To:	INCITS T10 Committee
From:	Paul Entzel, Quantum
Date:	13 February 2006
Document:	T10/05-446r5
Subject:	SSC-3: Add commands to control data encryption

1 Revision History

Revision 0:
Posted to the T10 web site on 16 November 2005.

Revision 1:
Posted to the T10 web site on 12 December 2005. Includes:
1. In 3.1.X, change “ciphering process was performed by the device server” to “ciphering process was performed by a device server”.
2. In 7.X.4, change AK_C to MAC_C and the paragraph that describes it.
3. Add CKOD bit to Set Data Encryption page.
4. Add microcode update and other vendor specific events to the list of events that clear a key.
5. Add a paragraph to 4.2.19.5 describing the loss of a key due to a vendor specific event.
6. Change GLOBAL scope to ALL I_T NEXUS.
7. Added key generation concept (4.2.18.6 and 4.2.19.9) and removed the ENCRYPTE bit from the WRITE(6) and WRITE(16) commands.
8. Added descriptors for key-associated data (subclause 8.5). Add these descriptors to the Data Encryption Status page and the Set Data Encryption page.
9. Added Next Block Encryption Status page to report encryption status and KAD data for the next block to read (still looking for a better name for this page).
10. Added ALGORITHM INDEX field to the Data Encryption Status page.
11. Expand the ENCRYPT and DECRYPT fields (renamed ENCRYPTION MODE and DECRYPTION MODE) in the Set Data Encryption page and the Data Encryption Status page to 4 bits and add multiple modes from HP’s strawman proposal (with some name changes).
12. Add LOCK bit to Set Data Encryption and Data Encryption Status pages and a model clause to describe it.
13. Add maximum key-associated data lengths to the algorithm descriptors.
14. Added acronyms for A-KAD and U-KAD.
15. Added a key descriptor for the application client to set the nonce and for the device server to report this capability/requirement.

Revision 2:
Updated with comments from the SSC-3 WG meeting on 10 January 2006. Posted to the T10 web site on 17 January 2006. Includes these major changes:
1. Corrected various spelling and grammatical errors.
2. Added a paragraph in 4.2.19.3 describing behavior when an encrypted block fails to authenticate.
3. Added test to 4.2.19.5 detailing the data to be saved from Set Encryption Mode page processing for each scope.
4. In 4.2.19.9 corrected several instances of DATA SECURITY IN to DADAT SECURITY OUT.
5. Swapped several bytes in the Data Encryption Status page to better align with the Set Encryption Mode page.
6. Fixed the alignment in the Next Block Encryption Status page.
7. Add code value of 5h to Table N3.
8. Added Vendor Specific code ranges to the KEY FORMAT and KEY DESCRIPTOR TYPE tables.
9. Removed the AUTH and VALID bits in the DATA SECURITY descriptor format table with a multiple bit field called AUTHENTICATED. Added table D3 to describe this field’s values.

Revision 3:
Posted to the T10 web site on 19 January 2006. Converted from DATA SECURITY IN and DATA SECURITY OUT commands (new commands defined by previous revisions of this proposal) to SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands.

Revision 4:
Posted to the T10 web site on 31 January 2006. Contain numerous editorial changes in addition to the following technical changes:

1. Added a definition for the term "data encryption parameters" and switched to using this term in place of "encryption key and mode" and other similar terms.
2. Replaced "hard reset event" with "power-on" as a reason to clear a key.
3. Moved the order of precedence for selecting data encryption parameters from subclause 8.5.3.2 to the Managing Keys model clause (4.2.19.5).
4. Fixed the description of Key Generation to include size, initialization, and rollover.
5. Changed the name of the Next Block Status page to be Next Block Encryption Status page.
6. Added several editor's notes in purple indicating comments received that should be discussed more.
7. Modified the description of the PUBLIC scope to ignore all but SCOPE and LOCK.
8. Add READ REVERSE(6) and READ REVERSE(16) to the list of commands that will decrypt.
9. Added a sentence to 8.5.3.2 to reference 4.2.19.9 for a description of the LOCK bit.
10. Added Supported Key Format page and removed KBR_C bit from Data Encryption Algorithms descriptor.
11. Incorporate 06-050r2 with appropriate changes.

Revision 5 (we're up to 30 pages now):
Posted to the T20 web site on 13 February 2006: Contains changes recommended by the SSC-3 working group during the 8 February conference call:

1. Moved the list of things that can cause encryption to be disabled from 4.2.19.2 to 4.2.19.5 and reworded it to things that can cause a data encryption parameter resource to be released.
2. Reworded 4.2.19.2 to indicate encryption is managed on a per I_T Nexus basis.
3. Added a reference for DED_C in 4.2.19.3 and a note that "it is bad to not be able to distinguish encrypted data from non-encrypted data". Also added a paragraph describing the order of precedence in reporting bad key over authentication failure.
4. Changed 4.2.19.4 to no longer describe how to prevent exhaustive search attacks.
5. Moved the stuff about what is saved on a per I_T Nexus basis into its own sub-clause (4.2.19.6). Added some test to describe the scope for the I_T Nexus.
6. Combined all of the data encryption parameter structures into one and added a paragraph to describe resource limitations.
7. Changed "key generation" to "key instance counter" and made it 32 bits.
8. Added some reserved space to the Data Encryption Capabilities page.
9. Changed the SCOPE and SOURCE fields in the Data Encryption Status page to I_T NEXUS SCOPE and KEY SCOPE respectively.
10. Added RECOVER BUFFERED DATA to the list of commands that decrypt.
11. Added Data Encryption Management Capabilities page.
12. Added multiple purple editors' notes to cover the areas we still need to address.

2 General

A great deal of interest has been expressed to add access security to data recorded on removable medium in the wake of several high profile cases where medium containing sensitive data has been lost or stolen. Many back-up application support adding passwords to data sets and/or encrypting the data. The password scheme does not protect from the data being recovered using a different application that understands the format but does not honor the password protection. The encryption approach is safer, but introduces significant performance degradation when used.

The major disadvantage to encrypting the data concerns the performance impact. Software algorithms to encrypt the data are available, but they take significant time to perform and this reduces the available performance. Also, since the encryption process removes redundancy in the data, attempts to compress the data after encryption are usually unsuccessful. In order to regain the performance boost from compression, the compression must be done before encryption, so it must be done in software. This adds even more time overhead, reducing the performance even more.

This proposal provides a method where encryption could be moved into the device, solving the problems described above. This proposal provides a method to control these features by introducing a new protocol for use with the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands. The new protocol is named “tape data encryption”.

The SECURITY PROTOCOL IN command using TAPE DATA ENCRYPTION protocol can request several reports containing supported features, enabled modes, and status.

The SECURITY PROTOCOL OUT command using TAPE DATA ENCRYPTION protocol is used to set encryption keys and modes of protection.

Perhaps the most difficult problem to solve with the addition of this interface is how to deal with multiple initiators. One would think that the parameters controlling encryption and decryption would be I_T nexus specific, and that they would not be used without reservations in place. However, that won’t work in many environments:

1. If the target port is behind a protocol bridge, the device server can not tell one initiator from another. In this environment, parameters that are normally per I_T nexus are shared at the target device level and the protocol bridge is required to sort out the individual initiator ports.
2. To support copy managers, we either need to add the ability to pass encryption modes and keys via the EXTENDED COPY command, have the ability to pass the set-up from one I_T Nexus to another, or have the ability to share set-up.

Future enhancements to this feature may include adding a method to encrypt the data encryption key before passing it to the device server. The addition of this single feature was proving to be significantly more complex than what was already contained in the proposal. We decided to postpone discussion of this feature until a later proposal. This proposal establishes plenty of reserved fields and page codes so that it can be added without breaking everything else.

3 Changes to SSC-3

3.1 Additions to the definition clause (3)

3.1.W Data encryption parameters – A set of parameters accessible through the Set Data Encryption page (see 8.5.3.2) that controls the data encryption and decryption process in the device server. See 4.2.19.5.

3.1.X Encrypted Block – A Logical Block in which the data has been subjected to a ciphering processes by the device server. Within this standard, a logical block is only considered encrypted if the ciphering process was performed by a device server.

3.1.Y Unencrypted Block – A logical Block in which the data has not been subjected to a ciphering process by the device server.

In subclause 3.2 add the following acronyms:

A-KAD	Authenticated key-associated data
U-KAD	Unauthenticated key-associated data

3.2 Addition to the model clause (4)

Add the following subclause to clause 4:

4.2.19 Data Encryption

4.2.19.1 Data Encryption overview

An SSC-3 compliant device may contain hardware or software that is capable of encrypting the data within logical blocks to provide security against unauthorized access to that data. The SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands specifying the tape data encryption protocol provide a means for the application client to monitor and control the encryption process within the device. A device server that supports the SECURITY PROTOCOL OUT command shall also support the SECURITY PROTOCOL IN command.

The SECURITY PROTOCOL OUT command specifying the tape data encryption protocol is used to set data encryption parameters. The SECURITY PROTOCOL IN command specifying the tape data encryption protocol is used to discover the type of data security features supported by the device server, the current configuration of data security features, and status of the encryption and decryption process.

4.2.19.2 Encrypting data on the medium

The application client controls the data encryption process by use of the SECURITY PROTOCOL OUT command specifying the tape data encryption protocol. Data encryption shall be managed within the device server on a per I_T Nexus basis. Data encryption is enabled for an I_T Nexus after the device server successfully processes a SECURITY PROTOCOL OUT command that sends a Set Data Encryption page (see 8.5.3.2) with the ENCRYPTION MODE field set to ENCRYPT and a valid key. If the data encryption scope parameter for an I_T Nexus is set to PUBLIC (See 4.2.19.6) it may also be enabled by another I_T Nexus that establishes a set of data encryption parameters with a key scope of ALL I_T NEXUS or RESERVATION GROUP (see 4.2.19.7).

If data encryption is enabled for an I_T Nexus, all data received by the device server from that I_T Nexus as part of a WRITE(6) or WRITE(16) command shall be encrypted before being recorded on the medium. Filemarks written due to a WRITE FILEMARKS(6) or WRITE FILEMARKS(16) command shall not be affected by the encryption process.

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. Encrypted blocks shall have no impact on space or locate commands.

A device server that supports encryption should be able to distinguish encrypted blocks from unencrypted blocks. The device server reports its ability to distinguish 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 able to distinguish 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.

Editor's note: UNABLE TO DECRYPT DATA is a new ASC.

If the device server is able to distinguish 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 can not distinguish encrypted blocks from unencrypted blocks to decrypt data that was not encrypted if instructed to do so. Application clients assume the responsibility of checking the integrity of the data in this environment.

Editor’s note: UNENCRYPTED DATA ENCOUNTERED WHILE DECRYPTING is a new ASC.

A device server that supports encryption and has been configured to decrypt the data may be able to determine if the encryption key is correct for an encrypted block. If the device server is able to determine if 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 encryption key 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 the encrypted block.

Editor’s note: INCORRECT DATA ENCRYPTION KEY is a new ASC.

If the device server is able to authenticate 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 authentication 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 DATA AUTHENTICATION FAILED. The device server shall establish the logical position at the BOP side the encrypted block.

Editor’s note: DATA AUTHENTICATION FAILED is a new ASC.

A device server that is able to distinguish 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) authenticate the data after decrypting it.

If a device server is able to authenticate the data after decrypting it and determine if the wrong key was used to decrypt the data, and determines that both the wrong key was used and the data failed to authenticate, it shall report the wrong key condition.

4.2.19.4 Exhaustive-search attack prevention

To prevent an exhaustive-search attack from discovering the encryption key, the device server should provide a mechanism to prevent unlimited attempts at setting a data encryption key 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 is capable of distinguishing encrypted data from unencrypted data and is capable of authenticating the data encryption key, it should provide a method to limit the number of attempts that it allows the application client to set the decryption key when an incorrect key is used. The method to do this is beyond the scope of this standard.

If the device server has reached its limit on failed attempts to set the data encryption key and decrypt data, it shall disable decryption for all I_T nexuses. All subsequent SECURITY PROTOCOL OUT commands specifying the tape data encryption protocol and with the SP_SPECIFIC field set to Set Data Encryption page and the DECRYPT or ENCRYPT 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 medium is dismounted from the device or a hard reset event.

Editor’s note: DATA DECRYPTION KEY FAIL LIMIT REACHED is a new ASC.

4.2.19.5 Managing keys within the device server

The security provided by data encryption is only as good as the security used when managing the keys. For this reason, 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 following methods for an application client to send the keys used for encrypting or decrypting data;

a) sending the key in plain text; or
b) sending a vendor specific key reference.

Editor's note: the paragraph and unordered list above came from 06-050r2. It places a requirement on the device server to support one of the two currently defined methods of passing a key. This means a device server may not support only a vendor unique method of passing the keys. It also will need to be revisited each time a new key format is added. Do we need this requirement? Comment from Chris Williams (HP): "I don't know if we need this requirement. I can see methods that would use a vendor unique key entry method (keyboard, IR, etc), but would still want to verify the encryption state and other parameters. That said, there may not be ISV support for a drive that does not implement a standard key entry method. I expect most drives will implement a), and probably also b)."

A vendor unique 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 unique 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 data encryption parameters and the volume is dismounted;
b) the CKORL bit is set to one in the data encryption parameters and the I_T nexus that had enabled encryption loses its reservation;
c) a microcode update is performed on the device;
d) power on; or
e) 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 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 clear all memory containing a key value. A unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR shall be established for all other I_T nexus that are affected by the loss of the key, (i.e., any I_T nexus that is using a scope of PUBLIC and sharing the key.)

Editor's note: DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR is a new ASC.

If a device server processes a Set Data Encryption page that includes a key, the device server shall release any resources that it had allocated to store a key value set by a previous SECURITY PROTOCOL OUT command from that I_T nexus and shall clear all memory containing the key value. A unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR shall be established for all other I_T nexus that is affected by the change of the key (i.e., any I_T nexus that is using a scope of PUBLIC and sharing the key).

A device server shall save at most one set of data encryption parameters with a 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 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 clear any memory containing a key value. A unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR shall be established for any other I_T nexus that is affected by the change of the key (i.e. any I_T nexus that is using a scope of PUBLIC and sharing the key.)

A device server shall save at most one set of data encryption parameters with a 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 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 clear any memory containing a key value. A unit attention condition with the additional sense of DATA ENCRYPTION PARAMETERS CHANGED BY ANOTHER INITIATOR shall be established for any other I_T nexus that is affected by the change of the key, (i.e., any I_T nexus that is using a scope of PUBLIC and sharing the key.)

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 key.

Editor's note: DATA ENCRYPTION PARAMETERS CHANGED BY VENDOR SPECIFIC EVENT is a new ASC.

Editor's note: Do we want to make support for any particular scope settings mandatory?

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;

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.

A device server shall set the data encryption scope for an I_T Nexus to LOCAL when it successfully completes the processing of Set Encryption Parameters page with a scope value of LOCAL from that I_T Nexus. The device server shall only use the data encryption parameters established by the Set Encryption Parameters page with a scope of LOCAL for processing commands from the I_T Nexus that established the parameters.

EDITOR'S NOTE: What should the data encryption scope for an I_T Nexus be set to if it was set to LOCAL and the device server releases the associated data encryption parameters? For instance, an I_T Nexus sets its key with a LOCAL scope, then the key is lost due to a dismount or reservation loss or the device server releases the key to re-use the storage for a different I_T Nexus. I think the I_T Nexus should stay with a LOCAL scope, but the parameters should be set to default, I'm just not sure how to word that.

A device server shall set the data encryption scope for an I_T Nexus to RESERVATION GROUP when it successfully completes the processing of Set Encryption Parameters 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 Encryption Parameters 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.

EDITOR'S NOTE: The previous two paragraphs incorporate the suggestion that the I_T Nexus data encryption scope shall be set to PUBLIC when it was either ALL I_T NEXUS or RESERVATION GROUP and the key is lost. Is everyone OK with that?

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) algorithm index;
h) key instance counter;
i) CKOD;
j) CKORL;
k) U-KAD;
l) A-KAD; and
m) nonce.

A device server may have limited resources for storage of sets of data encryption parameter (i.e., it may not have enough resources to store a unique set of data encryption parameters for every I_T Nexus that it can manage). 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.

Editor's Note: Once the CKORL parameter was added to the other 2 data encryption parameter sets, they became identical and that made it kind of silly to repeat them. So, I made one parameter list and added the key scope field to it. Then I added the 4 paragraphs above this note to describe how the structures are used. This may solve everyone's resource limitation problems. If it doesn't solve all of the resource limitation issues, it may make it easier to address them. Then again, it may just mess things up even more.

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 scope of RESERVATION GROUP if any data encryption parameters have been save for this scope;

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

    3) the default data encryption parameters.

EDITOR'S NOTE: the above precedence list assumes that the default scope value for each I_T nexus is PUBLIC. If this is not true, a second choice is required in entry "a" to select the default settings.

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 at power on. Any other event that sets, clears, or changes the key shall cause the key instance counter for that key 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.

4.2.19.9 Encryption mode locking

There are conditions outside of the control of an application client which cause the device server to release the resources used to save the data encryption parameters (see 4.2.19.5) or change the data encryption parameters used to control the encryption of data. Each of these conditions cause the device server to establish a unit attention condition to report the change of operating mode, but the unit attention condition may not always be reported to the application client through protocol bridges and driver stacks.

The LOCK bit in the Set Data Encryption page is set to one to lock the I_T Nexus that issued the SECURITY PROTOCOL OUT command to the set of data encryption parameters established at the completion of the processing of the command. The I_T nexus remains locked to that set of data encryption parameters and key instance counter value until a hard reset or another SECURITY PROTOCOL OUT command including a Set Data Encryption page from the same I_T Nexus is processed.

If the device server processes a WRITE(6) or WRITE(16) command for an I_T Nexus that is locked to a set of data encryption parameters and key instance counter, and the key instance counter value has changed since the time it was locked, the device server shall terminate the command with CHECK CONDITION status, with the sense key set to DATA PROTECT, and the additional sense code set to DATA ENCRYPTION KEY INSTANCE COUNTER HAS CHANGED. All subsequent WRITE(6) and WRITE(16) commands shall also be terminated this way until a hard reset event occurs or a SECURITY PROTOCOL OUT command including a Set Data Encryption page from the same I_T Nexus is processed.

Editor's note: DATA ENCRYPTION KEY INSTANCE COUNTER HAS CHANGED is a new ASC.

4.2.19.10 Nonce generation

Some encryption algorithms require the use of a nonce value or number used once value. Encryption algorithms that require a nonce value typically use it so that each logical block that is encrypted with a given key has a unique initialization vector or tweaking constant. The uniqueness of the nonce value prevents encryption breaking methods that determine the encryption key based on how multiple logical blocks are encrypted.

For a given encryption algorithm, the device server may:

a) not require a nonce value;
b) generate its own nonce value;
c) require a nonce value or part of the nonce value be provided by the application client; or
d) be configurable with respect to the source of the nonce value.

The device server reports its capability with respect to nonce values in the Encryption Algorithm Descriptions (see 8.5.3.2). If the device server reports that it requires a nonce value from the application client and a Set Data Encryption page is processed that does not include a nonce value descriptor, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INCOMPLETE KEY-ASSOCIATED DATA SET.

Editor's note: INCOMPLETE KEY-ASSOCIATED DATA SET is a new ASC.

3.3 *Changes to the explicit command set clause (5)*

3.3.1 Addition to subclause 5.1 Summary of commands for explicit address mode

In table 13, add the following commands:

Command Name	Type	Opcode	Synchronization Operation Required	Reference
SECURITY PROTOCOL IN	O	A2h	No	SPC-4
SECURITY PROTOCOL OUT	O	B5h	No	SPC-4

3.4 *Changes to the implicit command set clause (6)*

3.4.1 Addition to subclause 6.1 Summary of commands for implicit address mode

In table 20, add the following commands:

Command Name	Type	Opcode	Synchronization Operation Required	Reference
SECURITY PROTOCOL IN	O	A2h	No	SPC-4
SECURITY PROTOCOL OUT	O	B5h	No	SPC-4

3.5 *Additions to the Parameters for sequential-access devices clause (8)*

8.5. Security protocol parameters

8.5.1 Security protocol overview

This subclause describes the protocols, pages, and descriptors used by sequential-access devices with the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands (see SPC-4). Table P1 describes the protocols defined by this standard.

Table P1 - PROTOCOL field values

Code	Description
00h – 1Fh	See SPC-4
20h	Tape Data Encryption protocol
21h -27h	Reserved
28h – FFh	See SPC-4

8.5.2 SECURITY PROTOCOL IN command specifying tape data encryption protocol

8.5.2.1 SECURITY PROTOCOL IN command specifying tape data encryption protocol overview

The SECURITY PROTOCOL IN (see SPC-4) command specifying Tape Data Encryption protocol requests the device server to return information about the data security methods in the device server and on the medium. The command supports a series of pages that are requested individually. An application client requests a page by using a SECURITY PROTOCOL IN command with the SECURITY PROTOCOL field set to Tape Data Encryption protocol and the SP_SPECIFIC field set to the page code requested.

The SP_SPECIFIC field (see Table B1) indicates the type of report that the application client is requesting.

Table B1 – SP_SPECIFIC field values

Code	Description	Reference
0000h	Supported Tape Data Encryption In pages page	8.5.2.2
0001h	Supported Tape Data Encryption Out pages page	8.5.2.3
0002h – 000Fh	Reserved	
0010h	Data Encryption Capabilities page	8.5.2.4
0011h	Supported Key Formats page	8.5.2.5
0012h	Data Encryption Management Capabilities page	8.5.2.6
0013h – 001Fh	Reserved	
0020h	Data Encryption Status page	8.5.2.7
0021h	Next Block Encryption Status page	8.5.2.8
0013h - FFFFh	Reserved	

The ALLOCATION LENGTH field specifies the maximum number of bytes that the device server may return (see SPC-4).

8.5.2.2 Supported Tape Data Encryption In pages page

Table C1 shows the format of the Supported Tape Data Encryption In pages page.

Table C1 – Supported Tape Data Encryption In pages page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0000h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4		SUPPORTED PAGE LIST						
N								

The SUPPORTED PAGE LIST field shall contain a list of all of the pages that the device server supports for the SECURITY PROTOCOL IN command specifying tape data encryption protocol in numerical order starting with 0000h.

8.5.2.3 Supported Tape Data Encryption Out pages page

Table D1 shows the format of the Supported Tape Data Encryption Out pages page.

Table D1 – Supported Tape Data Encryption Out pages page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0001h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4		SUPPORTED PAGE LIST						
n								

The SUPPORTED PAGE LIST field shall contain a list of all of the pages that the device server supports for the SECURITY PROTOCOL OUT command specifying the tape data encryption protocol in numerical order.

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

Byte \ Bit	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 Aalgorithm 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

Byte \ Bit	7	6	5	4	3	2	1	0
0	ALGORITHM INDEX							
1	Reserved							
2	(MSB)	DESCRIPTOR LENGTH (m-3)						
3								(LSB)
4	Reserved		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							
17								
18	ALGORITHM NAME LENGTH (n-19)							
19								
20	ALGORITHM NAME							
m								

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 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 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 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) field shall be set to one if the initialization vector used by the encryption algorithm is unique for each medium. The initialization vector medium unique (IV_MU) field 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) field 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 initialization vector medium unique (IV_MU) field 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) field shall be set to one if the initialization vector used by the encryption algorithm is unique for each encrypted object on the medium. The IV_EOU field 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) field 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 field 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 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 that the device server can support for this algorithm.

Editor's note: Per David Black's (EMC) suggestion, we need a model clause that we can reference here to describe what KAD is and how it may be used. Chris Williams (HP) has promised to provide just such a clause. The editor will then provide the references.

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 ALGORITHM NAME LENGTH contains the length in bytes of the ALGORITHM NAME field.

The ALGORITHM NAME field contains a null terminated, null padded UTF-8 format string that describes the encryption algorithm. The string shall contain the standardization authority that has registered the algorithm if there is a standard that defines it.

Editor's note: David Black (EMC) has volunteered to research with IETF the use of a registry to register algorithm names to simplify the correlation of an algorithm name to a set of requirements.

8.5.2.5 Supported Key Formats page

Table F1 shows the format of the Supported Key Formats page.

Table F1 – Supported Key Formats page

Byte \ Bit	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0011h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	SUPPORTED KEY FORMATS LIST							
n								

The SUPPORTED KEY FORMATS LIST field shall contain a list of all of the key formats that the device server supports for the Set Data Encryption page (see 8.5.3.2) in numerical order.

8.5.2.6 Data Encryption Management Capabilities page

Table M1 shows the format of the Data Encryption Management Capabilities page.

Table M1 – Data Encryption Management Capabilities page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0012h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (11)						
3								(LSB)
4	Reserved							
5	Reserved					LOCK_C	CKOD_C	KCORL_C
6	Reserved							
7	Reserved				AITN_C	RG_C	LOCAL_C	PUBLIC_C
8		Reserved						
15								

The LOCK_C bit shall be set to one if the device server supports the LOCK bit in the Set Data Encryption page. The LOCK_C bit shall be set to zero if the device server does not support the LOCK bit in the Set Data Encryption page.

The CKOD_C bit shall be set to one if the device server supports the CKOD bit in the Set Data Encryption page. The CKOD_C bit shall be set to zero if the device server does not support the CKOD bit in the Set Data Encryption page.

The CKORL_C bit shall be set to one if the device server supports the CKORL bit in the Set Data Encryption page. The CKORL_C bit shall be set to zero if the device server does not support the CKORL bit in the Set Data Encryption page.

The AITN_C bit shall be set to one if the device server supports a scope of ALL I_T NEXUS. The AITN_C bit shall be set to zero if the device server does not support a scope of ALL I_T NEXUS.

The RG_C bit shall be set to one if the device server supports a scope of RESERVATION GROUP. The RG_C bit shall be set to zero if the device server does not support a scope of RESERVATION GROUP.

The LOCAL_C bit shall be set to one if the device server supports a scope of LOCAL. The LOCAL_C bit shall be set to zero if the device server does not support a scope of LOCAL.

The PUBLIC_C bit shall be set to one if the device server supports a scope of PUBLIC. The PUBLIC_C bit shall be set to zero if the device server does not support a scope of PUBLIC.

8.5.2.7 Data Encryption Status page

Table S1 shows the format of the Data Encryption Status page.

Table S1 – Data Encryption Status page

Byte \ Bit	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0020h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	I_T NEXUS SCOPE			Reserved		KEY SCOPE		
5	DECRYPTION MODE				ENCRYPTION MODE			
6	ALGORITHM INDEX							
7	Reserved							
8	(MSB)	KEY INSTANCE COUNTER						
11								(LSB)
12	KEY-ASSOCIATED DATA DESCRIPTORS LIST							
n								

EDITOR'S NOTE: Kevin Butt (IBM) has asked if there should be a way to report the I_T Nexus that has set up the current values for scopes of ALL I_T NEXUS and RESERVATION GROUP.

The I_T NEXUS SCOPE field (see table S4) indicates the data encryption scope saved for the I_T nexus on which this command was received.

Table S4 - I_T NEXUS SCOPE field values

Code	Description
0	The data encryption scope value for the I_T nexus on which this command was received is PUBLIC.
1	The data encryption parameters associated with the I_T nexus on which this command was received were set by the I_T nexus on which this command was received with a scope value of LOCAL.
2	The data encryption parameters associated with the I_T nexus on which this command was received were set by the I_T nexus on which this command was received were set with a scope value of RESERVATION GROUP.
3	The data encryption parameters associated with the I_T nexus on which this command was received were set by the I_T nexus on which this command was received were set with a scope value of ALL I_T NEXUS.
4 – 7	Reserved

The KEY SCOPE field (see table S5) indicates the scope of the data encryption parameters currently associated with the I_T nexus on which this command was received.

Table S5 - KEY SCOPE field values

Code	Description
0	The data encryption parameters are default values.
1	The data encryption parameters are unique to the I_T nexus on which this command was received (i.e., they were set by this I_T nexus with a LOCAL scope).
2	The data encryption parameters were established by an I_T nexus other then the one on which this command was received with a scope value of RESERVATION GROUP.
3	The data encryption parameters were established by an I_T nexus other then the one on which this command was received with a scope value of ALL I_T NEXUS.
4- 7	Reserved

Table S2 defined the values for the ENCRYPTION MODE field.

Table S2 – ENCRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data encryption is disabled.
1h	EXTERNAL	The device server has been configured to treat the data associated with WRITE(6) and WRITE(16) commands as if it has been encrypted by a system that is compatible with the algorithm specified by the ALGORITHM INDEX field.
2h	ENCRYPT	The device server has been configured to encrypt all data that it receives for a WRITE(6) or WRITE(16) using the algorithm specified in the ALGORITHM INDEX field.
3h – Fh		Reserved

Table S3 defined the values for the DECRYPTION MODE field. See 4.2.19.3 for exception condition requirements.

Table S3 – DECRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data decryption is disabled. If the device server encounters an encrypted block while reading, it shall not allow access to the data.
1h	RAW	Data decryption is disabled. If the device server encounters an encrypted block while reading, it shall pass the encrypted block and any additional metadata affixed to the encrypted block to the host without decrypting it.
2h	DECRYPT	The device server has been configured to decrypt all data that is read from the medium in response to a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verifying when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key provided in the KEY field.
3h	MIXED	The device server has been configured to decrypt all data that is read from the medium that it determines was encrypted in response to a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verifying when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key provided in the KEY field. If the device server encounters unencrypted data when processing a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), RECOVER BUFFERED DATA, VERIFY(6), or VERIFY(16) command, the data shall be processed without decrypting.
4h – Fh		Reserved

The ALGORITHM INDEX field indicates which of the encryption algorithms reported by the SECURITY PROTOCOL IN command Data Encryption Capabilities page is selected. If the ENCRYPT and DECRYPT bits are both set to DISABLE, the value in the ALGORITHM INDEX field is undefined.

The KEYINSTANCE COUNTER field contains the value of the key instance counter (see 4.2.19.8) assigned to the key indicated by the KEY SCOPE field value.

If encryption and decryption are both disabled, the KEY-ASSOCIATED DATA DESCRIPTORS LIST field shall not be included in the page.

If encryption or decryption is enabled, the KEY-ASSOCIATED DATA DESCRIPTORS LIST field shall contain data security descriptors (see 8.5) describing attributes assigned to the key defined by the I_T NEXUS SCOPE and KEY SCOPE fields at the time the key was established in the device server by processing a Set Data Encryption page. If more than one key associated descriptor is included, they shall be order of increasing value of the DESCRIPTOR TYPE field. Descriptors shall be included as defined by the following paragraphs.

An unauthenticated key-associated data descriptor (see 8.5.4.3) shall be included if an unauthenticated key-associated data descriptor was included in the Set Data Encryption page that established the key in the device server. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the key.

An authenticated key-associated data descriptor (see 8.5.4.4) shall be included if an authenticated key-associated data descriptor was included in the Set Data Encryption page that established the key in the device server. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the A-KAD value associated with the key.

A nonce value descriptor (see 8.5.4.5) shall be included if a nonce value descriptor was included in the Set Data Encryption page that established the key in the device server. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the nonce value associated with the key. A nonce value descriptor may be included if no nonce value descriptor was included in the Set Data Encryption page that established the key in the device server. In this case, the KEY DESCRIPTOR field shall be set to the nonce value established by the device server for use with the selected key.

8.5.2.8 Next Block Encryption Status page

Table N1 shows the format of the Next Block Encryption Status page.

Table N1 – Next Block Encryption Status page

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)	PAGE CODE (0021h)						
1								(LSB)
2	(MSB)	PAGE LENGTH (n-3)						
3								(LSB)
4	(MSB)	LOGICAL OBJECT NUMBER						
11								(LSB)
12	COMPRESSION STATUS				ENCRYPTION STATUS			
13	ALGORITHM INDEX							
14	Reserved							
15								
16	KEY-ASSOCIATED DATA DESCRIPTORS							
n								

The LOGICAL OBJECT NUMBER field contains the logical object identifier of the next logical object (see 4.2.5.2)

The COMPRESSION STATUS field is described in table N2.

Table N2 – COMPRESSION STATUS field values

Code	Description
0h	The device server is incapable of determining if the next logical block to read has been compressed.
1h	The device server is capable of determining if the next logical block to read has been compressed, but is not able to at this time. Possible reasons are: a) The next logical block has not yet been read into the buffer; b) There was an error reading the next logical block; or c) There are no more logical blocks (i.e., end of data).
2h	The next logical block is not compressed.
3h	The next logical block is compressed.
4h – Fh	Reserved

The ENCRYPTION STATUS field is described in table N3.

Table N3 – ENCRYPTION STATUS field values

Code	Description
0h	The device server is incapable of determining if the next logical block to read has been encrypted.
1h	The device server is capable of determining if the next logical block to read has been encrypted, but is not able to at this time. Possible reasons are: a) The next logical block has not yet been read into the buffer; b) There was an error reading the next logical block; or c) There are no more logical blocks (i.e: end of data).
2h	The next logical block is not encrypted.
3h	The next logical block is encrypted by an algorithm that is not supported by this device server. The values in the KEY-ASSOCIATED DATA DESCRIPTORS fields contain information pertaining to the encrypted block.
4h	The next logical block is encrypted by an algorithm supported by this device server. The values in the ALGORITHM INDEX and KEY-ASSOCIATED DATA DESCRIPTORS fields contain information pertaining to the encrypted block.
5h	The next logical block is encrypted by an algorithm supported by this device server but the device server either is not enabled to decrypt or does not have the correct key or nonce value to decrypt the data.
6h – Fh	Reserved

The ALGORITHM INDEX field indicates which of the encryption algorithms reported by the SECURITY PROTOCOL IN command Data Encryption Capabilities page was used to encrypt the logical block. For values in the ENCRYPTION STATUS field (see table N3) that do not indicate the ALGORITHM INDEX field is valid, the algorithm index is undefined.

If the encryption status indicates that the next logical block is encrypted by a supported algorithm, the device server shall include in the KEY-ASSOCIATED DATA DESCRIPTORS field all key-associated data that is associated with the encrypted block that was recorded on the medium. If more than one key-associated data descriptor is include in the Next Block Encryption Status page, they shall be in increasing numeric order of the value in the DESCRIPTOR TYPE field.

An unauthenticated key-associated data descriptor (see 8.5.4.3) shall be included if any unauthenticated key-associated data is associated with the next logical block. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the encrypted block.

An authenticated key-associated data descriptor (see 8.5.4.4) shall be included if any authenticated key-associated data is associated with the next logical block. The AUTHENTICATED field shall indicate the status of the authentication done by the device server. The KEY DESCRIPTOR field shall contain the A-KAD value associated with the encrypted block.

A nonce value descriptor (see 8.5.4.5) shall be included if a nonce value was not generated by the device server (i.e., it was established by a nonce value descriptor included in the Set Data Encryption page use to set the key and algorithm used to encrypt the logical block.) or if the device server can not determine if the nonce was generated by the device server that encrypted the logical block. A nonce value descriptor may be included if the nonce value was generated by the device server that encrypted the logical block. The AUTHENTICATED field shall indicate the status of the authentication done by the device server. The KEY DESCRIPTOR field shall contain the nonce value associated with the encrypted block

8.5.3 SECURITY PROTOCOL OUT command specifying tape data encryption protocol

8.5.3.1 SECURITY PROTOCOL OUT command specifying tape data encryption protocol overview

The SECURITY PROTOCOL OUT command specifying the tape data security protocol is used to configure the data security methods in the device server and on the medium. The command supports a series of pages that are requested individually. An application client requests a page by using a SECURITY PROTOCOL OUT command with the SECURITY PROTOCOL field set to Tape Data Encryption protocol and the SP_SPECIFIC field set to the page code requested.

The SP_SPECIFIC field (see Table B2) indicates the type of page that the application client is sending.

Table B2 – SP_SPECIFIC field values

Code	Description	Reference
00h – 0Fh	Reserved	
10h	Set Data Encryption page	8.5.3.2
11h - FFh	Reserved	

If the SP_SPECIFIC field is set to a reserved or unsupported value, 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.

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

Byte \ Bit	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		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.

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 clear key on dismount (CKOD) bit is set to one the device server shall set the data encryption parameters to default values after completing a dismount of a volume. If the CKOD bit is set to zero, the dismounting 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.

Table Y3 defined the values for the ENCRYPTION MODE field.

Table Y3 – ENCRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data encryption is disabled.
1h	EXTERNAL	The data associated with WRITE(6) and WRITE(16) commands has been encrypted by a system that is compatible with the algorithm specified by the ALGORITHM INDEX field.
2h	ENCRYPT	The device server shall encrypt all data that it receives for a WRITE(6) or WRITE(16) using the algorithm specified in the ALGORITHM INDEX field and the key provided in the KEY field.
3h – Fh		Reserved

EDITOR'S NOTE: Kevin Butt (IBM) has asked for another encryption mode value named EXTERNAL WITH VERIFY. He will provide a description.

Table Y4 defined the values for the DECRYPTION MODE field. See 4.2.19.3 for configuration and exception condition requirements.

Table Y4 – DECRYPTION MODE field values

Code	Name	Description
0h	DISABLE	Data decryption is disabled. If the device server encounters an encrypted block while reading, it shall not allow access to the data.
1h	RAW	Data decryption is disabled. If the device server encounters an encrypted block while reading, it shall pass the encrypted and any additional metadata affixed to the encrypted block to the host without decrypting it.
2h	DECRYPT	The device server shall decrypt all data that is read from the medium in response to a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verifying when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key provided in the KEY field.
3h	MIXED	The device server shall decrypt all data that is read from the medium that it determines was encrypted in response to a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), or RECOVER BUFFERED DATA command or verifying when processing a VERIFY(6) or VERIFY(16) command. The data shall be decrypted using the algorithm specified in the ALGORITHM INDEX field and the key provided in the KEY field. If the device server encounters unencrypted data when processing a READ(6), READ(16), READ REVERSE(6), READ REVERSE(16), RECOVER BUFFERED DATA, VERIFY(6), or VERIFY(16) command, the data shall be processed without decrypting.
4h – Fh		Reserved

EDITOR'S NOTE: Kevin Butt (IBM) has asked for another decryption mode value named RAW WITH KEY CHANGE NOTIFICATION. He will provide a description.

If the device server does is not capable of distinguishing encrypted blocks from unencrypted blocks using the algorithm specified in the ALGORITHM INDEX field and the DECRYPTION MODE field is set to MIXED, 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 ENCRYPTION MODE field is set to ENCRYPT and the KEY LENGTH field is set to zero, 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 DECRYPTION MODE field is set to DECRYPT or MIXED and the KEY LENGTH field is set to zero, 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.

The ALGORITHM INDEX field indicates which of the encryption algorithms reported by the SECURITY PROTOCOL IN command Data Encryption Capabilities pages shall be used to encrypt and decrypt data.

The KEY FORMAT field indicates the format of the value in the KEY field. Values for this field are described in table Y5.

Table Y5 – KEY FORMAT field values

Code	Description
00h	The KEY field contains the key to be used to encrypt or decrypt data in plain text.
01h	The KEY field contains a vendor specific key reference
02h – BFh	Reserved
C0h – FFh	Vendor specific

Editor's note: Chris Williams (HP) asks: "Are we specifying the format of the plaintext key?"

Editor's note: Comment from David Black (EMC):

For discussion purposes, please put the following in an Editor's note, and I'll take the brickbats on the next concall:

The only transports for which this is even passably acceptable are parallel SCSI, and perhaps SAS - it is not acceptable for iSCSI or for FC fabrics. The intended request is for a "shall implement" requirement, not a "shall use" requirement - if a customer wants to pass keys in the clear, we don't need to prevent that, but it is wrong to force a customer to pass keys in the clear because there's no other way to do it.

A simple unauthenticated key exchange mechanism that uses the exchanged key to encrypt the encryption key solely for transmission to the device is sufficient to solve this problem and can be designed in a matter of months. The primary purpose of such a mechanism is to prevent an eavesdropper from learning the encryption key. The mechanism design would be based on at least these three simplifying design assumptions:

- No authentication. Neither the initiator nor the device does anything to prove its identity. There's no way to cryptographically prove or determine who encrypted the key, aside from knowing what initiator sent the encrypted key.
- No persistence. The encryption key is encrypted for transport to the device and immediately decrypted. There is no need to remember what key it was encrypted with.
- No repeat. Every time an encryption key is sent (even the same key), it can or will be encrypted differently.

One approach is to have the device generate an RSA public/private key pair on startup and use the public key to encrypt the encryption key, although the RSA operations will be slow, suggesting that a better optimized (but more complex) design be used. A Diffie-Hellman key exchange is another possible building block.

Interoperability is also an issue - as the proposal is currently written, interoperability would require that "initiators shall support both methods" in 4.2.19.5 - this is already noted as an open issue.

The KEY LENGTH field indicates the length of the key field in bytes.

If the KEY FORMAT field is 01h, the KEY field shall contain 8 bytes of T10 vendor identification (see SPC-4) followed immediately by a vendor specific key reference identifying the key to be used to encrypt or decrypt data. If the KEY field contains a vendor specific key reference that is unknown to the device server or the T10 vendor identification does not match the T10 vendor identification provided by the device server, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to VENDOR SPECIFIC KEY REFERENCE NOT FOUND.

Editors Note: VENDOR SPECIFIC KEY REFERENCE NOT FOUND is a new ASC.

Editor's note: Several comments have been received that indicate there is little value provided by the requirement to include the T10 vendor identification with the key reference. Gideon Avida (Decru) commented: "If what we are trying to achieve is to allow multiple key management systems in the same environment, how about adding a "domain ID"? Then the pair "domain ID"/key ID should be unique in the environment."

If the ENCRYPTION MODE field is set to ENCRYPT the device server shall save the key-associated descriptors in the KEY-ASSOCIATED DATA DESCRIPTORS LIST field and associate them with every logical block that is encrypted with this key by the device server. If more than one key-associated data descriptor is include in the Set Data Encryption page, they shall be in increasing numeric order of the value in the DESCRIPTOR TYPE field. If the ENCRYPTION MODE field is not set to ENCRYPT and key-associated descriptors are included in the KEY-ASSOCIATED DATA DESCRIPTORS LIST field, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

An unauthenticated key-associated data descriptor (see 8.5.4.3) may be included if any unauthenticated key-associated data is to be associated with logical blocks encrypted with the algorithm and key. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the U-KAD value associated with the encrypted block.

An authenticated key-associated data descriptor (see 8.5.4.4) may be included if any authenticated key-associated data is to be associated with the next encrypted block. The AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the A-KAD value associated with the encrypted block.

EDITOR'S NOTE: I have received several comments that the AUTHENTICATED field value should be specified as 1 instead of zero. I felt that the field's value adds no value at this point, but by specifying it as zero we have the flexibility to add meaning to other codes in the future. Is this reasonable?

If a nonce value descriptor (see 8.5.4.5) is included and the algorithm and the device server supports application client generated nonce values, the value in the KEY DESCRIPTOR field shall be used as the nonce value for the encryption process. If the encryption algorithm or the device server does not support application client generated nonce values and a nonce value descriptor is included, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST. If the encryption algorithm or the device server requires an application client generated nonce value and a nonce value descriptor is not included, the device server shall terminate the command with CHECK CONDITION, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INCOMPLETE KEY-ASSOCIATE DATA SET. If a nonce value descriptor is included, the AUTHENTICATED field shall be set to zero. The KEY DESCRIPTOR field shall contain the nonce value associated with the encrypted block

8.5.4 SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT descriptors

8.5.4.1 Tape data encryption protocol descriptors overview

Several of the parameter pages in used by the SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT commands allow for the inclusion of descriptors to provide additional optional data. This subclause defines the descriptors that are common between multiple pages.

8.5.4.2 Tape data encryption descriptors format

Each tape data encryption descriptor shall take the form as defined in table D1.

Table D1 –Tape data encryption descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	KEY DESCRIPTOR TYPE							
1	Reserved					AUTHENTICATED		
2	(MSB)	KEY DESCRIPTOR LENGTH (n-3)						
3								(LSB)
4	(MSB)	KEY DESCRIPTOR						
n								(LSB)

The KEY DESCRIPTOR TYPE field contains a value from table D2 that defines the contents of the KEY DESCRIPTOR field

Table D2 – KEY DESCRIPTOR TYPE field values

Code	Description	Reference
00h	Unauthenticated key-associated data	8.5.4.3
01h	Authenticated key-associated data	8.5.4.4
02h	Nonce value	8.5.4.5
03h – BFh	Reserved	
C0h – FFh	Vendor specific	

Table D3 defines the values for the AUTHENTICATED field.

Table D3 – AUTHENTICATED field values

Code	Description
0	The value in the KEY DESCRIPTOR field is not covered by the authentication (i.e. U-KAD)
1	No attempt has been made to authenticate the value in the KEY DESCRIPTOR field.
2	The value in the KEY DESCRIPTOR field has been authenticated.
3	The value in the KEY DESCRIPTOR field has failed authentication.
4 - 7	Reserved

8.5.4.3 Unauthenticated key-associated data key descriptor

The AUTHENTICATED field in an unauthenticated key-associated data descriptor shall be set to zero.

The KEY DESCRIPTOR field of an unauthenticated key-associated data descriptor shall contain any unauthenticated key-associated data assigned to the key.

8.5.4.4 Authenticated key-associated data key descriptor

The AUTHENTICATED field shall be set as defined by the page in which the descriptor is included.

The KEY DESCRIPTOR field of an authenticated key-associated data descriptor shall contain any authenticated key-associated data assigned to the key.

8.5.4.5 Nonce value descriptor

The AUTHENTICATED field shall be set as defined by the page in which the descriptor is included.

The KEY DESCRIPTOR field of a nonce value descriptor shall contain the nonce value used with the data encryption key.