March 27, 2006 06-051r6 The Requirement for More than One Decryption Key

To: T10 Technical Committee
From: Dwayne Edling (dwayne.edling@sun.com)
Date: 27 Mar 2006
Subject: T10/06-051r6 The Requirement for More than One Decryption Key

Revision History

Revision 0(7 Jan 2006): Presentation to explain the need for more than one decryption key.
Revision 1(19 Jan 2006): First proposal to implement more than one read Key.
Revision 2(23 Jan 2006): incorporated various edits.
Revision 3(28 Jan 2006): Remove encryption key to simplify proposal.
Revision 4(10 Feb 2006): Consistently used supplemental decryption key and increased field sizes to 32 bits for future growth.
Revision 5(11 Mar 2006): Incorporate HP comments.
Revision 6(27 Mar 2006): Incorporate comments from 23 Mar 2006, SSC-3 teleconference.

Related Documents

05-446rx- SSC-3: Add commands to control data encryption

Overview

Revision 1

This proposal incorporates changes that would be necessary to implement passing more than one decryption key in proposal 05-446r1.

- Added to Data encryption algorithm descriptor page:
  - MAXIMUM NUMBER OF ENCRYPTION KEYS
  - MAXIMUM NUMBER OF DECRYPTION KEYS
  - MAXIMUM NUMBER OF COMBINED KEYS (Used for both encryption and decryption)
  - MAXIMUM NUMBER OF KEYS FOR ALL MODES AND SCOPES
- Added to the Data encryption status page
  - NUMBER OF ENCRYPTION KEYS IN USE
  - NUMBER OF DECRYPTION KEYS IN USE
  - NUMBER OF COMBINED KEYS IN USE
  - NUMBER OF KEYS IN USE FOR ALL MODES AND SCOPES
- Added to the Set data encryption page.
  - KEY USE field; indicates whether this is a combined use key, encryption key or decryption key.
  - Keep Previous Decryption Key (KPDK) field; this field indicates whether the device server should keep the previously send decryption key.
  - Add scenarios how keys are sent and status that gets reported.

- I believe that I retained the functionality of the previous draft so that the model should not have to change.
- There were also some edits in the model sections to incorporate the concept of more than one key being used in the system.

Revision 2

Added edits from Paul Entzel.

- Changed wording in model section.
- Moved the following fields to the Data Encryption capabilities page
    - MAXIMUM NUMBER OF ENCRYPTION KEYS
    - MAXIMUM NUMBER OF DECRYPTION KEYS
    - MAXIMUM NUMBER OF COMBINED KEYS (Used for both encryption and decryption)
    - MAXIMUM NUMBER OF KEYS FOR ALL MODES AND SCOPES
- Changed wording in Set data encryption page to clarify various SCOPE and mode uses. Also changed various wording to add clarity. Removed last paragraph.

Revision 3

Change format to use only supplemental decryption key (SDK).

Removed

- MAXIMUM NUMBER OF ENCRYPTION KEYS
- MAXIMUM NUMBER OF COMBINED KEYS (Used for both encryption and decryption)
- MAXIMUM NUMBER OF KEYS FOR ALL MODES AND SCOPES

Removed

- NUMBER OF ENCRYPTION KEYS IN USE
- NUMBER OF COMBINED KEYS IN USE
- NUMBER OF KEYS IN USE FOR ALL MODES AND SCOPES

Removed KEY TYPE and KPDK

Revision 4

Replaced:

- MAXIMUM NUMBER OF DECRYPTION KEYS
- NUMBER OF DECRYPTION KEYS SAVED

With:

- MAXIMUM NUMBER OF SUPPLEMENTAL DECRYPTION KEYS
- NUMBER OF SUPPLEMENTAL DECRYPTION KEYS SAVED

In the Data Encryption Capabilities page and the Data Encryption Status page respectively.

Replaced reference to supported and values with references for storage space.

Revision 5

Incorporated comments from HP, which were not provided until the SSC meeting:

Remove

- MAXIMUM NUMBER OF SUPPLEMENTAL DECRYPTION KEYS
- NUMBER OF SUPPLEMENTAL DECRYPTION KEYS SAVED

Add model references for more than one key in 4.2.19.3 and 4.2.19.4

Add SDK_C field to algorithm descriptor page.

Define behavior with SDK set to zero.

Removed references to a key list

Revision 6

Incorporated comments from the SSC-3 teleconference meeting on 23 Mar 2006:

Add:

- Definition for SDK after furst use.
- A line in paragraph 4.2.19.3 stating that the method for determining which decryption key to use in vendor specific.
- A line in paragraph 4.2.19.7 with only the SDK addressed on it.
- A line in paragraph 4.2.19.8 stating that SDK's will not cause the Key Instance Counter to be incremented.
- Replace

Replace

- SDC_C with SDK_C on table E2.
- Replace "has not previously saved" with "does not currently have" on the first line of the second paragraph on page 14 of this proposal.

These last changes are highlighted in red.

Suggested Changes

4.2.19.3 Reading encrypted data on the medium

A volume may contain no encrypted blocks, all encrypted blocks, or a mixture of encrypted blocks and unencrypted blocks. The fact that blocks are encrypted shall not alter space or locate operations.

A device server that supports encryption should be capable of distinguishing encrypted blocks from unencrypted blocks. The device server reports its capability of distinguishing encrypted blocks from unencrypted blocks through the DED_C bit in the Data Encryption Algorithm descriptor (see 8.5.2.4). If the device server is capable of distinguishing encrypted blocks from unencrypted blocks, an attempt to read or verify an encrypted block when the decryption mode is set to DISABLED shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to UNABLE TO DECRYPT DATA. The device server shall establish the logical position at the BOP side of the encrypted block.

If the device server is capable of distinguishing encrypted blocks from unencrypted blocks and the decryption mode is set to DECRYPT or RAW, an attempt to read or verify an unencrypted block shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to UNENCRYPTED DATA ENCOUNTERED WHILE DECRYPTING. The device server shall establish the logical position at the BOP side of the unencrypted block.

NOTE E1: It is possible for a device server that is not capable of distinguishing encrypted blocks from unencrypted blocks to decrypt data that was not encrypted. Application clients are responsible for checking the integrity of the data in this environment.

A device server that supports encryption and has been configured to decrypt the data may be capable of determining that the encryption key is correct for an encrypted block. The correct key to use for this encrypted block may be either the encryption key or one of the supplemental decryption keys (SDK). The method for determine which key to use for decryption shall be vendor specific. If the device server is capable of determining that the ~~encryption~~ key is correct, an attempt to read or verify an encrypted block when the decryption mode is set to either DECRYPT or MIXED but the all of the ~~encryption~~ keys provided are ~~is~~ incorrect for the encrypted block shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to INCORRECT DATA ENCRYPTION KEY. The device server shall establish the logical position at the BOP side of the encrypted block.

A device server that supports encryption and has been configured to decrypt the data may be capable of validating the integrity of the data after decrypting it (i.e., that the decrypted data matches the data that was encrypted). If the device server is capable of validating the integrity of the data after decrypting it, an attempt to read or verify an encrypted block when the decryption mode is set to either DECRYPT or MIXED but the

data fails the integrity validation process shall cause the device server to terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to CRYPTOGRAPHIC INTEGRITY VALIDATION FAILED. The device server shall establish the logical position at the BOP side the encrypted block.

A device server that is capable of distinguishing encrypted blocks from unencrypted blocks and has been configured to decrypt the data should perform at least one of the following for each encrypted block that is decrypted:

a) determine if the encryption key is correct for the encrypted block; or
b) validate the integrity of the data after decrypting it.

A device server that is capable of both determining if the encryption key or one of the supplemental decryption keys is correct for the encrypted block and validating the integrity of the data after decrypting it shall:

1. determine if the encryption key is correct for the encrypted block; and
2. validate the integrity of the data.

4.2.19.4 Exhaustive-search attack prevention

To prevent an exhaustive-search attack from discovering the encryption key or supplemental decryption keys, the device server should provide a mechanism to prevent unlimited attempts at setting a data encryption key or supplemental decryption keys and then attempting to read the data. The use of such a mechanism may protect against an encryption algorithm being broken due to undiscovered mathematical weaknesses in the encryption algorithm.

If the device server has reached its limit on failed attempts to set the data encryption key or supplemental decryption keys and decrypt data, it shall disable decryption for all I_T nexuses. All subsequent SECURITY PROTOCOL OUT commands specifying the Tape Data Encryption security protocol and with the SP_SPECIFIC field set to Set Data Encryption page with the DECRYPT field or ENCRYPT field set to any value other than DISABLE shall be terminated with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to DATA DECRYPTION KEY FAIL LIMIT REACHED. This condition shall persist until the volume is de-mounted from the device or a hard reset condition occurs.

4.2.19.5 Managing keys within the device server

To increase the security of keys, the data encryption parameters are volatile in the device server and the data encryption keys are never reported to an application client. The device server also may have limited resources for storage of keys.

A device server that supports encryption shall support at least one of the key formats that are defined in this standard (see table Y5).

A vendor specific key reference is an identifier that is associated with a specific key. The method by which keys and their associated vendor specific key references are made available to the device server is outside the scope of this standard. A device server that

supports passing keys by vendor specific key reference shall include the code for vendor specific key reference format (see table Y5) in the SUPPORTED KEY FORMATS LIST field in the Supported Key Formats Page (see 8.5.2.5).

The device server shall release the resources used to save a set of data encryption parameters under the following conditions:

a) the CKOD bit is set to one in the saved data encryption parameters and the volume is de-mounted;

b) the CKORL bit is set to one and the key scope is set to LOCAL in the saved data encryption parameters and the I_T nexus that established the set of data encryption parameters loses its reservation;

c) the CKORL bit is set to one and the key scope is set to RESERVATION GROUP or ALL I_T NEXUS in the saved data encryption parameters and the device server transitions from reserved to not reserved;

d) a microcode update is performed on the device;

e) a power on condition occurs; or

f) other vendor specific events.

If a device server processes a Set Data Encryption page with the ENCRYPTION MODE field set to DISABLE and DECRYPTION MODE field set to DISABLE or RAW, the device server shall:

a) release any resources that it had allocated to store data encryption parameters for the I_T nexus associated with the SECURITY PROTOCOL OUT command and shall change the contents of all memory containing ~~a~~ key values associated with the data encryption parameters that are released; and

b) establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR for all other I_T nexus that are affected by the loss of the keys, (i.e., any I_T nexus that is using a data encryption scope of PUBLIC and sharing the keys.)

If a device server processes a Set Data Encryption page that includes a key and the SDK bit is set to zero, the device server shall:

a) release all resources that it had allocated to store a key values set by a previous SECURITY PROTOCOL OUT command from that I_T nexus and shall change the contents of all memory containing ~~a~~ key values associated with the data encryption parameters that are released; and

b) establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR for all other I_T nexus that is affected by the change of the keys (i.e., any I_T nexus that is using a data encryption scope of PUBLIC and sharing the keys).

A device server shall save at most one set of data encryption parameters with a key scope of ALL I_T NEXUS. If a device server processes a Set Data Encryption page with the SCOPE field set to ALL I_T NEXUS, the device server shall:

a) release any resources that it had allocated to store data encryption parameters set by a previous Set Data Encryption page with a scope value of ALL I_T NEXUS and shall change the contents of all memory containing ~~a~~ key values associated with the data encryption parameters that are released; and

b) establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR for all other I_T nexus that are affected by the change of the keys (i.e. any I_T nexus that is using a data encryption scope of PUBLIC and sharing the keys.)

A device server shall save at most one set of data encryption parameters with a key scope of RESERVATION GROUP. If a device server processes a Set Data Encryption page with the SCOPE field set to RESERVATION GROUP, the device server shall:

a) release any resources that it had allocated to store data encryption parameters set by a previous Set Data Encryption page with a scope value of RESERVATION GROUP and shall change the contents of all memory containing ~~a~~ key values; and

b) establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR for any other I_T nexus that is affected by the change of the keys, (i.e., any I_T nexus that is using a data encryption scope of PUBLIC and sharing the keys.)

If a vendor specific event occurs that changes or clears a set of data encryption parameters, the device server shall establish a unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY VENDOR SPECIFIC EVENT for any I_T nexus that is affected by the change of the keys.

4.2.19.6 Saved information per I_T Nexus

If the device server supports data encryption it shall save the following information on a per I_T nexus basis:

a) data encryption scope;
b) lock;
c) key instance counter value at lock; and
d) key instance counter value assigned to the last key established by a Set Data Encryption page for this I_T Nexus with a scope value of LOCAL and the SDK bit is set to zero;

The set of possible data encryption scope values for an I_T nexus is limited to:

a) PUBLIC;
b) LOCAL;
c) RESERVATION GROUP; or
d) ALL I_T NEXUS

If an I_T Nexus data encryption scope is set to PUBLIC it indicates the device server does not have a saved set of data encryption parameters that were established by that I_T

Nexus. Device servers that support encryption shall support an I_T Nexus data encryption scope of PUBLIC.

A device server shall set the data encryption scope for an I_T Nexus to LOCAL when it successfully completes the processing of a Set Data Encryption page with a scope of LOCAL from that I_T Nexus. The device server shall only use the data encryption parameters established by the Set Data Encryption page with a scope of LOCAL for processing commands from the I_T Nexus that established the parameters. A device server shall revert to using default data encryption parameters for an I_T Nexus that is configured with a data encryption scope of LOCAL if the resources used to save the data encryption parameters for the I_T Nexus are released.

A device server shall set the data encryption scope for an I_T Nexus to RESERVATION GROUP when it successfully completes the processing of a Set Data Encryption page with a scope value of RESERVATION GROUP from that I_T Nexus. At most, one I_T Nexus shall be assigned the data encryption scope of RESERVATION GROUP. If the device server releases resources used to store a set of data encryption parameters with a key scope of RESERVATION GROUP, it shall change the data encryption scope for the I_T Nexus that established that set of data encryption parameters to PUBLIC.

A device server shall set the data encryption scope for an I_T Nexus to ALL I_T NEXUS when it successfully completes the processing of Set Data Encryption page with a scope value of ALL I_T NEXUS from that I_T Nexus. At most, one I_T Nexus shall be assigned the data encryption scope of ALL I_T NEXUS. If the device server releases resources used to store a set of data encryption parameters with a key scope of ALL I_T NEXUS, it shall change the data encryption scope for the I_T Nexus that established that set of data encryption parameters to PUBLIC. Device servers that support encryption shall support an I_T Nexus data encryption scope of ALL I_T NEXUS.

By default, the device server shall set the saved I_T Nexus parameters data encryption scope value to PUBLIC and lock value to zero.

4.2.19.7 Data encryption parameters

A device server that supports data encryption shall have the ability to save the following information as a set of data encryption parameters when a Set Data Encryption page is processed:

a) For SCSI transport protocols where initiator port names are required, the initiator port name; otherwise, the initiator port identifier;
b) indication of the target port through which the data encryption parameters were established;
c) key scope;
d) encryption mode;
e) decryption mode;
f) key;
g) supplemental decryption keys where supported;
h) algorithm index;
i) key instance counter;
j) CKOD;
k) CKORL;
l) U-KAD;
m) A-KAD; and
n) nonce.

A device server may have limited resources for storage of sets of data encryption parameters (i.e., it may not have enough resources to store a unique set of data encryption parameters for every I_T Nexus that it is capable of managing). A device server may release a previously established set of data encryption parameters when a Set Data Encryption page is processed and there are no unused resources available. The method of choosing which set of data encryption parameters to release is vendor specific. If the device server does release a previously established set of data encryption parameters to free the resource, it shall establish a unit attention condition for every affected I_T Nexus (see 4.2.19.5).

If the device server supports an encryption key scope of ALL I_T NEXUS, it shall have resources to save one set of data encryption parameters with this scope.

If the device server supports an encryption key scope of RESERVATION GROUP, it shall have resources to save one set of data encryption parameters with this scope.

If the device server supports an encryption key scope value of LOCAL, it shall have resources to save one or more sets of data encryption parameters with this scope.

The data encryption parameters that shall be used for an I_T nexus shall be established by the following order of precedence:

a) If the data encryption scope for the I_T nexus is set to LOCAL, RESERVATION GROUP, or ALL I_T NEXUS (see 4.2.19.6), the data encryption parameters set by the last Set Data Encryption page from that I_T Nexus; or

b) If the data encryption scope for the I_T nexus is set to PUBLIC:

    1) If there is a reservation in effect for the logical unit, the data encryption parameters that have been saved by the device server with a key scope of RESERVATION GROUP if any data encryption parameters have been saved with this key scope;

    2) the data encryption parameters that have been saved by the device server with a key scope of ALL I_T NEXUS if any data encryption parameters have been saved with this key scope; or

    3) the default data encryption parameters.

4.2.19.8 Key instance counter

The device server shall keep a counter for each key that it is managing called the key instance counter. All key instance counters shall be set to zero when a power on condition occurs. Any other event that sets, clears, or changes a parameter in a set of data encryption parameters, except the supplemental decryption keys, shall cause the key instance counter for that set of data encryption parameters to be incremented. The value of the key instance counter associated with the currently selected key for an I_T Nexus is reported in the Data Encryption Status page of SECURITY PROTOCOL IN command. The key instance counters are 32 bits and shall roll over to zero when incremented past their maximum value.

8.5.2.4 Data Encryption Capabilities page

Table E1 shows the format of the Data Encryption Capabilities page.

Table E1 – Data Encryption Capabilities page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (0010h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4				Reserved				
19								
	Encryption algorithm descriptor list							
20				Data Encryption Algorithm descriptor (first)				
	:							
				Data Encryption Algorithm descriptor (last)				
n								

See SPC-4 for a description of the PAGE LENGTH field.

Each Data Encryption Algorithm descriptor (see table E2) contains information about a data encryption algorithm supported by the device server. If more than one descriptor is included, they shall be sorted in ascending order of the value in the ALGORITHM INDEX field.

Table E2 – Data Encryption Algorithm descriptor

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM INDEX							
1	Reserved							
2	(MSB)	DESCRIPTOR LENGTH (m-3)						
3								(LSB)
4	Reserved	SDK_C	MAC_C	DED_C	DECRYPT_C		ENCRYPT_C	
5	Reserved		NONCE_C		IV_RN	IV_EOU	IV_WPU	IV_MU
6	(MSB)	MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED DATA BYTES						
7								(LSB)
8	(MSB)	MAXIMUM AUTHENTICATED KEY-ASSOCIATED DATA BYTES						
9								(LSB)
10		Reserved						
19								
20	(MSB)	ENCRYPTION ALGORITHM IDENTIFIER						
23								(LSB)

The ALGORITHM INDEX field is a device server assigned value associated with the algorithm that is being described. The value in the ALGORITHM INDEX field is used by the SECURITY PROTOCOL OUT command Set Data Encryption page to select this algorithm.

The ENCRYPT_C field (see table E3) indicates the encryption capabilities of the device.

Table E3 - ENCRYPT_C field values

Code	Description
0	The device server has no data encryption capability using this algorithm.
1	The device server has the ability to encrypt data using this algorithm in software.
2	The device server has the ability to encrypt data using this algorithm in hardware.
3	Reserved

The DECRYPT_C field (see table E4) indicates the decryption capabilities of the device.

Table E4 - DECRYPT_C field values

Code	Description
0	The device server has no data decryption capability using this algorithm.
1	The device server has the ability to decrypt data using this algorithm in software.
2	The device server has the ability to decrypt data using this algorithm in hardware.
3	Reserved

The supplemental decryption key capable (SDK_C) bit shall be set to one if the device server is capable of supporting one or more supplemental decryption keys. The supplemental decryption keys shall be used for decryption only. The SDK_C bit shall be set to zero if the device server is not capable of supporting supplemental decryption keys.

The distinguish encrypted data capable (DED_C) bit shall be set to one if the device server is capable of distinguishing encrypted data from unencrypted data when reading it from the medium. The DED_C bit shall be set to zero if the device server is not capable of distinguishing encrypted data from unencrypted data when reading it from the medium. If the ability to distinguish encrypted data from unencrypted data is format specific and a volume is mounted, the DED_C bit shall be set based on the current format of the medium. If no volume is mounted, the DED_C bit shall be set to one if the device server is capable of distinguishing encrypted data from unencrypted data in any format that the device server supports.

The message authentication code capable (MAC_C) bit shall be set to one if the algorithm includes a message authentication code added to encrypted blocks. The MAC_C bit shall be set to zero if the algorithm does not include a message authentication code added to encrypted blocks. If the inclusion of a message authentication code is format specific and a volume is mounted, the MAC_C bit shall be set based on the current format of the medium. If no volume is mounted, the MAC_C bit shall be set to one if the device server adds a message authentication code to data encrypted with this algorithm in any format that the device server supports.

The initialization vector medium unique (IV_MU) bit shall be set to one if the initialization vector used by the encryption algorithm is unique for each medium. The IV_MU) bit shall be set to zero if the initialization vector used by the encryption algorithm is not unique for each medium.

The initialization vector write pass unique (IV_WPU) bit shall be set to one if the initialization vector used by the encryption algorithm is unique for each write operation that over writes the same portion of the medium. The IV_WPU bit shall be set to zero if the initialization vector used by the encryption algorithm is not unique for each write operation that over writes the same portion of the medium.

The initialization vector encrypted object unique (IV_EOU) bit shall be set to one if the initialization vector used by the encryption algorithm is unique for each encrypted block on the medium. The IV_EOU bit shall be set to zero if the initialization vector used by the encryption algorithm is not unique for each encrypted object on the medium.

The initialization vector random number (IV_RN) bit shall be set to one if the initialization vector used by the encryption algorithm is either in part or wholly a random number. The IV_RN bit shall be set to zero if the initialization vector used by the encryption algorithm is not in part or wholly a random number.

The MAXIMUM UNAUTHENTICATED KEY-ASSOCIATED DATA BYTES field indicates the maximum size of the unauthenticated key-associated data (see 4.2.19.10) that the device server can support for this algorithm.

The MAXIMUM AUTHENTICATED KEY-ASSOCIATED DATA BYTES field indicates the maximum size of the authenticated key-associated data (see 4.2.19.10) that the device server can support for this algorithm.

Table E5 describes the values in the NONCE_C field.

Table E5 - NONCE_C field values

Code	Description
0	This algorithm does not require a nonce value.
1	The device server generates the nonce value.
2	The device server requires all or part of the nonce value to be provided by the application client.
3	The device server supports all or part of the nonce value provided by the application client. If the Set Data Encryption page that enables encryption does not include a nonce value descriptor, the device server generates the nonce value.

The ENCRYPTION ALGORITHM IDENTIFIER field contains an Encryption Algorithm Identifier (see SPC-4).

8.5.3.2 Set Data Encryption page

Table Y1 shows the parameter list format of the Set Data Encryption page.

Table Y1 – Set Data Encryption page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (0010h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (m-3)				
3								(LSB)
4	SCOPE			Reserved	SDK	LOCK	CKOD	CKORL
5	DECRYPTION MODE				ENCRYPTION MODE			
6	ALGORITHM INDEX							
7	KEY FORMAT							
8				Reserved				
17								
18	(MSB)			KEY LENGTH (n-19)				
19								(LSB)
20				KEY				
n								
N+1				KEY-ASSOCIATED DATA DESCRIPTORS LIST				
m								

The PAGE LENGTH field indicates the number of bytes of parameter data to follow. If the page length value results in the truncation of any field, 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 PARAMETER LIST.

The SCOPE field (see table Y2) indicates the scope of the data encryption parameters. Support for scope values of PUBLIC and ALL I_T NEXUS are mandatory for device servers that support the Set Data Encryption page.

Table Y2 – SCOPE field values

Code	Name	Description
0	PUBLIC	All fields other than the SCOPE field and lock bit shall be ignored. The I_T nexus shall use data encryption parameters that are shared by other I_T nexuses. If no I_T nexuses are sharing data encryption parameters, the device server shall use default data encryption parameters.
1	LOCAL	The data encryption parameters are unique to the I_T nexus associated with the SECURITY PROTOCOL OUT command and shall not be shared with other I_T nexuses.
2	RESERVATION GROUP	The data encryption parameters shall be shared with all participants in a reservation.
3	ALL I_T NEXUS	The data encryption parameters shall be shared with all I_T nexuses.
4 – 7		Reserved

See 4.2.19.9 for a description of the LOCK bit.

If the supplemental decryption key (SDK) bit is set to one, the key sent in this page shall be added to the set of data encryption parameters used by the device server for the selected scope. The KEY INSTANCE COUNTER shall not be incremented for supplemental decryption keys. The ENCRYPTION MODE and LOCK fields shall be ignored and the DECRYPTION MODE shall match the current setting for this scope. If the DECRYPTION MODE does not match the current settings for this scope 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 PARAMETER LIST.

If the device server does not currently have a saved set of data encryption parameters associated with the I_T Nexus that sent the Set Data Encryption page or the scope or decryption mode values do not match the values in that set of saved data encryption parameters, 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 PARAMETER LIST.

If the SDK bit is set to one and the SDK_C field is set to zero in the Data Encryption Algorithm descriptor field that matches the ALGORITHM INDEX in the Data Encryption capabilities page, 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 PARAMETER LIST.

If the device server is processing a Set Data Encryption page with the SDK bit set to one and does not have the resource available to store this key 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 MAXIMUM NUMBER OF SUPPLEMENTAL DECRYPTION KEYS EXCEEDED. Any previously saved supplemental decryption keys shall not be affected by this error.

Editors Note: MAXIMUM NUMBER OF SUPPLEMENTAL DECRYPTION KEYS EXCEEDED is a new ASC.

If the supplemental decryption key (SDK) bit is set to zero, the key sent in this page shall be the key used for both encryption and decryption. Any keys that have been previously stored by the device server shall be removed from memory, See 4.2.19.5.

If the clear key on de-mount (CKOD) bit is set to one the device server shall set the data encryption parameters to default values after completing a de-mount of a volume. If the CKOD bit is set to zero, the de-mounting of a volume shall not affect the data encryption parameters. If the CKOD bit is set to one and there is no volume mounted in the device, 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 PARAMETER DATA.

If the clear key on reservation loss (CKORL) bit is set to one the device server shall set the data encryption parameters to default values on the loss or change in scope of the reservation. If the CKORL bit is set to zero, the loss of a reservation shall not affect the data encryption parameters. If the CKORL bit is set to one and there is no reservation in affect for the I_T nexus associated with the SECURITY PROTOCOL OUT command, 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 PARAMETER DATA.