Date: July 14, 2004

To: T10 Committee (SCSI)

From: George Penokie (IBM/Tivoli)

Subject: SBC-2: SPC-3: Protection Information Fixes

1 Overview

Several minor errors have been found in the latest versions of SBC-2 and SPC-3 in the protection information description which need to be corrected for proper implementation of end-to-end protection.

2 SBC-2 issues

2.1 Issue #1

In the RDPROTECT field of the READ (10), READ (12), and READ (16) commands there is no definition as to what to do for a legacy read when the REF_CHK bit is set to one. It should be the same as what is defined in the READ (6) command. The below change to table 1 fixes this issue.

2.2 Issue #2

In the RDPROTECT field of the READ (10), READ (12), and READ (16) commands there is a reference to footnote (h) in the value = 010b / APP_CHK =1 cell and the value = 010b / REF_CHK =1 cell. Those two footnote references should be deleted as that footnote now applies (as is correctly labeled in the current table 1) to the entire Field in protection information column. The below change to table 1 fixes this issue.

.

Table 1 — RDPROTECT field (part 1 of 3)

Value	Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information [h]	Extended INQUIRY Data VPD page bit value [g]	If check fails [d] [f], additional sense code
000b [i]	Yes	No	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	No check performed
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	No check performed
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 [i]	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	No check performed
	No		No protection information available to check		

Table 1 — RDPROTECT field (part 2 of 3)

Value	**Logical unit formatted with protection information**	**Shall device server transmit protection information?**	**Field in protection information** [h]	**Extended INQUIRY Data VPD page bit value** [g]	**If check fails** [d] [f]**, additional sense code**
001b [b] [i]	Yes	Yes [e]	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	No check performed
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	No check performed
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	No check performed
	No [a]	No protection information available to transmit to the application client or for checking			
010b [b] [i]	Yes	Yes [e]	LOGICAL BLOCK GUARD	Shall not	No check performed
			LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c] ~~h~~	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
				APP_CHK = 0	No check performed
			LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 ~~h~~	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
				REF_CHK = 0	No check performed
	No [a]	No protection information available to transmit to the application client or for checking			
011b [b] [i]	Yes	Yes [e]	LOGICAL BLOCK GUARD	Shall not	No check performed
			LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
			LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No [a]	No protection information available to transmit to the application client or for checking			

Table 1 — RDPROTECT field (part 3 of 3)

Value	Logical unit formatted with protection information	Shall device server transmit protection information?	Field in protection information [h]	Extended INQUIRY Data VPD page bit value [g]	If check fails [d] [f], additional sense code
100b [b] [i]	Yes	Yes [e]	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
				GRD_CHK = 0	No check performed
			LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
			LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No [a]	No protection information available to transmit to the application client or for checking			
101b - 111b	Reserved				

[a] A read operation to a logical unit that supports protection information (see 4.15) and has not been formatted with protection information shall fail with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB.
[b] If the logical unit does not support protection information the requested command should fail with CHECK CONDITION status with a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.
[c] The device server checks the logical block application tag only if it has knowledge of the contents of the LOGICAL BLOCK APPLICATION TAG field. This knowledge may be obtained by use of the READ (32) command (see 5.12) or by a method not defined by this standard.
[d] If an error is reported the sense key shall be set to ABORTED COMMAND.
[e] Transmit protection information to the application client.
[f] If multiple errors occur, the selection of which error to report is not defined by this standard.
[g] See the Extended INQUIRY Data VPD page (see SPC-3) for a description of the GRD_CHK, APP_CHK, and REF_CHK bits.
[h] If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field set to FFFFh, the checking of all protection information in the associated logical block shall be disabled.
[i] If the RTO_EN bit is set to zero in the long read capacity data (see 5.14), the device server may process the command. If the RTO_EN bit is set to one, READ (10), READ (12),and READ (16) commands with the RDPROTECT field set to 000b may be processed by the device server. If the RTO_EN bit is set to one, the device server shall terminate READ (10), READ (12), and READ (16) commands with the RDPROTECT field not set to 000b with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and an additional sense code of INVALID COMMAND OPERATION CODE.
[j] If the RTO_EN bit is set to zero in the long read capacity data (see 5.14), the device server checks the logical block reference tag by comparing it to the lower 4 bytes of the LBA associated with the logical block. If the RTO_EN bit is set to one, the device server checks the logical block reference tag only if it has knowledge of the contents of the LOGICAL BLOCK REFERENCE TAG field. The method for acquiring this knowledge is not defined by this standard.

2.3 Issue #3

In the READ (32) command description the paragraphs below there is no indication as to what the value of the APP_CHK bit should be. The corrections are indicated in the paragraphs.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 33x) ~~by as defined in the RDPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3),~~ the EXPECTED LOGICAL BLOCK APPLICATION TAG field contains a value that is expected in the LOGICAL BLOCK APPLICATION TAG with the LOGICAL BLOCK APPLICATION TAG MASK applied in the protection information of logical blocks for this command.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 33x) ~~by as defined in the RDPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3),~~ the LOGICAL BLOCK APPLICATION MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG in the protection information for each logical block of the range of logical blocks for this command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit in the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the LOGICAL BLOCK APPLICATION TAG.

2.4 Issue #4

In the VRPROTECT field of the VERIFY (10), VERIFY (12), and VERIFY (16) commands there is a reference to footnote (g) in the value = 000b / GRD_CHK =1 cell and the value = 000b / REF_CHK =1 cell. Those two footnote references should be deleted and the (g) footnote added to the Field in protection information column heading and this footnote applies to the entire Field in protection information column. The below change to table 2 fixes this issue.

Table 2 — VRPROTECT field with BYTCHK set to one - checking protection information read from the medium

Value	Logical unit formatted with protection information	Field in protection information [g]	Extended INQUIRY Data VPD page bit value [f]	If check fails [d] [e], additional sense code
000b	Yes	LOGICAL BLOCK GUARD	GRD_CHK = 1~~[g]~~	LOGICAL BLOCK GUARD CHECK FAILED
			GRD_CHK = 0	No check performed
		LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c] [g]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1~~[g]~~	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
	No	No protection information on the medium available to check		
001b 010b 011b 100b [b]	Yes	LOGICAL BLOCK GUARD	Shall not	No check performed
		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition [a]		
101b - 111b	Reserved			

[a] A verify operation to a logical unit that supports protection information (see 4.15) and has not been formatted with protection information shall fail with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB.
[b] If the logical unit does not support protection information the requested command should fail with CHECK CONDITION status with a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB.
[c] The device server checks the logical block application tag only if it has knowledge of the contents of the LOGICAL BLOCK APPLICATION TAG field. This knowledge may be obtained by use of the VERIFY (32) command (see 5.26) or by a method not defined by this standard.
[d] If an error is reported, the sense key shall be set to ABORTED COMMAND.
[e] If multiple errors occur, the selection of which error to report is not defined by this standard.
[f] See the Extended INQUIRY Data VPD page (see SPC-3) for a description of the GRD_CHK, APP_CHK, and REF_CHK bits.
[g] If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field set to FFFFh, the checking of all protection information shall be disabled for the associated logical block.

2.5 Issue #5

In the VERIFY (32) command description the paragraphs below there is no reason for the APP_CHK bit to be checked beyond what is already in the VRPROTECT field description. The corrections are indicated in the following paragraphs.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 58x, table 59x, table 60x, and table 61x) ~~by as defined in the VRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the EXPECTED LOGICAL BLOCK APPLICATION TAG field contains a value that is expected in the LOGICAL BLOCK APPLICATION TAG with the LOGICAL BLOCK APPLICATION TAG MASK applied in the protection information of logical blocks for this command.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 58x, table 59x, table 60x, and table 61x) ~~by as defined in the VRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the LOGICAL BLOCK APPLICATION MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG in the protection information for each logical block of the range of logical blocks for this command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit in the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the LOGICAL BLOCK APPLICATION TAG.

2.6 Issue #6

In the description of the WRPROTECT field in Table 68 — WRPROTECT field footnote (g) does not make it clear that FFFF FFFFh shall be written to on all data blocks not just the first one. The correction are indicated in the following paragraph:

g If the RTO_EN bit is set to zero in the long read capacity data (see 5.14), the device server shall write the least significant four bytes of ~~the~~ each LBA into the LOGICAL BLOCK REFERENCE TAG field of each of the written logical blocks. If the RTO_EN bit is set to one, the device server shall write a value of FFFFFFFFh into the LOGICAL BLOCK REFERENCE TAG field of each of the written logical blocks.

2.7 Issue #7

In the WRITE (32) command description the paragraphs below there is no reason for the APP_CHK bit to be checked beyond what is already in the WRPROTECT field description. The corrections are indicated in the following paragraphs.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3),~~ the EXPECTED LOGICAL BLOCK APPLICATION TAG field contains a value that is expected in the LOGICAL BLOCK APPLICATION TAG with the LOGICAL BLOCK APPLICATION TAG MASK applied in the protection information of logical blocks for this command.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the LOGICAL BLOCK APPLICATION MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG in the protection information for each logical block of the range of logical blocks for this command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit in the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the LOGICAL BLOCK APPLICATION TAG.

2.8 Issue #8

In the WRITE AND VERIFY (32) command description the paragraphs below there is no reason for the APP_CHK bit to be checked beyond what is already in the WRPROTECT field description. The corrections are indicated in the following paragraphs.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the EXPECTED LOGICAL BLOCK APPLICATION TAG field contains a value that is expected in the LOGICAL BLOCK APPLICATION TAG with the LOGICAL BLOCK APPLICATION TAG MASK applied in the protection information of logical blocks for this command.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the LOGICAL BLOCK APPLICATION MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG in the protection information for each logical block of the range of logical blocks for this command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit in the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the LOGICAL BLOCK APPLICATION TAG.

2.9 Issue #9

In the WRITE SAME (32) command description the paragraphs below there is no reason for the APP_CHK bit to be checked beyond what is already in the WRPROTECT field description. The corrections are indicated in the following paragraphs.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the EXPECTED LOGICAL BLOCK APPLICATION TAG field contains a value that is expected in the LOGICAL BLOCK APPLICATION TAG with the LOGICAL BLOCK APPLICATION TAG MASK applied in the protection information of logical blocks for this command.

When checking of the LOGICAL BLOCK APPLICATION TAG is enabled (see table 68x) ~~by as defined in the WRPROTECT field in the CDB and the APP_CHK bit in the Extended INQUIRY Data VPD page (see SPC-3)~~, the LOGICAL BLOCK APPLICATION MASK field contains a value that is a bit mask for enabling the checking of the LOGICAL BLOCK APPLICATION TAG in the protection information for each logical block of the range of logical blocks for this command. A LOGICAL BLOCK APPLICATION TAG MASK bit set to one enables the checking of the corresponding bit in the EXPECTED LOGICAL BLOCK APPLICATION TAG field with the LOGICAL BLOCK APPLICATION TAG.

2.10 Issue #10

The current definition of write same requires the recalculation of CRC on each block when the LBDATA bit or the PBDATA bit is set to one. This has the same problem as format had in that it requires the drive to generate the CRC on the back side of the data stream. In format we solved the problem by writing all FFh's in the protection fields. But in Write Same we solved the problem with a check condition with no indication (e.g., a bit in inquiry that states the drive does not support write same with LBDATA bit or the PBDATA bit is set to one with protection disabled). Giving a check condition will cause problems in the real world. The proposed fix is shown in table 3.

.

Table 3 — LBDATA bit and PBDATA bit

LBDATA	PBDATA	Description
0	0	The device server shall write the single block of data received from the application client data-out buffer to each logical block without modification.
0	1 [a]	The device server shall replace the first eight bytes of the block received from the application client data-out buffer to each physical sector with the physical address of the sector being written using the physical sector format (see 5.3.3.5).
1 [a]	0	The device server shall replace the first four bytes of the block received from the application client data-out buffer with the least significant four bytes of the LBA of the block being written. The most significant byte of the four bytes shall be written first.
1	1	The device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

[a] If the medium is formatted with protection information then ~~the logical unit~~ protection information shall be written to a default value of FFFFFFFF FFFFFFFFh in each of the written logical blocks. ~~is required to recalculate a CRC for each logical block written to the medium and place the new CRC value into the corresponding LOGICAL BLOCK GUARD field. If the logical unit does not support recalculation of the CRC, the device server shall terminate the command with a CHECK CONDITION status with a sense key set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB.~~

2.11 issue 11#

In the table (LBDATA bit and PBDATA bit table in the WRITE SAME description) that describes the LBDATA bit set to zero and PBDATA bit set to zero case it implies that all the information received (including the protection information) is written into each logical block unmodified. However, the protection information is modified as described directly above the table. The follow wording change in the table should clear this up:

~~If the medium is formatted with protection information, the value in the LOGICAL BLOCK REFERENCE TAG field received from the application client shall be placed into the LOGICAL BLOCK REFERENCE TAG field of the first logical block written to the medium. Into each of the following logical blocks the logical block reference tag received in the data transferred from the application client, incremented by one, shall be placed into the LOGICAL BLOCK REFERENCE TAG field of that logical block (i.e., each logical block written to the medium has a logical block reference tag value of one greater than the previous logical block).~~

~~If the APP_TAG_OWN bit is set to one in the Control mode page (see SPC-3), the logical block application tag received in the single block of data shall be placed in the LOGICAL BLOCK APPLICATION TAG field of each logical block. If the APP_TAG_OWN bit is set to zero, the logical block application tag received in the single block of data may be placed in the LOGICAL BLOCK APPLICATION TAG field of each logical block.~~

.

Table 4 — LBDATA bit and PBDATA bit

LBDATA	PBDATA	**Description**
0	0	The device server shall write the single block of user data, received from the application client data-out buffer to each logical block without modification. If the medium is formatted with protection information, the value in the LOGICAL BLOCK REFERENCE TAG field received from the application client shall be placed into the LOGICAL BLOCK REFERENCE TAG field of the first logical block written to the medium. Into each of the following logical blocks the logical block reference tag received in the data transferred from the application client, incremented by one, shall be placed into the LOGICAL BLOCK REFERENCE TAG field of that logical block (i.e., each logical block written to the medium has a logical block reference tag value of one greater than the previous logical block). If the APP_TAG_OWN bit is set to one in the Control mode page (see SPC-3), the logical block application tag received in the single block of data shall be placed in the LOGICAL BLOCK APPLICATION TAG field of each logical block. If the APP_TAG_OWN bit is set to zero, the logical block application tag received in the single block of data may be placed in the LOGICAL BLOCK APPLICATION TAG field of each logical block.
0	1 [a]	The device server shall replace the first eight bytes of the block received from the application client data-out buffer to each physical sector with the physical address of the sector being written using the physical sector format (see 5.3.3.5).
1 [a]	0	The device server shall replace the first four bytes of the block received from the application client data-out buffer with the least significant four bytes of the LBA of the block being written. The most significant byte of the four bytes shall be written first.
1	1	The device server shall terminate the command with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST and the additional sense code set to INVALID FIELD IN CDB.

[a] If the medium is formatted with protection information then the logical unit is required to recalculate a CRC for each logical block written to the medium and place the new CRC value into the corresponding logical block guard field. If the logical unit does not support recalculation of the CRC, the device server shall terminate the command with a CHECK CONDITION status with a sense key set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB.

2.12 issue 12#

There is an issue regarding VERIFY with VRPROTECT = 000 and BYTECHK = 0 for the REF_CHK case? If RTO_EN = 0, this value will be the LBA. But if RTO_EN = 1, it might be FFFFFFFFh or it might be the LBA or it might be some other value (unlocked case), depending on the source and type of the WRITE command. To fix this it is recommended the same wording that is in READ(6) table 31 footnote f be added to the VERIFY table. As shown in table 5.

Table 5 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the medium (part 1 of 3)

Value	Logical unit formatted with protection information	Field in protection information [g]	Extended INQUIRY Data VPD page bit value [f]	If check fails [d] [e], additional sense code
000b	Yes	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
			GRD_CHK = 0	No check performed
		LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1 [h]	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
	No	No protection information on the medium to check. Only user data is checked.		
001b [b]	Yes	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
			GRD_CHK = 0	No check performed
		LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
	No	Error condition [a]		
010b [b]	Yes	LOGICAL BLOCK GUARD	Shall not	No check performed
		LOGICAL BLOCK APPLICATION TAG	APP_CHK = 1 [c]	LOGICAL BLOCK APPLICATION TAG CHECK FAILED
			APP_CHK = 0	No check performed
		LOGICAL BLOCK REFERENCE TAG	REF_CHK = 1	LOGICAL BLOCK REFERENCE TAG CHECK FAILED
			REF_CHK = 0	No check performed
	No	Error condition [a]		

Table 5 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the medium (part 2 of 3)

Value	Logical unit formatted with protection information	Field in protection information [g]	Extended INQUIRY Data VPD page bit value [f]	If check fails [d] [e], additional sense code
011b [b]	Yes	LOGICAL BLOCK GUARD	Shall not	No check performed
		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition [a]		
100b [b]	Yes	LOGICAL BLOCK GUARD	GRD_CHK = 1	LOGICAL BLOCK GUARD CHECK FAILED
			GRD_CHK = 0	No check performed
		LOGICAL BLOCK APPLICATION TAG	Shall not	No check performed
		LOGICAL BLOCK REFERENCE TAG	Shall not	No check performed
	No	Error condition [a]		
101b - 111b	Reserved			

Table 5 — VRPROTECT field with BYTCHK set to zero - checking protection information read from the medium (part 3 of 3)

Value	Logical unit formatted with protection information	Field in protection information [g]	Extended INQUIRY Data VPD page bit value [f]	If check fails [d] [e], additional sense code
[a] A verify operation to a logical unit that supports protection information (see 4.15) and has not been formatted with protection information shall fail with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST with the additional sense code set to INVALID FIELD IN CDB. [b] If the logical unit does not support protection information the requested command should fail with CHECK CONDITION status with a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB. [c] The device server checks the logical block application tag only if it has knowledge of the contents of the LOGICAL BLOCK APPLICATION TAG field. This knowledge may be obtained by use of the VERIFY (32) command (see 5.26) or by a method not defined by this standard. [d] If an error is reported, the sense key shall be set to ABORTED COMMAND. [e] If multiple errors occur, the selection of which error to report is not defined by this standard. [f] See the Extended INQUIRY Data VPD page (see SPC-3) for a description of the GRD_CHK, APP_CHK, and REF_CHK bits. [g] If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field set to FFFFh, the checking of all protection information shall be disabled for the associated logical block. [h] If the RTO_EN bit is set to zero in the long read capacity data (see 5.14), the device server checks the logical block reference tag by comparing it to the lower 4 bytes of the LBA associated with the logical block. If the RTO_EN bit is set to one, the device server checks the logical block reference tag only if it has knowledge of the contents of the LOGICAL BLOCK REFERENCE TAG field. The method for acquiring this knowledge is not defined by this standard.				

3 SPC-3 issues

3.1 Issue #1

In the description of the guard check, application tag check, and the reference tag check bits there are words describing what to do if the logical block application tag field contains FFFFh. This is defined in SBC-2 and should not be restated here as it has nothing to do with the description of the bits themselves. The corrections are indicated in the following paragraphs.

A guard check (GRD_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK GUARD field in the protection information (see SBC-2) before transmitting it to an application client. A GRD_CHK bit set to one indicates the device server checks the LOGICAL BLOCK GUARD field in the protection information before transmitting it to an application client. ~~If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK GUARD field in the protection information shall not be performed for the associated logical block.~~

An application tag check (APTG_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK APPLICATION TAG field in the protection information (see SBC-2) before transmitting it to an application client. An APTG_CHK bit set to one indicates the device server checks the LOGICAL BLOCK APPLICATION TAG field in the protection information before transmitting it to an application client. ~~If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK APPLICATION TAG field in the protection information shall not be performed for the associated logical block.~~

A reference tag check (RFTG_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK REFERENCE TAG field in the protection information (see SBC-2) before transmitting it to an application client. A RFTG_CHK bit set to one indicates the device server checks the LOGICAL BLOCK REFERENCE TAG field in the protection information before transmitting it to an application client. ~~If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK REFERENCE TAG field in the protection information shall not be performed for the associated logical block.~~

3.2 Issue #2 (This affects SPC-3 and SBC-2)

There is wording in the description of the guard check, application tag check, and the reference tag check bits that indicate they are checked before being transmitted to the application client, however, there are cases (e.g., the VERIFY command) where the checking is required but there is no transmission of the data. To fix this those words should be moved from the bit descriptions to the READ command were the rule applies. Those changes would be as follows:

SPC-3 change:

A guard check (GRD_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK GUARD field in the protection information (see SBC-2)~~ before transmitting it to an application client~~. A GRD_CHK bit set to one indicates the device server checks the LOGICAL BLOCK GUARD field in the protection information~~ before transmitting it to an application client~~. If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK GUARD field in the protection information shall not be performed for the associated logical block.

An application tag check (APTG_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK APPLICATION TAG field in the protection information (see SBC-2)~~ before transmitting it to an application client~~. An APTG_CHK bit set to one indicates the device server checks the LOGICAL BLOCK APPLICATION TAG field in the protection information~~ before transmitting it to an application client~~. If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK APPLICATION TAG field in the protection information shall not be performed for the associated logical block.

A reference tag check (RFTG_CHK) bit set to zero indicates the device server does not check the LOGICAL BLOCK REFERENCE TAG field in the protection information (see SBC-2)~~ before transmitting it to an application client~~. A RFTG_CHK bit set to one indicates the device server checks the LOGICAL BLOCK REFERENCE TAG field in the protection information~~ before transmitting it to an application client~~. If the application client or device server detects a LOGICAL BLOCK APPLICATION TAG field containing FFFFh, the checking of the LOGICAL BLOCK REFERENCE TAG field in the protection information shall not be performed for the associated logical block.

SBC-2 change:

In section 5.9 (READ (10) command) the sentence right above table 33 should be:

The device server shall check the protection information read from the medium before returning status for the command based on the RDPROTECT field as described in table 33.