To: INCITS Technical Committee T10
From: Kevin Butt
Date: September 4, 2007 7:29 pm
Document: T10/07-374r2 — SSC-3: End-to-end Logical Block Protection

I. Revisions

07-374r0: Initial revision (August 16, 2007) using SSC-3r03d as base.

07-374r1: Incorporate feedback received from Paul Entzel, Dave Peterson, and Gerry Houlder. Due to the large number of changes, change bars are not useful and have been removed.

a. Delete descriptions for FC_CRC and RS_CRC fields.

b. Remove protection of parameter data and CDB - Belongs in SPC-4 or SAM and may be covered by other proposals currently in work in CAP.

c. Added explanation of benefit over protection information applications may already embed in their data.

d. Deleted several sentences and phrases pointed to that added no value or that were likely to not survive letter ballot.

e. Changed a “should” to a “shall” on application client validation and generation of protection information when configured to do so.

f. Some editorial modifications.

g. Added clarifying text about when protection information on data sent in a write is to be validated with respect to the command status.

h. Clarified when protection information on read and on write is required to be supported (i.e., I put it in terms of mode page fields).

i. Removed deferred check condition reporting on Read command when protection information validation fails.

j. Added which sense key to use on reporting validation failure.

k. Deleted the CDBTL field.

l. Added section and figure to describe how protection information is recorded to the medium.

m. Added sections and figures showing the protection information on logical blocks and how the data is laid out for the RECOVER BUFFERED DATA command.

n. Added text to specify the values of the length and endian for each protection method specified.

o. Pointed to updated presentation document 07-373r1.

p. Specified the specific commands that add protection information during transfers.

EDITOR'S NOTE: Paul Entzel was concerned that I be consistent in my use of "device" versus "device server", especially in the model clause. I have changed the specific location he pointed to but I still use both terms. Please watch to make sure they are being used correctly and consistently.

EDITOR'S NOTE: Gerry Houlder suggested the mode page policy for the Control Data Protection mode page be changed from Per I_T nexus to Shared. I rejected this suggestion because there is not much more complexity by using Per I_T nexus and it removes that possibility of application clients changing it on each other. I have included this note in case others wish to argue for this change.

07-374r2: Incorporate feedback from Bob Nixon and mark the text that could be removed by options currently listed that will likely be removed if nobody argues to keep them. Removed the RDPR and RBDPR fields. Changed the term "end-to-end logical block protection" to "logical block protection".

II. Introduction

KEY:

~~Deleted Text~~

Added Text

Updates to added text

Potential Removal if nobody fights to keep

EDITOR'S NOTE: <Text>

Questions

Please see SSC-3: Tape end-to-end data protection (Presentation) (07-373r2) as the introduction to this proposal.

IBM Tape has been required to address the question of why our tape drives do not support the T10 standard end-to-end data protection that is available for disk drives. While we have been able to show that it is a disk drive centric solution and that it does not work for tapes, we then have to address the issue of this being one more reason disk should replace tape. That is, disk is viewed as more reliable than tape. We believe that because disk devices have a standard end-to-end data protection they are given an additional marketing tool to win over a tape solution.

IBM believes that SSC-3 needs to provide an end-to-end data protection for tape devices. IBM enterprise drives have been using a proprietary method of end-to-end data protection for over 12 years and we believe that this concept can be easily adapted to fit into a standard.

This end-to-end data protection is accomplished by adding a 32 bit CRC to each data block at the host and transferring that CRC along with the data and validating and storing that CRC with the data on media.

III. Proposal Against SSC-3

3.1.a little-endian byte ordering: For a field consisting of more than one byte and containing a single value, the byte containing the MSB is stored at the highest address and the byte containing the LSB is stored at the lowest address. In diagrams of a field using little-endian byte ordering, the MSB and LSB are labeled.

EDITOR'S NOTE: All new sections.

4.2.23 End-to-end data protection

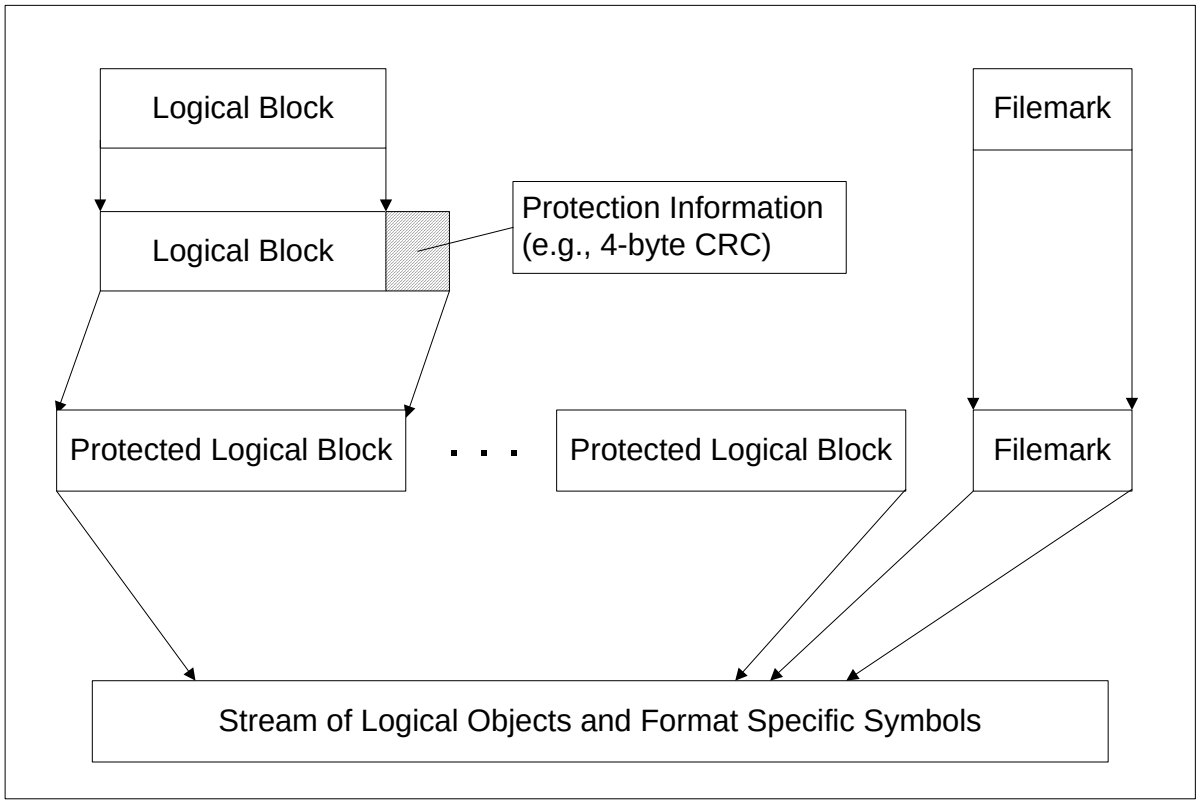
4.2.23.1 End-to-end data protection overview

A device compliant with this standard may contain hardware or software that is capable of checking and/or generating protection information that is transferred with data between the device server and an application client. This protection information transferred with logical blocks is saved to media with each logical block and read from media with each logical block. This protection information is validated at the destination prior to completing the task thereby ensuring that the data has not been corrupted. This provides a level of detection above what an application client can achieve by inserting its own data protection since the device server validates the protection information. The configuration of this capability is performed using the Control Data Protection mode page (see 8.3.9). A device that supports using this protection information shall support the Control Data Protection mode page.

4.2.23.2 Protection information on a volume

A recorded volume contains logical objects (see 4.2.7.1) and format specific symbols. Logical objects are application client accessible but format specific symbols are not. Format specific symbols are used by the device server to provide methods for recording logical objects on the medium in a manner that allows them to be successfully read at a later date. Often format specific symbols contain information used to protect logical objects. When a device server supports a non-zero value in the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page it shall include the protection information field as one of the format specific symbols and save it to the medium with every logical block. A representation of this is shown in Figure 1.

FIGURE 1. Protection information shown in relation to logical objects and format specific symbols



QUESTION: Should there be added the option to allow a Transform of the protection information into some format specific protection information that is instead saved to medium? This would allow us to require the same RS CRC protection information for all devices (and only require HBAs to support the one algorithm). This would also allow a path from today to the future for those formats that do not include the CRC algortihm(s) supported.

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page is set to zero, a device server that supports a non-zero value in the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page shall:

a) when recording data to the medium, generate the protection information and add it to the logical block before recording the logical block to the medium; and

b) when reading data from the medium, read the protection information from the medium, validate it, and remove it from the logical block before transferring the logical block to the application client. If validation of the protection information fails, report the error in the manner specified by the RDPR field of the Control Data Protection mode page.

4.2.23.3 Logical blocks and protection information

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page is set to zero for a specific I_T nexus then a logical block transferred between the application cli-

ent and the device server through that I_T nexus is defined by Table x1 and the READ BLOCK LIMITS command shall return:

a) a value in the MINIMUM BLOCK LENGTH LIMIT field greater than or equal to one, and

b) a value in the MAXIMUM BLOCK LENGTH LIMIT field less than or equal to 2^{24} bytes.

TABLE x1. Logical block with no protection information

Byte	Bit							
	7	6	5	4	3	2	1	0
0	Data							
n								

n = one less than the TRANSFER LENGTH field specified in CDB for variable length transfers; one less than the BLOCK LENGTH field specified in the mode parameter header (see SPC-4) for fixed block transfers.

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to a non-zero value and the BPP field is set to 00b for a specific I_T nexus then a logical block transferred between the application client and the device server through that I_T nexus is defined by Table x2 and the READ BLOCK LIMITS command shall return to that I_T nexus:

a) a value in the MINIMUM BLOCK LENGTH LIMIT field greater than or equal to one, and

b) a value in the MAXIMUM BLOCK LENGTH LIMIT field less than or equal to 2^{24} - LOGICAL BLOCK PROTECTION INFORMATION LENGTH bytes.

TABLE x2. Logical block with appended protection information

Byte	Bit							
	7	6	5	4	3	2	1	0
0	Data							
(n-x) - 1								
n-x	Protection Information							
n								

n = one less than the TRANSFER LENGTH field specified in CDB for variable length transfers; one less than the BLOCK LENGTH field specified in the mode parameter header (see SPC-4) for fixed block transfers.
x = one less than the LOGICAL BLOCK PROTECTION INFORMATION LENGTH specified in the Control Data Protection mode page

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to a non-zero value and the BPP field is set to 01b for a specific I_T nexus then a logical block

transferred between the application client and the device server thorugh that I_T nexus is defined by Table x3 and the READ BLOCK LIMITS command shall return to that I_T nexus:

a) a value in the minimum block length limit field greater than or equal to one, and

b) a value in the maximum block length limit field less than or equal to 2^{24} - LOGICAL BLOCK PROTECTION INFORMATION LENGTH bytes..

TABLE x3. Logical block with prepended protection information

Byte	Bit							
	7	6	5	4	3	2	1	0
0	Protection Information							
x								
x+1	Data							
n								

n = one less than the TRANSFER LENGTH field specified in CDB for variable length transfers; one less than the BLOCK LENGTH field specified in the mode parameter header (see SPC-4) for fixed block transfers.
x = one less than the LOGICAL BLOCK PROTECTION INFORMATION LENGTH specified in the Control Data Protection mode page

4.2.23.4 Protection information for Recover Buffered Data

In response to a RECOVER BUFFERED DATA command the device server transfers unwritten data from the logical units object buffer to the application client. This data is shown in Table x4.

TABLE x4. Data transferred from the logical units object buffer in response to RECOVER BUFFERED DATA command

Byte	Bit							
	7	6	5	4	3	2	1	0
0	Recover Buffered Data descriptor (first)							
n								
	.							
	.							
	.							
sn-n	Recover Buffered Data descriptor (last)							
sn								

n = one less than the block length
s = number of blocks in the logical units object buffer prior to any filemark

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to zero for a specific I_T nexus then the Recover Buffered Data descriptor used on that I_T nexus is defined by Table x1.

When the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to a non-zero value and the BPP field is set to 00b for a specific I_T nexus then the Recover Buffered Data descriptor used on that I_T nexua is defined by Table x2

When the logical block protection method field of the Control Data Protection mode page set to a non-zero value and the bpp field is set to 01b for a specific I_T nexus then the Recover Buffered Data descriptor used on that I_T nexus is defined by Table x3

4.2.23.5 Protecting logical blocks transferred during writes

When the WDP bit of the Control Data Protection mode page (see 8.3.9) is set to one for a specific I_T nexus, each logical block transferred from the application client through that I_T nexus contains protection information. The commands for which this applies are:

a) WRITE(6);

b) WRITE(16);

c) VERIFY(6) with the BYTCMP field set to one; and

d) VERIFY(16) with the BYTCMP field set to one.

For the WRITE(6) and WRITE(16) commands, the device server shall validate the protection information before the logical block is written to medium. If the validation of the protection information fails, the device server shall respond as defined by the WDPR field in the Control Data Protection mode page. When the WDPR field of the Control Data Protection mode page is set to 00b, the protection information shall be validated before sending status to the command that caused the transfer of the logical block.

For the VERIFY(6) and VERIFY(16) commands with the BYTCMP field set to one the protection information is validated prior to the byte-by-byte compare of the data on the medium and the data transferred from the application client. If the validation of the protection information fails, the device server shall respond as defined by the WDPR field in the Control Data Protection mode page. If the validation of the protection information does not fail, the byte-by-byte compare of the data on the medium and the data transferred from the application client takes place. This byte-by-byte compare also includes the protection information on the medium and the protection information transferred from the application client.

A device server that supports the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to a non-zero value shall support the WDP bit of the Control Data Protection mode page set to one.

An application client that has set the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page to a non-zero value and the WDP bit of the Control Data Protection mode page to one shall add the protection information on each logical block before transferring that data and shall increase the TRANSFER LENGTH field by four. The application client should add the protection information to the data at the earliest point possible. If the data has had the protection information added to the logical block at some point in the application client prior to the hardware that transfers the logical block, the protection information should be validated when it is

transferred. If the validation fails, the application client should abort the command and report a status to the user that validation failed.

QUESTION: What is appropriate to do here regarding how the protection information is generated in the application client and it is used? Can I specify this and is this the correct way to specify it?

NOTE yy When the bpp field of the Control Data Protection mode page is set to 00b, the device server treats the last LOGICAL BLOCK PROTECTION INFORMATION LENGTH number of bytes as the protection information. When the bpp field of the Control Data Protection mode page is set to 01b, the device server treats the first LOGICAL BLOCK PROTECTION INFORMATION LENGTH number of bytes as the protection information. If the protection information is not added to the logical block, then the validation fails when the bytes used do not validate (e.g., the last 4-bytes of data are treated as the CRC and the last 4-bytes of the data do not calculate as the CRC of the previous data)

4.2.23.6 Protecting logical blocks transferred during reads

When the RDP bit of the Control Data Protection mode page (see 8.3.9) is set to one for a specific I_T nexus, the protection information shall be read from the medium and transferred with the logical block to the application client on that I_T nexus. The commands for which this applies are:

a) READ(6);
b) READ(16);
c) READ REVERSE(6); and
d) READ REVERSE(16).

EDITOR'S NOTE: Bob Nixon asked if we need to cover VERIFY commands with the BYTCMP field set to zero. My thinking is that there is no data begin transferred from the drive or to the drive so it is not needed. However, if the device supports end-to-end then there will be the CRC that could be validated - but the transfer length specified would need to be 4 bytes longer. The verify command may already cover this without additional text. It says:
"A byte compare (BYTCMP) bit of zero specifies the verification shall be simply a medium verification (e.g., CRC, ECC). No data shall be transferred from the application client to the device server."

The protection information shall be validated by the device server before sending status to the command that caused the transfer of the logical block. If the validation of the protection information fails, the device server shall report a Check Condition using a Sense Code of Current Sense, the sense key set to HARDWARE ERROR and the additional sense code set to LOGICAL BLOCK PROTECTION ERROR ON READ. A device that supports the LOGICAL BLOCK PRO-

TECTION METHOD field of the Control Data Protection mode page set to a non-zero value shall support the RDP bit of the Control Data Protection mode page set to one.

EDITOR'S NOTE: LOGICAL BLOCK PROTECTION ERROR ON READ is a new additional sense code

An application client that has set the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page to a non-zero value and the RDP bit of the Control Data Protection mode page to one should validate the protection information on each logical block at the latest point possible before using the data.

4.2.23.7 Protecting data transferred from the object buffer in response to a RECOVER BUFFERED DATA command

When the RBDP bit of the Control Data Protection mode page (see 8.3.9) is set to one for a specific I_T nexus, each logical block transferred between the device server and the application client on that I_T nexus during a RECOVER BUFFERED DATA command (see 8.3.9) shall include the protection information. The device server shall:

a) read the protection information from the object buffer if it exists; or

b) generate the protection information if it does not exist.

The protection information for each block shall be validated before sending status to the command. If the validation of the protection information fails for any logical block, the device server shall terminate the command without transferring any additional logical blocks that may exist in the object buffer and report a Check Condition using a Sense Code of Current Sense, the sense key set to HARDWARE ERROR and the additional sense code set to LOGICAL BLOCK PROTECTION ERROR ON RECOVER BUFFERED DATA. A device shall support the RBDP bit of the Control Data Protection mode page set to one if it supports:

a) the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page set to a non-zero value; and

b) the RECOVER BUFFERED DATA command.

EDITOR'S NOTE: LOGICAL BLOCK PROTECTION ERROR ON RECOVER BUFFERED DATA is a new additional sense code

An application client that has set the LOGICAL BLOCK PROTECTION METHOD field of the Control Data Protection mode page to a non-zero value and the RBDP bit of the Control Data Protection mode page to one should validate the protection information on each logical block at the latest point possible before using the data.

8.3.9 Control Data Protection mode page

The Control Data Protection mode page provides controls that allow selective use of end-to-end data protection. The mode page policy of this page shall be Per I_T nexus.

TABLE x5. Control Data Protection mode page format

Byte	Bit							
	7	6	5	4	3	2	1	0
0	PS	SPF(1b)	PAGE CODE (0Ah)					
1	SUBPAGE CODE (F0h)							
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	LOGICAL BLOCK PROTECTION METHOD							
5	BPP		LOGICAL BLOCK PROTECTION INFORMATION LENGTH					
6	WDP	RDP	RBDP	Reserved				PIE
7	WDPR		Reserved					
8	Reserved							
n								

EDITOR'S NOTE: I wish to leave reserved bytes - at least show that there may be some additional bytes in the future - in case anybody ever decides to do anything on a basis larger than logical block (e.g., file) for end-to-end protection. This seems like a potential extension - although a difficult one.

QUESTION: Should we add the option to pad the protection information. In the only defined protection information a CRC is used. Currently the CRC is added on a byte boundary (i.e., immediately with no padding). If we add a pad to align to a 4-byte boundary, how do we indicate in the data stream the size of the pad? This would add great complexity or require the CRC to be zero neutral and for this reason, I do not want to do it.

The LOGICAL BLOCK PROTECTION METHOD is defined in Table x6.

TABLE X6. END-TO-END DATA PROTECTION METHOD values

Value	Description
00h	Do not use logical block protection
01h	Use the Reed-Solomon CRC (see ECMA-319) as the logical block protection information.
02h	Use the 4-byte Fibre Channel CRC (see FC-FS-2) as the logical block protection information. EDITOR'S NOTE: If nobody comes forward with a desire to keep this method it will be removed.
03h - FFh	Reserved

The block protection placement (BPP) field is defined in Table x7.

TABLE x7. Block protection placement values

Value	Definition
00b	The logical block protection information is appended to the data
01b	The logical block protection information is prepended to the data EDITOR'S NOTE: I have been asked if this option to prepend the protection can be removed. If nobody comes forward with a desire to keep this, this option will be removed and this field will no longer be needed and will also be removed.
10b - 11b	Reserved

The LOGICAL BLOCK PROTECTION INFORMATION LENGTH specifies the length in bytes of the logical block protection information. If the LOGICAL BLOCK PROTECTION METHOD is set to 01h or 02h, then the LOGICAL BLOCK PROTECTION INFORMATION LENGTH shall be set to 04h.

The write data protected (WDP) bit set to one indicates that the protection information is included with logical blocks transferred during commands specified in 4.2.23.5. The WDP bit set to zero indicates that the protection information is not included with data transferred when writing.

The read data protected (RDP) bit set to one indicates that the protection information is included with logical blocks transferred during the commands specified in 4.2.23.6. The RDP bit set to zero indicates that the protection information is not included with data transferred when reading.

The recover buffered data protected (RBDP) bit set to one indicates that the protection information is transferred with the data transferred by the RECOVER BUFFERED DATA command. The

RBDP bit set to zero indicates that the protection information is not transferred with the data transferred by the RECOVER BUFFERED DATA command.

The protection information endian (PIE) bit set to one indicates that the protection information is big-endian byte ordering. The PIE bit set to zero indicates that the protection information is little-endian byte ordering. If the LOGICAL BLOCK PROTECTION METHOD is set to 01h then the PIE bit shall be set to one. If the LOGICAL BLOCK PROTECTION METHOD is set to 02h then the PIE bit shall be set to zero.

EDITOR'S NOTE: I know that the endian-ness of Reed-Solomon CRC is opposite that of Fibre Channel CRC, but I am not positive I have the correct endians specified.

EDITOR'S NOTE: (Note from Bob Nixon) It may not be necessary to get into endian-ness at all...We honestly do not care how the data is stored (or if we do, it's a media-format question, not T10). We only care how it is represented in the data stream to and from the device.

So far, each protection method specifies a single (although different) ordering. Presuming we could make that a restriction, the diagram of each DIF format as a data structure, showing MSB and LSB (they would be opposite for RS CRC and FC CRC), would be sufficient. You never need to bring up endian-ness, and you can kill the PIE bit.

The write data protection reporting (WDPR) field is defined in Table x8

TABLE x8. WDPR definition

Value	Device server behavior when the validation of the data fails
00b	Report a Check Condition using a Sense Code of Current Sense, the sense key set to HARDWARE ERROR and the additional sense code set to LOGICAL BLOCK PROTECTION ERROR ON WRITE.
01b	Establish a Check Condition for return on the next eligible command with the Sense Code set to Deferred Sense, the sense key set to HARDWARE ERROR and the additional sense code set to LOGICAL BLOCK PROTECTION ERROR ON WRITE.
10b - 11b	Reserved

EDITOR'S NOTE: LOGICAL BLOCK PROTECTION ERROR ON WRITE is a new additional sense code