to the logical unit. The data sent specifies one or more operations to be performed by the logical unit. The format and function of the operations depends on the contents of the SECURITY PROTOCOL field (see table 178). Depending on the protocol specified by the SECURITY PROTOCOL field, the application client may use the SECURITY PROTOCOL IN command (see 6.27) to retrieve data derived from these operations.

**Table 177 — SECURITY PROTOCOL OUT command**

| Bit / Byte | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | OPERATION CODE (B5h) | | | | | | | |
| 1 | SECURITY PROTOCOL | | | | | | | |
| 2 | SECURITY PROTOCOL SPECIFIC | | | | | | | |
| 3 | | | | | | | | |
| 4 | INC_512 | Reserved | | | | | | |
| 5 | Reserved | | | | | | | |
| 6 | (MSB) ALLOCATION LENGTH | | | | | | | |
| 9 | | | | | | | | (LSB) |
| 10 | Reserved | | | | | | | |
| 11 | CONTROL | | | | | | | |

The SECURITY PROTOCOL field (see table 178) specifies which security protocol is being used.

**Table 178 — SECURITY PROTOCOL field in SECURITY PROTOCOL OUT command**

| Code | Description | Reference |
|---|---|---|
| 00h | Reserved | |
| 01h - 06h | Defined by the TCG | 3.1.122 |
| 07h - 1Fh | Reserved | |
| 20h | Tape Data Encryption | SSC-3 |
| 21h - 3Fh | Reserved | |
| 40h - 45h | Defined by the SNIA | 3.1.x |
| 46h - EDh | Reserved | |
| EEh | Authentication in Host Attachments of Transient Storage Devices | IEEE P1667 |
| EFh | ATA Device Server Password Security | TBD |
| F0h - FFh | Vendor Specific | |

Editors Note 2 - ROW: SECURITY PROTOCOL field code values 21h – 2Fh are tentatively reserved for SSC-x uses.

The contents of the SECURITY PROTOCOL SPECIFIC field depend on the protocol specified by the SECURITY PROTOCOL field (see table 178).

# 8 Well known logical units

## 8.1 Model for well known logical units

Well known logical units are addressed using the well known logical unit addressing method of extended logical unit addressing (see SAM-4). Each well known logical unit has a well known logical unit number (W-LUN) as shown in table 363.

**Table 363 — Well known logical unit numbers**

| W-LUN | Description | Reference |
|---|---|---|
| 00h | Reserved | |
| 01h | REPORT LUNS well known logical unit | 8.2 |
| 02h | ACCESS CONTROLS well known logical unit | 8.3 |
| 03h | TARGET LOG PAGES well known logical unit | 8.4 |
| 04h | SECURITY PROTOCOL well known logical unit | 8.5 |
| 05h | MANAGEMENT well known logical unit | 8.6 |
| 06h-FFh | Reserved | |

If a well known logical unit is supported within a SCSI target device, then that logical unit shall support all the commands defined for it.

Access to well known logical units shall not be affected by access controls.

The SCSI target device name of the well known logical unit may be determined by issuing an INQUIRY command (see 6.4) requesting the Device Identification VPD page (see 7.6.3).

All well known logical units shall support the INQUIRY command's Device Identification VPD page as specified in 7.6.3.2.2.

## 8.5 SECURITY PROTOCOL well known logical unit

The SECURITY PROTOCOL well known logical unit shall only process the commands listed in table 418. If a command is received by the SECURITY PROTOCOL well known logical unit that is not listed in table 418, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID COMMAND OPERATION CODE.

**Table 418 — Commands for the SECURITY PROTOCOL well known logical unit**

| Command name | Operation code | Type | Reference |
|---|---|---|---|
| INQUIRY | 12h | M | 6.4 |
| REQUEST SENSE | 03h | M | 6.26 |
| TEST UNIT READY | 00h | M | 6.30 |
| SECURITY PROTOCOL IN [a] | A2h | M | 6.27 |
| SECURITY PROTOCOL OUT [a] | B5h | M | 6.28 |
| Key: M = Command implementation is mandatory.<br>[a] Commands with Security Protocol values other than those described as "Defined by the SNIA" | | | |

## 8.6 MANAGEMENT well known logical unit

The MANAGEMENT well known logical unit shall only process the commands listed in table 419. If a command is received by the MANAGEMENT well known logical unit that is not listed in table 419, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID COMMAND OPERATION CODE.

Table 419 — Commands for the MANAGEMENT well known logical unit

| Command name | Operation code | Type | Reference |
|---|---|---|---|
| INQUIRY | 12h | M | 6.4 |
| REQUEST SENSE | 03h | M | 6.26 |
| TEST UNIT READY | 00h | M | 6.30 |
| SECURITY PROTOCOL IN [a] | A2h | M | 6.27 |
| SECURITY PROTOCOL OUT [a] | B5h | M | 6.28 |

Key: M = Command implementation is mandatory.
[a] Commands with Security Protocol values described as “Defined by the SNIA” only