ENDL
TEXAS

Date: 10 June 2008
To: T10 Technical Committee
From: Ralph O. Weber
Subject: Capability-based Command Security (CbCS) [the rewrite]

Introduction

The rush to approve 07-454r5 concurrently with other security proposals such as IKEv2-SCSI and ESP-SCSI has lead to several nettlesome problems. Some prime examples are:

- The ability of the RECEIVE CREDENTIAL command to be used for a plaintext attack on the CbCS SA.
- There is too much dependence on a specific I_T Nexus for SA usage.

Rather than nickel-and-dime these changes, it is proposed to translate 07-454r5 into text that is at least 90% ready for incorporation in SPC-4 (i.e., the content of this proposal), review/revise that, and approve the result as a replacement for the already approved 07-454r5.

Related/Source Documents

This proposal incorporates material from the following T10 proposals:

07-454r5	Capability based Command Security
08-101r1	SPC-4: CbCS field byte alignment changes
08-128r0	SPC-4 RECEIVE CREDENTIAL command 'adjustments'
08-129r0	SPC-4 CbCS capability validation omissions
08-138r0	Constraints on SPC-4 SA creation based on Usage Type
08-141r0	CbCS SECURITY PROTOCOL IN/OUT tweaks in SPC-4

Revision History

r0 Initial revision
r1 Incorporated changes requested by March CAP working group and Sivan Tal. Updated SPC-4 subclause, table, etc. references to SPC-4 r14. Added CbCS page format definitions. Modified the CbCS validation requirements for the SECURITY PROTOCOL IN command to facilitate security token transfers and similar tasks.
r2 Incorporated changes requested by April 22 conference call.
r3 Incorporated changes requested by the May CAP working group.

Changes between r2 and r3 are indicated by change bars.

Unless otherwise indicated additions are shown in blue, deletions in ~~red strikethrough~~, and comments in green.

18484 Preston Road, Suite 102 PMB 178, Dallas, TX 75252 214-912-1373 972-596-2775 Fx ENDL_TX@computer.org

Proposed Changes in SPC-4 r13

3.1 Definitions

{{Insert the following alphabetically.}}

3.1.a CbCS capability: A descriptor that is a component of a CbCS credential (see 3.1.c) and a CbCS extension descriptor (see 5.13.6.8.16) that specifies access to a logical unit or volume (see SSC-3) for specific commands. See 5.13.6.8.13.

3.1.b CbCS capability key: A shared key (see 3.1.147) that is cryptographically associated with a specific CbCS capability (see 3.1.a).

3.1.c CbCS credential: The combination of a CbCS capability (see 3.1.a) and a CbCS capability key (see 3.1.b) that is returned by a RECEIVE CREDENTIAL command (see 6.r). See 5.13.6.8.12.

3.1.d CbCS Management Application Client class: The CbCS (see 5.13.6.8) class whose objects manage, or an object that manages CbCS shared keys (see 5.13.6.8.11). See 5.13.6.8.4.

3.1.e CbCS Management Device Server class The CbCS (see 5.13.6.8) class whose objects prepare, or an object that prepares CbCS credentials (see 5.13.6.8.12) in response to RECEIVE CREDENTIAL command (see 6.r) requests from secure CDB originator (see 5.13.6.8.5). See 5.13.6.8.3.

3.1.f CbCS master key: A shared key (see 3.1.147) from which CbCS working keys (see 3.1.h) are derived that is composed of an authentication key component and a generation key component. See 5.13.6.8.11.

3.1.g CbCS security token: A random nonce (see 3.1.107) that is chosen by the enforcement manager (see 5.13.6.8.7) for a given I_T nexus and returned to a secure CDB originator (see 5.13.6.8.5). See 5.13.6.8.10.

3.1.h CbCS working keys: Shared keys (see 3.1.147) that are used to cryptographically associate a CbCS capability (see 3.1.a) with a CbCS capability key (see 3.1.b). See 5.13.6.8.11.

…

4.3.4.2 The XCDB format

…

The EXTENSION TYPE field (see table 12) specifies the size and format of the extension parameters that follow in the XCDB descriptor.

Table 12 — EXTENSION TYPE field

Code	Descriptor Order [a]	Description	Extension size (bytes)	Reference
40h	first	CbCS extension descriptor	140	5.13.6.8.16
all others	Reserved			

[a] The order in which XCDB descriptors appear in an XCDB is arranged so that all the XCDB descriptors that follow an XCDB descriptor defined in a future version of this standard are also XCDB descriptors defined in a future version of this standard (i.e., after encountering one unrecognized XCDB descriptor, all subsequent XCDB descriptors are also going to be unrecognized).

{{The reference to the table footnote should have appeared in SPC-4 r12, but it was not present.}}

…

5.13.2.2 SA parameters

…

The USAGE_TYPE SA parameter shall be one of the values shown in table 49.

Table 49 — USAGE_TYPE SA parameter values

Value [a]	Description	Usage model	Usage data description	Reference
0000h - 0080h	Reserved			
0081h	Tape data encryption	ESP-SCSI [b]	None [c]	SSC-3
0082h - 8000h	Reserved			
8001h	CbCS authentication and credential encryption	ESP-SCSI [b]	None [b]	5.13.6.8
8002h - FFFFh	Reserved			
~~0082h - FFFFh~~	~~Reserved~~			

[a] USAGE_TYPE values between 8000h and CFFFh inclusive place additional constraints on how an SA is to be created as described in 7.6.3.5.13.

[b] ESP-SCSI usage is defined in 5.13.7.

[c] The usage data length field in the IKEv2-SCSI SAUT Cryptographic Algorithms payload (see 7.6.3.5.13) shall contain zero.

…

5.13.6.8 Capability-based command security technique

5.13.6.8.1 Overview

CbCS is a credential-based system that manages access to a logical unit by the coordination between shared keys and security ~~attributes~~ parameters set by the CbCS management application client (see 5.13.6.8.4) and credentials generated by the CbCS management device server (see 5.13.6.8.3). The mechanism for coordination between the CbCS management device server and the CbCS management application client is not defined in this standard.

…

{{At the bottom of figure 15, change the Receive shared keys() operation in the Enforcement Manager to Exchange shared keys() and change the Send shared keys() operation in the CbCS Management Application Client to Exchange shared keys(). This is necessary to correctly reflect the nature of the Diffie-Hellman protocol used to manage the shared keys. Also change Receive/Send security settings() to Receive/Send CbCS parameters. This is necessary to align the figure with the nomenclature used in this proposal.}}

…

5.13.6.8.2 Security Manager class

The Security Manager class for the CbCS technique manages access of secure CDB originators to logical units or volumes (see SSC-3). It uses a decision database to obtain the authorization information required for deciding the type and duration of access granted to secure CDB originator to a given logical unit (see 3.1.71) or volume (see SSC-3). It communicates with secure CDB originators to provide them CbCS credentials ~~(see x.x.x)~~ (see 6.r.2.3), and with enforcement managers (see 3.1.38) to ~~provide them~~ exchange shared keys (see 5.13.6.8.11) ~~(see 3.1.147)~~ and security ~~settings~~ parameters (see 5.13.6.8.15).

…

5.13.6.8.3 CbCS Management Device Server class

5.13.6.8.3.1 CbCS Management Device Server class overview

The CbCS Management Device Server class returns a CbCS capability and a CbCS capability key (i.e., Capability-Key) with each CbCS credential giving the secure CDB originator access to a specific logical unit, and optionally to a volume (see SSC-3).

This standard defines the RECEIVE CREDENTIAL command (see 6.r) that the secure CDB originator may use to request a CbCS capability and a CbCS capability key from a CbCS management device server.

~~The CbCS management device server shall authenticate the secure CDB originator unless the BASIC CbCS method is used (see x.x.x).~~

Commands to and responses from the CbCS management device server are protected by use of an SA whose creation included the Authentication Step (see 6.r).

The CbCS management device server shall set the CBCS bit to zero in any Extended INQUIRY Data VPD page (see 7.7.4) that it returns.

…

5.13.6.8.4 CbCS Management Application Client class

The CbCS Management Application Client class ~~sends~~ exchanges shared keys (see 5.13.6.8.11) ~~(see x.x.x)~~ with and sends CbCS parameters (see 5.13.6.8.15) to ~~security settings to~~ the enforcement manager using SECURITY PROTOCOL OUT commands (see 6.31) and SECURITY PROTOCOL IN commands (see 6.30) sent over the SCSI domain's service delivery subsystem.

The CbCS capability keys are computed by the CbCS management device server using shared keys that are shared between the:

a) Enforcement manager;
b) CbCS management application client; and
c) CbCS management device server.

The shared keys are managed by the security manager. ~~This standard defines SCSI commands (see clause 6) the CbCS management application client may use to set and manage the shared keys stored in an enforcement manager.~~

{{The commands are SECURITY PROTOCOL IN and SECURITY PROTOCOL OUT, and they are already mentioned by name in the first paragraph of 5.13.6.8.4.}}

5.13.6.8.5 Secure CDB Originator class

The Secure CDB Originator class requests CbCS capabilities and CbCS capability keys from the CbCS management device server for a specific logical unit (see 3.1.71) or volume (see SSC-3). The secure CDB originator sends the CbCS capability ~~(see x.x.x)~~ and ~~CbCS validation tag~~ integrity check value ~~(see x.x.x)~~ to the logical unit's secure CDB processor as part of a CbCS extended CDB as described in 5.13.6.8.16.

For more information on the Secure CDB Originator class see 5.13.6.2.

5.13.6.8.6 Secure CDB Processor class

The Secure CDB Processor class:

a) Receives a CbCS capability descriptor (see 6.r.2.3) ~~(see x.x.x)~~ from a secure CDB originator in a CbCS extension descriptor (see 5.13.6.8.16);
b) Requests the SCSI command be validated by the enforcement manager; and
c) If the Enforcement Manager validates the SCSI command, then the secure CDB processor processes that SCSI command.

The secure CDB processor indicates that CbCS is applied to a logical unit by setting the CBCS bit to one in the Extended INQUIRY Data VPD page (see 7.7.4). If the CbCS bit is set to one, the logical unit shall support the following:

a) Extended CDBs (see 4.3.4);
b) CbCS extension type (see 5.13.6.8.16) ~~(see x.x.x)~~;
c) SECURITY PROTOCOL IN commands (see 6.30) specifying the CbCS security protocol (see 7.6.c) ~~(see x.x.x)~~; and
d) SECURITY PROTOCOL OUT commands (see 6.31) specifying the CbCS security protocol (see 7.6.c) ~~(see x.x.x)~~.

For more information on the Secure CDB Processor class see 5.13.6.3.

~~Editors Note 2 - ROW: The CBCS bit described above is not currently defined in the Extended INQUIRY Data VPD page. Its definition appears in the yet-to-be-approved 08-145.~~

5.13.6.8.7 Enforcement Manager class

The Enforcement Manager class:

a) Receives shared keys (see 5.13.6.8.11) ~~(see 3.1.147)~~ and other ~~security settings (see x.x.x)~~ CbCS parameters (see 5.13.6.8.15) from the CbCS management application client;
b) Authenticates the CbCS capability received from a secure CDB processor ~~(see x.x.x)~~ with an integrity check value ~~(see x.x.x)~~ ~~using the CbCS capability received from a secure CDB processor:~~ as described in 5.13.6.8.13.2; ~~and~~
c) Validates SCSI commands sent by secure CDB originators~~.~~ as described in 5.13.6.8.13.2; and
d) If the CAPKEY CbCS method (see 5.13.6.8.8.3) is supported, supplies one security token (see 5.13.6.8.10) for each active I_T nexus to the secure CDB processor for delivery to the secure CDB originator that is using that I_T nexus.

The enforcement manager may be contained within the secure CDB processor, or within the SCSI target device. If the enforcement manager is contained within the secure CDB processor then, the shared keys and CbCS parameters ~~security settings~~ it uses pertain to ~~the~~ that logical unit (see 3.1.71) ~~(see x.x.x)~~. If the enforcement manager is contained within the SCSI target device then, the shared keys and CbCS parameters ~~security settings~~ it uses pertain to the SCSI target device, and the ~~security protocol~~ SECURITY PROTOCOL well-known logical unit (see 8.5) is used for the commands ~~to set~~ that exchange shared keys and set CbCS parameters ~~security settings~~.

If ~~the~~ a shared key is stored in a well known logical unit then ~~the~~ that key is shared between all logical units within the SCSI target device but shall only used by a logical unit if there has been no shared key assigned to that logical unit (i.e., a shared key assigned to a logical unit always overrides any shared key assigned to a well known logical unit).

For more information on the Enforcement Manager class see 5.13.6.4.

5.13.6.8.8 CbCS methods

{{All of 5.13.6.8.8 is new. Editing markups suspended.}}

5.13.6.8.8.1 Overview

The CbCS methods defined by this standard are summarized in table x1.

Table x1 — CbCS methods

CbCS method	Protection provided by the enforcement manager and secure CDB processor	Level of security provided, including I_T nexus security provided by SCSI transport (e.g., FC-SP)	
		Without I_T nexus security	With I_T nexus security
BASIC	Protection against errors provided by verifying that the CbCS capability allows processing, but no validation of the authenticity of the CbCS capability.	No protection against attacks	Same protection as is provided by the SCSI transport on the I_T nexus
CAPKEY	Protection against errors and some attacks by both verifying that the CbCS capability allows processing, and validating the authenticity of the CbCS capability.	CbCS capability authenticity assured, but still subject to network attacks (e.g., replay attacks)	CbCS capability authenticity assured and bound to an I_T nexus; other network attacks (e.g., data privacy) thwarted by SCSI transport security on the I_T nexus

If a secure CDB processor receives a command for a logical unit that has CbCS enabled, the enforcement manager shall validate the command as described in 5.13.6.8.13.2 before any other field in the CDB is validated, including the operation code.

5.13.6.8.8.2 The BASIC CbCS method

The BASIC CbCS method validates that the CbCS capability authorizes the encapsulated command for each CDB. It provides centrally-managed policy-driven command access control mechanism that enforces authorized access based on capabilities.

The BASIC CbCS method does not validate the authenticity of the CbCS capability.

Preparing CbCS credentials for the BASIC CbCS method does not require the knowledge of CbCS shared keys and may be done by the secure CDB originator without coordination with the CbCS management device server. In the CbCS extension descriptor (see 5.13.6.8.16):

a) The CbCS capability descriptor (see 6.r.2.3) CBCS METHOD field is set to BASIC;
b) The following CbCS capability descriptor fields are ignored:
   A) The KEY VERSION field; and
   B) The INTEGRITY CHECK VALUE ALGORITHM field;
   and
c) The INTEGRITY CHECK VALUE field is set to zero.

The BASIC CbCS method controls access between the secure CDB originator and the secure CDB processor without requiring authentication of the secure CDB originator. It is sufficient for the secure CDB processor to request that the enforcement manager verify the CbCS capabilities sent by the secure CDB originator.

5.13.6.8.8.3 The CAPKEY CbCS method

The CAPKEY CbCS method provides centrally-managed policy-driven command access control mechanism that enforces authorized access based on capabilities.

In addition, the CAPKEY CbCS method assures the integrity and authenticity of the CbCS capability transferred with each command.

The CAPKEY CbCS method provides for security of commands delivered to the secure CDB processor. When used in conjunction with a secure service delivery subsystem, it provides additional protection against network attacks (see 5.13.6.8.8.1).

Each capability (see 5.13.6.8.13) is cryptographically associated with a capability key, and the pair is returned to a secure CDB originator (see 5.13.6.8.5) in credential (see 5.13.6.8.12) in response to a RECEIVE CREDENTIAL command (see 6.r).

When the capability is associated with a specific command using a CbCS extension descriptor (see 5.13.6.8.16), an integrity check value is computed using the capability key and a CbCS security token (see 5.13.6.8.10). The enforcement manager (see 5.13.6.8.7) validates this integrity check value as described in 5.13.6.8.13.3

5.13.6.8.9 CbCS trust assumptions

{{All of 5.13.6.8.9 is new. Editing markups suspended.}}

After the logical unit is trusted (i.e., after a secure CDB originator authenticates that it is communicating with a specific logical unit), the secure CDB originator trusts the secure CDB processor and the enforcement manager to do the following:

a) Deny any access attempt from any application client that is not authorized by the security manager; and
b) Deny access from any application client that does not perform the CbCS protocols and functions defined by this standard.

The CbCS management device server and the CbCS management application client are trusted after:

a) The CbCS management device server is authenticated by the secure CDB originator; and
b) The CbCS management application client is authenticated by the CbCS Enforcement Manager.

The CbCS management device server and the CbCS management application client are trusted to do the following:

a) Securely store long-lived shared keys and capability keys;
b) Grant credentials to secure CDB originators according to access control policies that are outside the scope of this standard; and
c) Perform the defined security functions.

The service delivery subsystem between the secure CDB originator and the secure CDB processor is not trusted. However, the CbCS security model for CbCS methods other than BASIC is defined so that commands generated by the secure CDB originator are processed by the secure CDB processor only after the secure CDB originator interacts with both the CbCS management device server and the secure CDB processor as defined in this standard.

The communications trust requirements shown in table x2 provide a basis for the CbCS trust assumptions.

Table x2 — CbCS communications trust requirement

For connections between	CbCS cryptographic communications trust requirements	Requirement level
secure CDB originator and secure CDB processor	Message integrity [b]	Optional [a]
secure CDB originator and CbCS management application client	Message confidentiality [c] and integrity [b]	Mandatory
CbCS management application client and enforcement manager	Message integrity [b]	Mandatory
CbCS management application client and CbCS management device server	Message confidentiality [c] and integrity [b]	Mandatory

[a] If this requirement is not met, then the conditions that table x1(see 5.13.6.8.8) show for an I_T nexus without security apply.
[b] Message integrity algorithms are those that table 79 (see 5.13.8) describes as integrity checking (i.e., AUTH) algorithms.
[c] Message confidentiality algorithms are those that table 79 (see 5.13.8) describes as encryption algorithms.

5.13.6.8.10 CbCS security tokens

{{All of 5.13.6.8.10 is new. Editing markups suspended.}}

A CbCS security token is a random nonce (see 3.1.107) that is at least eight bytes in length that is chosen by the enforcement manager (see 5.13.6.8.7) and returned to a secure CDB originator (see 5.13.6.8.5) by the secure CDB processor (see 5.13.6.8.6) in response to a SECURITY PROTOCOL IN command (see 6.30) with the SECURITY PROTOCOL field set to 07h (i.e., CbCS) and the SECURITY PROTOCOL SPECIFIC field set to 3Fh (i.e., the Security Token CbCS page) as described in 7.6.c.3.4.

The security token shall be unique to each instance of an I_T nexus known to the secure CDB processor and enforcement manager. Security tokens shall be reset and maintained as described in this subclause.

Each security token shall contain at least as many bytes as the largest cipher block size for all the integrity checking algorithms supported by the SCSI target device (see 7.6.c.3.3).

If a hard reset, logical unit reset, or I_T nexus loss is detected by the secure CDB processor, the enforcement manager shall be notified of the event once for each affected I_T nexus. In response to such a notification the enforcement manager shall discard the security token, if any, associated with the affected I_T nexus.

After a power on, the enforcement manager shall discard all security tokens, if any, that it had been maintaining before the power on.

In response to a request for a security token for a given I_T nexus from the secure CDB processor, the enforcement manager shall do one of the following:

a) Return the security token value, if any, that is being maintained for the specified I_T nexus to the secure CDB processor; or
b) If no security token is being maintained for the specified I_T nexus, then:
   1) A security token shall be prepared;

2) The security token shall be returned to the secure CDB processor; and
3) The security token shall be maintained in association with the specified I_T nexus until one of the events described in this subclause causes it to be discarded.

5.13.6.8.11 CbCS shared keys

{{All of 5.13.6.8.11 is new. Editing markups suspended.}}

5.13.6.8.11.1 Overview

Cryptographic integrity checking for CbCS capabilities depends on the following hierarchy of shared keys (see 3.1.147) that is specific to the CbCS model:

1) A master key that is composed of:
   A) An authentication key; and
   B) A generation key;
   and
2) Up to 16 working keys whose values are generated from the generation key component of the master key.

Each CbCS shared key set shall support:

a) One master key; and
b) At least two working keys.

Each CbCS shared key in a set has an associated identifier (see 5.13.6.8.11.2) that describes the CbCS shared key to the objects that are managing it without revealing the CbCS shared key's value.

Coordinated sets of CbCS shared keys that conform to this hierarchy are maintained by:

a) The CbCS management application client (see 5.13.6.8.4);
b) The CbCS management device server (see 5.13.6.8.3); and
c) The enforcement manager (see 5.13.6.8.7).

The mechanism for coordinating CbCS shared key sets between the CbCS management application client and the CbCS management device server is outside the scope of this standard.

The following security protocols are provided by this standard for coordinating CbCS shared key sets between the CbCS management application client and the enforcement manager:

a) A Diffie-Hellman key exchange protocol for changing all the components of the master key (see 5.13.6.8.11.4); and
b) Mechanisms that invalidate a working key or set a working key based on the generation key component of the current master key (see 5.13.6.8.11.5).

These key management protocols associate a key identifier with each shared key (i.e., master key or working key). The CbCS management application client may use these key identifiers to describe the shared key in some way (e.g., when the shared key was last refreshed or how the intended use of the shared key). Key identifiers shall not be used to contain shared key values.

Within any SCSI target device that contains an enforcement manager, sets of CbCS shared keys are maintained as follows:

a) A separate set of per-logical unit CbCS shared keys for a logical unit that has CbCS enabled;

b) One target-wide set of CbCS shared keys that are accessible to all logical units that have CbCS enabled; or
c) Both a target-wide set of CbCS shared keys and per-logical unit sets of CbCS shared keys.

The enforcement manager in any logical unit that has CbCS enabled shall have access to at least one set of CbCS shared keys.

If an enforcement manager has access to both a target-wide set of CbCS shared keys and a per-logical unit set of CbCS shared keys, then a working key defined in the per-logical unit set of CbCS shared keys, if any, shall be used instead of the equivalent working key from the target-wide set of CbCS shared keys.

All CbCS shared keys should be retired from active use (i.e., set, changed, or discarded) often enough to thwart key attacks. How often to retire CbCS shared keys from active use is outside the scope of this standard.

The shared keys in a CbCS shared key set and the CbCS pages used to manage them between the CbCS management application client and the enforcement manager are summarized in table x3.

Table x3 — Summary of CbCS shared keys

CbCS shared key	Applicable CbCS page	Data direction	Capability protection	Reference
Master Authentication, and Generation	Current CbCS Parameters [a]	In	Working key	7.6.c.3.5
	Set Master Key – Seed Exchange Set Master Key – Seed Exchange Set Master Key – Change Master Key [b]	Out In Out	Authentication key	7.6.c.5.5 7.6.c.3.6 7.6.c.5.6
Working key	Current CbCS Parameters [a]	In	Working key	7.6.c.3.5
	Invalidate Key Set Key	Out	Authentication key	7.6.c.5.3 7.6.c.5.4

[a] Only key identifiers are returned, not shared key values.
[b] The new authentication key computed during the seed exchange is the authentication key used for this CbCS page.

5.13.6.8.11.2 CbCS shared key identifiers

CbCS shared key identifiers (see table x4) are set by the CbCS pages (see table x3 in 5.13.6.8.11.1) that change the values of a CbCS shared key and reported by the Current CbCS Parameters CbCS page (see 7.6.c.3.5).

Table x4 — CbCS shared key identifier values

Value	Description
0000 0000 0000 0000h [a]	The associated CbCS shared key has not been modified since the SCSI target device was manufactured
0000 0000 0000 0001h - FFFF FFFF FFFF FFFDh	The associated CbCS shared key has a valid value that has been set by the applicable CbCS page
FFFF FFFF FFFF FFFEh [a]	The associated CbCS shared key does not have a valid value
FFFF FFFF FFFF FFFFh [a]	The associated CbCS shared key is not supported

[a] The use of this value in the CbCS pages that change CbCS shared key values is reserved.

5.13.6.8.11.3 Specifying which CbCS shared key to change

The logical unit to which a SECURITY PROTOCOL IN command (see 6.30) or a SECURITY PROTOCOL OUT command (see 6.31) that specifies one of the CbCS pages shown in table x3 (see 5.13.6.8.11.1) is addressed determines which CbCS shared key is modified as follows:

a) If the command is addressed to the SECURITY PROTOCOL well known logical unit (see 8.5), then the CbCS the target-wide (see 5.13.6.8.11.1) set of CbCS shared keys is modified; or
b) If the command is addressed to a logical unit that is not a well known logical unit, then the per-logical unit (see 5.13.6.8.11.1) set of CbCS shared keys for the specified logical unit are modified.

5.13.6.8.11.4 Updating a CbCS master key

Both components of the CbCS master key (i.e., the authentication key and the generation key) are changed using a single Diffie-Hellman exchange CCS (see 3.1.24) as summarized in this subclause. Use of the Diffie-Hellman exchange ensures forward secrecy of the master key.

The CbCS master key update CCS is composed of the following commands:

1) A SECURITY PROTOCOL OUT command processing the Set Master Key – Seed Exchange CbCS page (see 7.6.c.5.5);
2) A SECURITY PROTOCOL IN command processing the Set Master Key – Seed Exchange CbCS page (see 7.6.c.3.6); and
3) A SECURITY PROTOCOL OUT command processing the Set Master Key – Change Master Key CbCS page (see 7.6.c.5.6).

NOTE x1 - The capability key used to access the two Set Master Key – Seed Exchange CbCS pages is different from the capability key used to access the Set Master Key – Change Master Key CbCS page (see 5.13.6.8.12.3).

The device server shall maintain CCS state for only one CbCS master key update CCS for all I_T nexuses. The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to COMMAND SEQUENCE ERROR, if any of the following conditions occur:

a) A command attempts to start a new CbCS master key update CCS while state is being maintained for another; or
b) A sequence of CbCS master key update CCS commands other than the one shown in this subclause is attempted.

The device server shall discard the CbCS master key update CCS state if any of the following occur:

a) A command in the CCS does not complete with GOOD status; or
b) The entire CbCS master key update CCS command sequence shown in this subclause is not completed within ten seconds of the successful completion of processing for the SECURITY PROTOCOL OUT command for the Set Master Key – Seed Exchange CbCS page.

5.13.6.8.11.5 Changing a CbCS working keys

A working key is invalidated using the Invalidate Key CbCS page (see 7.6.c.5.3) and set to a new value using the Set Key CbCS page (see 7.6.c.5.4).

New working keys are computed based on the applicable master key and a random number seed.

Working keys are tracked using CbCS shared key identifiers (see 5.13.6.8.11.2).

5.13.6.8.12 CbCS credentials

{{All of 5.13.6.8.12 is new. Editing markups suspended.}}

5.13.6.8.12.1 Overview

Each CbCS credential authorizes access to:

a) A logical unit (see 3.1.71); or
b) A specific volume (see SSC-3) mounted in a specific logical unit.

The applicability of CbCS credentials to the BASIC CbCS method (see 5.13.6.8.8.2) is outside the scope of this standard.

The format of a CbCS credential is described in 6.r.2.2.

The primary components of a CbCS credential are as follows:

a) A CbCS capability whose format and preparation are described in 5.13.6.8.13; and
b) The CbCS capability key that is an integrity check value (see 3.1.64) that is computed as follows:
   A) If the credential is to be sent to a secure CDB originator (see 5.13.6.8.5), the CbCS capability key is computed based on a working key as described in 5.13.6.8.12.2; or
   B) If the credential is being prepared for use by the CbCS management application client (see 5.13.6.8.4), the CbCS capability key is computed based on a working key or the master key as described in 5.13.6.8.12.3.

Regardless of how it is computed, the CbCS capability key is used as follows:

a) By the secure CDB originator to prepare CbCS extension descriptors (see 5.13.6.8.16) sent to the secure CDB processor (see 5.13.6.8.6);
b) By the CbCS management application client to prepare CbCS extension descriptors sent to the enforcement manager (see 5.13.6.8.7); and
c) By the enforcement manager to validate the integrity of capabilities (see 5.13.6.8.13.3) in CbCS extension descriptors received by the secure CDB processor.

5.13.6.8.12.2 CbCS capability key computations for the secure CDB originator

For credentials sent to the secure CDB originator (see 5.13.6.8.5), the CbCS capability key (see 5.13.6.8.12.1) is computed without knowledge of the command for which it is being prepared using the following inputs:

a) The integrity check value algorithm specified by the INTEGRITY CHECK VALUE ALGORITHM field (see 6.r.2.3) in the CbCS capability descriptor; and
b) The following inputs to this integrity check value algorithm:
   A) All the bytes in the CbCS capability descriptor (see 6.r.2.3); and
   B) The working key specified by the KEY VERSION field (see 6.r.2.3) in the CbCS capability descriptor.

5.13.6.8.12.3 CbCS capability key computations for general use

The computation of the CbCS capability key depends on the command for which the CbCS capability key is being computed as described in this subclause. This computation is more general than the computation described in 5.13.6.8.12.2, but it produces the same results because the secure CDB originator should not be allowed to use the commands that produce the exceptional cases described in this subclause.

The CbCS capability key computations described in this subclause are used:

a) For credentials that are to be used by the CbCS management application client (see 5.13.6.8.4); and
b) By the enforcement manager (see 5.13.6.8.7) when validating a CbCS capability descriptor (see 5.13.6.8.13).

When the enforcement manager is validating a CbCS capability descriptor, the command is determined by inspecting the CDB that is being processed.

Based on the command associated with the credential, the CbCS capability key (see 5.13.6.8.12.1) is computed using the following inputs:

a) The integrity check value algorithm specified by the INTEGRITY CHECK VALUE ALGORITHM field (see 6.r.2.3) in the CbCS capability descriptor; and
b) The following inputs to this integrity check value algorithm:
   A) All the bytes in the CbCS capability descriptor (see 6.r.2.3); and
   B) The following CbCS shared key:
      a) If the command is a SECURITY PROTOCOL IN command (see 6.30) or a SECURITY PROTOCOL OUT command (see 6.30) with the SECURITY PROTOCOL field set to 07h (i.e., CbCS) and the SECURITY PROTOCOL SPECIFIC field set to a value that is greater than CFFFh, then the following shared key is used:
         A) If the command does not access the Set Master Key - Change Key CbCS page (see 7.6.c.5.6), the authentication key component of the master key (see 5.13.6.8.11) is used; or
         B) If the command accesses the Set Master Key - Change Key CbCS page, then the authentication key component of the new master key (see 7.6.c.3.6) maintained in the CbCS master key update CCS (see 5.13.6.8.11.4) is used;

      or

      b) If the command is not one of those described in step b) B) a), then the working key specified by the KEY VERSION field (see 6.r.2.3) in the CbCS capability descriptor.

5.13.6.8.13 CbCS capability descriptors

{{All of 5.13.6.8.13 is new. Editing markups suspended.}}

5.13.6.8.13.1 Overview

CbCS capability descriptors are components of:

a) CbCS credentials (see 5.13.6.8.12); and
b) CbCS extension descriptors (see 5.13.6.8.16).

The format of a CbCS capability descriptor is described in 6.r.2.3.

CbCS capability descriptors contain:

a) Information about what commands are allowed when the CbCS capability descriptor is associated with a specific CDB via a CbCS extension descriptor;
b) Information that identifies a specific logical unit (see 3.1.71) or a specific volume (see SSC-3) mounted in a specific logical unit to which the CbCS capability is bound;
c) An optional time limit on the validity of the CbCS capability descriptor;
d) Information that the CbCS management application client (see 5.13.6.8.4) may use to invalidate one or more CbCS capability descriptors before the time limit expires; and
e) Information that the enforcement manager uses to cryptographically validate the CbCS capability descriptor, if specified, as described in 5.13.6.8.13.

5.13.6.8.13.2 CbCS extension descriptor validation

The enforcement manager (see 5.13.6.8.7) shall validate the CbCS capability descriptor (see 6.r.2.3) included in the CbCS extension descriptor (see 5.13.6.8.16). If the validation fails, the enforcement manager shall interact with the secure CDB processor (see 5.13.6.8.6) in a way that causes the command containing the CbCS extension descriptor to be terminated with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

The enforcement manager's validation of a CbCS extension descriptor shall fail if any of the following conditions occur:

{{The following list has been modified to be an ordered list in r2. The intent of this change is to force CAPKEY method integrity checking to be performed as early as possible. 07-454r5 required the integrity checking to be performed before other validation tests. Change bars have not been added to mark letter-to-number changes.}}

1) A CbCS extension descriptor is not present on a command that table x5 (see 5.13.6.8.14) shows as requiring a CbCS capability;
2) The command is one that table x5 (see 5.13.6.8.14) shows as never being allowed when CbCS is enabled;
3) The CBCS METHOD field is set to a value that is less than the value in the minimum CbCS method CbCS parameter (see 5.13.6.8.15);
4) The CBCS METHOD field contains a value that table x18 defines as reserved (see 6.r.2.3) or a value that the enforcement manager does not support (see 7.6.c.3.3);
5) If the CBCS METHOD field is set to CAPKEY and the integrity validation described in 5.13.6.8.13.3 fails;
6) The DESIGNATION TYPE field contains a value that table x17 defines as reserved (see 6.r.2.3);
7) The DESIGNATION TYPE field value is set to 1h (i.e., logical unit designation descriptor), and the contents of the DESIGNATION DESCRIPTOR field in which a logical unit name (see SAM-4) is indicated does not match the addressed logical unit;
8) The DESIGNATION TYPE field value is set to 2h (MAM attribute descriptor) and either of the following are true:
   A) The ATTRIBUTE IDENTIFIER field in the DESIGNATION DESCRIPTOR field contains any value other than 0401h (i.e., MEDIUM SERIAL NUMBER); or
   B) The DESIGNATION DESCRIPTOR field contents do not match the MAM attribute of the volume that is accessible via the addressed logical unit;
9) The CAPABILITY EXPIRATION TIME field contains a non-zero value and the value in the CAPABILITY EXPIRATION TIME field is less than (i.e., prior to) the current time in the clock CbCS parameter (see 5.13.6.8.15);
10) The POLICY ACCESS TAG field contains a non-zero value that does not match the policy access tag CbCS parameter (see 5.13.6.8.15); or
11) The command in the CDB field of the extended CDB (see 4.3.4) that contains the CbCS extension descriptor is not permitted by the PERMISSIONS BIT MASK field (see 5.13.6.8.14).

5.13.6.8.13.3 CAPKEY CbCS method capability integrity validation

If the CbCS method is CAPKEY, the enforcement manager's validation of a CbCS capability descriptor shall fail the integrity tests if any of the described in this subclause occur.

Before attempting to cryptographically validate the integrity of the CbCS capability descriptor, the enforcement manager shall fail the validation if any of the following conditions occur in the CbCS capability descriptor (see 6.r.2.3):

a) The KEY VERSION field specifies an invalid working key as follows:
   A) The command is not a SECURITY PROTOCOL IN command (see 6.30) or a SECURITY PROTOCOL OUT command (see 6.30) with the SECURITY PROTOCOL field set to 07h (i.e., CbCS) and the SECURITY PROTOCOL SPECIFIC field set to a value that is greater than CFFFFh (i.e., the KEY VERSION field is ignored for these commands);

B) The per-logical unit working key (see 5.13.6.8.11), if any, specified by the KEY VERSION field is invalid (see 5.13.6.8.11.2); and
C) The target-wide working key (see 5.13.6.8.11), if any, specified by the KEY VERSION field is invalid (see 5.13.6.8.11.2);

or

b) The INTEGRITY CHECK VALUE ALGORITHM field contains a value that is:
A) Not one of those that table 71 (see 5.13.8) lists as being an integrity checking (i.e., AUTH) algorithm;
B) Is AUTH_COMBINED; or
C) Is a value that the enforcement manager does not support (see 7.6.c.3.3).

If no integrity checking configuration errors are found in the CbCS capability descriptor, the enforcement manager shall:

1) Compute the CbCS capability key for the CbCS capability descriptor as described in 5.13.6.8.12.3; and
2) Compute the expected contents of CbCS extension descriptor INTEGRITY CHECK VALUE field (see 5.13.6.8.16), using the following inputs:
A) The integrity check value algorithm specified by the INTEGRITY CHECK VALUE ALGORITHM field (see 6.r.2.3) in the CbCS capability descriptor; and
B) The following inputs to this integrity check value algorithm:
a) All the bytes in the security token (see 5.13.6.8.10) for the I_T nexus on which the command was received as the string for which the integrity check value is to be computed; and
b) The CbCS capability key computed in step 1) as the cryptographic key.

The enforcement manager shall fail the validation if the contents of CbCS extension descriptor INTEGRITY CHECK VALUE field do not match the computed expected contents of CbCS extension descriptor INTEGRITY CHECK VALUE field.

5.13.6.8.14 Association between commands and permission bits

{{All of 5.13.6.8.14 is new. Editing markups suspended.}}

The PERMISSIONS BIT MASK field in the CbCS capability (see 6.r.2.3) specifies which commands are allowed by the CbCS capability. When processing commands with the CbCS extension, the enforcement manager shall verify that the bits applicable to the encapsulated SCSI command are all set to one in the PERMISSIONS BIT MASK field before

processing the command. The associations between commands and permission bits are specified in table x5 and table x6 for commands defined in this standard.

Table x5 — Associations between commands and permissions (part 1 of 2)

Command	PERMISSIONS BIT MASK bits [a]						
	DATA READ	DATA WRITE	PARM READ	PARM WRITE	SEC MGMT	RESRV	MGMT
ACCESS CONTROL IN	never allow [b]						
ACCESS CONTROL OUT	never allow [b]						
CHANGE ALIASES	always allow [c]						
EXTENDED COPY	never allow [b]						
INQUIRY	always allow [c]						
LOG SELECT				1			
LOG SENSE			1				
MANAGEMENT PROTOCOL IN							1
MANAGEMENT PROTOCOL OUT							1
MODE SELECT(6)				1			
MODE SELECT(10)				1			
MODE SENSE(6)			1				
MODE SENSE(10)			1				
PERSISTENT RESERVE IN			1				
PERSISTENT RESERVE OUT						1	
READ ATTRIBUTE			1				
READ BUFFER					1		
READ MEDIA SERIAL NUMBER			1				
RECEIVE COPY RESULTS	never allow [b]						
RECEIVE CREDENTIAL	always allow [c]						
RECEIVE DIAGNOSTIC RESULTS			1				
REPORT ALIASES	always allow [c]						
REPORT IDENTIFYING INFORMATION			1				
REPORT LUNS	always allow [c]						
REPORT PRIORITY			1				

[a] The command in the CDB field of the extended CDB (see 4.3.4) that contains the CbCS extension descriptor shall be allowed only when all of the bits marked with a 1 in the row for that command are set in the PERMISSIONS BIT MASK field of the CbCS capability in the CbCS extension descriptor. The permissions bits represented by the empty cells in a row are ignored.

[b] If the CBCS bit is set to one in the Extended INQUIRY Data VPD page (see 7.7.4), this command shall never be allowed.

[c] This command shall always be allowed regardless of whether the CbCS extension descriptor is present and if the CbCS extension descriptor is present regardless of the value in the PERMISSIONS BIT MASK field.

Table x5 — Associations between commands and permissions (part 2 of 2)

Command	PERMISSIONS BIT MASK bits [a]						
	DATA READ	DATA WRITE	PARM READ	PARM WRITE	SEC MGMT	RESRV	MGMT
REPORT SUPPORTED OPERATION CODES	always allow [c]						
REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS	always allow [c]						
REPORT TARGET PORT GROUPS	always allow [c]						
REPORT TIMESTAMP			1				
REQUEST SENSE			1				
SECURITY PROTOCOL IN	see table x6						
SECURITY PROTOCOL OUT	see table x6						
SEND DIAGNOSTIC				1			
SET IDENTIFYING INFORMATION				1			
SET PRIORITY				1			
SET TARGET PORT GROUPS				1			
SET TIMESTAMP				1	1		
TEST UNIT READY	always allow [c]						
WRITE ATTRIBUTE				1			
WRITE BUFFER					1		

[a] The command in the CDB field of the extended CDB (see 4.3.4) that contains the CbCS extension descriptor shall be allowed only when all of the bits marked with a 1 in the row for that command are set in the PERMISSIONS BIT MASK field of the CbCS capability in the CbCS extension descriptor. The permissions bits represented by the empty cells in a row are ignored.

[b] If the CBCS bit is set to one in the Extended INQUIRY Data VPD page (see 7.7.4), this command shall never be allowed.

[c] This command shall always be allowed regardless of whether the CbCS extension descriptor is present and if the CbCS extension descriptor is present regardless of the value in the PERMISSIONS BIT MASK field.

The usage of the PERMISSIONS BIT MASK field for the SECURITY PROTOCOL IN command and the SECURITY PROTOCOL OUT command depend on the following characteristics as shown in table x6:

a) The device server that is processing the command;
b) The contents of the SECURITY PROTOCOL field; and
c) The contents of the SECURITY PROTOCOL SPECIFIC field.

Table x6 — Associations between security protocol commands and permissions

Command	SECURITY PROTOCOL field	SECURITY PROTOCOL SPECIFIC field	Description
SECURITY PROTOCOL IN	00h	any	Always allowed regardless of whether the CbCS extension descriptor is present and if the CbCS extension descriptor is present regardless of the value in the PERMISSIONS BIT MASK field
SECURITY PROTOCOL IN	07h	0000h - 003Fh	
SECURITY PROTOCOL IN	07h	0040h - FFFFh	Allowed only if the CbCS extension descriptor is present and the SEC MGMT bit is set to one in the PERMISSIONS BIT MASK field
SECURITY PROTOCOL OUT	any	any	

Command standards (see 3.1.18) may describe the associations between commands and permission bits for the commands that they define. The processing requirements for those associations are the same as those described in this standard.

{{The paragraph immediately above is much less wordy than 07-454r5, but it should be saying the same thing. Reviewers should verify this.}}

5.13.6.8.15 CbCS parameters

{{All of 5.13.6.8.15 is new. Editing markups suspended.}}

5.13.6.8.15.1 Overview

CbCS parameters:

a) Provide the CbCS Management Application Client class (see 5.13.6.8.4) with a means to control the operation of the Enforcement Manager class(see 5.13.6.8.7);
b) Allow the Secure CDB Processor class(see 5.13.6.8.6) to receive security tokens and other CbCS information from the Enforcement Manager class; and
c) Allow any application client to receive basic operational CbCS information from the Enforcement Manager class.

CbCS parameters that are not changeable indicate which CbCS features and algorithms are supported. An application client may retrieve the unchangeable CbCS parameters by using the SECURITY PROTOCOL IN command to return the Unchangeable CbCS Parameters CbCS page (see 7.6.c.3.3).

The CbCS parameters with values that change in response to various conditions are summarized in table x7.

Table x7 — Summary of changeable CbCS parameters

Parameter	Support	Applicable CbCS page	Data direction	Capability protection	Reference
CbCS parameters that are updated automatically based on I_T nexus					
Security token	Optional [a]	Security Token	In	None	7.6.c.3.4
CbCS parameters that provide initial values for dynamically created logical units [b]					
Initial minimum CbCS method	Optional	Current CbCS Parameters	In	Working key	7.6.c.3.5
		Set Minimum CbCS Method [c]	Out	Working key	7.6.c.5.2
Initial policy access tag	Optional	Current CbCS Parameters	In	Working key	7.6.c.3.5
		Set Policy Access Tag [d]	Out	Working key	7.6.c.5.1
CbCS parameters that affect the CbCS enforcement manager processing					
Minimum CbCS method	Mandatory	Current CbCS Parameters	In	Working key	7.6.c.3.5
		Set Minimum CbCS Method [c]	Out	Working key	7.6.c.5.2
Policy Access Tag	Mandatory	Current CbCS Parameters	In	Working key	7.6.c.3.5
		Set Policy Access Tag [d]	Out	Working key	7.6.c.5.1
Clock	Mandatory	Current CbCS Parameters	In	Working key	7.6.c.3.5
CbCS Shared keys and CbCS shared key identifiers	Optional [a]	see 5.13.6.8.11			

[a] Mandatory if the CAPKEY CbCS method (see 5.13.6.8.8.3) is supported.

[b] SCSI target devices that do not dynamically create logical units may not implement these CbCS parameters. Retrieving and setting these CbCS parameters is possible only if the SECURITY PROTOCOL well known logical unit (see 8.5) is implemented. SCSI target devices that dynamically create logical units but do not implement the SECURITY PROTOCOL well known logical unit shall provide a means outside the scope of this standard for managing the values of these CbCS parameters.

[c] If a Set Minimum CbCS Method CbCS page is processed by the SECURITY PROTOOCL well known logical unit, then the initial Minimum CbCS Method CbCS parameter is changed. If a Set Minimum CbCS Method CbCS page is processed by any logical unit other than the SECURITY PROTOOCL well known logical unit, then the minimum CbCS method CbCS parameter for that logical unit is changed.

[d] If a Set Policy Access Tag CbCS page is processed by the SECURITY PROTOOCL well known logical unit, then the initial policy access tag CbCS parameter is changed. If a Set Policy Access Tag CbCS page is processed by any logical unit other than the SECURITY PROTOOCL well known logical unit, then the policy access tag CbCS parameter for that logical unit is changed.

The security token CbCS parameter is described in 5.13.6.8.10.

The minimum CbCS method parameter indicates the minimum allowable CbCS method (see 5.13.6.8.8) to be used in capabilities (see 6.r.2.3) processed by an enforcement manager (see 5.13.6.8.7). The initial minimum CbCS method CbCS parameter provides an initial value for the minimum CbCS method CbCS parameter for dynamically created logical units.

The policy access tag parameter indicates the allowable contents of the POLICY ACCESS TAG field in capabilities (see 6.r.2.3) processed by an enforcement manager (see 5.13.6.8.7). The initial policy access tag CbCS parameter provides an initial value for the policy access tag CbCS parameter for dynamically created logical units.

The clock CbCS parameter indicates the time used by the enforcement manager (see 5.13.6.8.7) when evaluating the contents of the CAPABILITY EXPIRATION TIME field in a capability (see 6.r.2.3). The clock CbCS parameter is timestamp (see 5.12) and its value is managed using the same mechanisms that are used to manage a timestamp.

The CbCS shared keys and CbCS shared key identifiers are described in 5.13.6.8.11.

5.13.6.8.16 CbCS extension descriptor format

{{All of 5.13.6.8.16 is new. Editing markups suspended.}}

The CbCS extension descriptor (see table x8) allows the capability-based command security technique (see 5.13.6.8) to used with a SCSI command via the parameters specified in this subclause. Support for the CbCS extension descriptor is mandatory if CBCS bit to one in the Extended INQUIRY Data VPD page (see 7.7.4). When an extended CDB (see 4.3.4) includes a CbCS extension descriptor the CDB field may contain any CDB defined in this standard or any SCSI command standard (see 3.1.18).

Table x8 — CbCS extension descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	EXTENSION TYPE (40h)							
1	Reserved							
3								
4	CbCS capability descriptor (see 6.r.2.3)							
75								
76	(MSB)							
139	INTEGRITY CHECK VALUE							(LSB)

The EXTENSION TYPE field contains 40h (i.e., CbCS extension descriptor).

The CbCS capability descriptor is defined in 6.r.2.3.

The contents of INTEGRITY CHECK VALUE field depend on the contents of the CBCS METHOD field in the CbCS capability descriptor as follows:

a) If the CBCS METHOD field does not contain CAPKEY or a vendor specific value (see table x18 in 6.r.2.3), then the INTEGRITY CHECK VALUE field is reserved;
b) If the CBCS METHOD field contains a vendor specific value, then the contents of the INTEGRITY CHECK VALUE field are vendor specific; or
c) If the CBCS METHOD field contains CAPKEY, then the INTEGRITY CHECK VALUE field contains an integrity check value (see 3.1.65) that is computed using the integrity check value algorithm specified by the INTEGRITY CHECK VALUE ALGORITHM field in the CbCS capability descriptor (see 6.r.2.3) and the following inputs to the integrity check value algorithm:
    A) All the bytes in the security token (see 5.13.6.8.10) for the I_T nexus on which the command is being sent as the string for which the integrity check value is to be computed; and
    B) The CbCS capability key from the credential (see 5.13.6.8.12) in which the CbCS capability descriptor was received by the secure CDB originator as the cryptographic key.

The enforcement manager shall validate the CbCS capability descriptor and the INTEGRITY CHECK VALUE field as described in 5.13.6.8.13.2.

…

6.r RECEIVE CREDENTIAL command

{{All of 6.r is new. Editing markups suspended.}}

6.r.1 RECEIVE CREDENTIAL command description

6.r.1.1 Overview

The RECEIVE CREDENTIAL command (see table x9) allows a secure CDB originator (see 5.13.6.2) to receive a credential from a security manager device server (e.g., a CbCS management device server (see 5.13.6.8.3)) for use in a CDB (e.g., use in the CbCS extension descriptor (see 5.13.6.8.16)).

Table x9 — RECEIVE CREDENTIAL command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (7Fh)							
1	CONTROL							
2	Reserved							
6								
7	ADDITIONAL CDB LENGTH (n-7)							
8	(MSB)			SERVICE ACTION (1800h)				
9								(LSB)
10	(MSB)			ALLOCATION LENGTH				
11								(LSB)
12	(MSB)			AC_SAI				
15								(LSB)
16	ENCRYPTED REQUEST DESCRIPTOR							
n								

The ALLOCATION LENGTH field is defined in 4.3.5.6.

The AC_SAI field contains the value of the AC_SAI SA parameter (see 5.13.2.2) for the SA to be used to encrypt the parameter data as described in 6.r.2.1.

The encrypted request descriptor field shall contain an ESP-SCSI CDB descriptor (see 5.13.7.4). Before encryption and after decryption, the UNENCRYPTED BYTES field (see 5.13.7.3) that are used to compute the ENCRYPTED OR AUTHENTICATED DATA field (see 5.13.7.5) contents shall have the format shown in table x10.

Table x10 — RECEIVE CREDENTIAL command unencrypted bytes format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			CREDENTIAL REQUEST TYPE				
1								(LSB)
2				CREDENTIAL REQUEST DESCRIPTOR				
n								

The CREDENTIAL REQUEST TYPE field (see table x11) specifies type of credential being requested and the format of the CREDENTIAL REQUEST DESCRIPTOR field.

Table x11 — CREDENTIAL REQUEST TYPE field

Code	Description	Reference
0001h	CbCS logical unit	6.r.1.2
0002h	CbCS logical unit and volume	6.r.1.3
all other codes	Reserved	

{{A row in this table might define a range of codes as restricted to OSD, but first the interests of the SNIA OSD TWG must be assessed.}}

The CREDENTIAL REQUEST DESCRIPTOR field specifies the information needed to request the credential as described in table x11.

If return of the requested credential is not permitted, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to ACCESS DENIED - NO ACCESS RIGHTS.

The DS_SAI field in the ENCRYPTED REQUEST DESCRIPTOR field contains the value of the DS_SAI SA parameter (see 5.13.2.2) for the SA to be used to encrypt the unencrypted bytes and the parameter data as described.

If the device server is not maintaining an SA with an AC_SAI SA parameter that matches the AC_SAI field contents and a DS_SAI SA parameter that matches the DS_SAI field contents, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

If the device server is maintaining the SA specified by the AC_SAI field and the DS_SAI field, then the SA shall be verified for use by this RECEIVE CREDENTIAL command as follows:

a) The USAGE_TYPE SA parameter (see 5.13.2.2) shall be verified to be equal to 82h (i.e., CbCS authentication and credential encryption; and
b) The USAGE_DATA SA parameter (see 5.13.2.2) shall be verified not to contain an ALGORITHM IDENTIFIER field (see 7.6.3.6) that is set to ENCR_NULL based on the contents the IKEv2-SCSI SAUT Cryptographic Algorithm payload (see 7.6.3.15.13) for the ENCR algorithm type (see 7.6.3.6.2) during creation of the SA (see 5.13.2.3).

If any of these SA verifications fails, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.r.1.2 CbCS logical unit credential request descriptor

If the credential request type field is set to 0001h (i.e., CbCS logical unit), then the format of the CREDENTIAL REQUEST DESCRIPTOR field is as shown in table x12.

Table x12 — CbCS logical unit credential request descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESIGNATION DESCRIPTOR							
19								

The format of the DESIGNATION DESCRIPTOR field is defined in table 422 (see 7.7.3.1). The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB if any of the fields in the DESIGNATION DESCRIPTOR field are set as follows:

a) The DESIGNATOR TYPE field contains any value other than 3h (i.e., NAA);
b) The ASSOCIATION field contains any value other than 00b (i.e., logical unit) or 10b (i.e., SCSI target device); or
c) The DESIGNATOR LENGTH field is set to a value that is larger than 16.

6.r.1.3 CbCS logical unit and volume credential request descriptor

If the credential request type field is set to 0002h (i.e., CbCS logical unit and volume), then the format of the CREDENTIAL REQUEST DESCRIPTOR field is as shown in table x13.

Table x13 — CbCS logical unit and volume credential request descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESIGNATION DESCRIPTOR							
19								
20	MAM ATTRIBUTE							
56								

The format of the DESIGNATION DESCRIPTOR field is defined in table 422 (see 7.7.3.1). The command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB if any of the fields in the DESIGNATION DESCRIPTOR field are set as follows:

a) The DESIGNATOR TYPE field contains any value other than 3h (i.e., NAA);
b) The ASSOCIATION field contains any value other than 00b (i.e., logical unit) or 10b (i.e., SCSI target device); or
c) The DESIGNATOR LENGTH field is set to a value that is larger than 16.

The format of the MAM ATTRIBUTE field is defined in table 310 (see 7.3.1). If the ATTRIBUTE IDENTIFIER field in the MAM ATTRIBUTE field contains any value other than 0401h (i.e., MEDIUM SERIAL NUMBER), the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

6.r.2 RECEIVE CREDENTIAL parameter data

6.r.2.1 RECEIVE CREDENTIAL parameter data encryption

The RECEIVE CREDENTIAL parameter data shall be one of the ESP-SCSI data-in buffer descriptors shown in table 74 (see 5.13.7.5.1). The SA specified by the AC_SAI field and the DS_SAI field in the CDB shall be used to construct the ESP-SCSI data-in buffer descriptor as described in 5.13.7.5.

Before processing the parameter data, the application client should validate and decrypt the ESP-SCSI data-in buffer descriptor as described in 5.13.7.5. If any errors are detected by the validation and decryption processing, the parameter data should be ignored.

6.r.2.2 RECEIVE CREDENTIAL decrypted parameter data

Before encryption and after decryption, the UNENCRYPTED BYTES field (see 5.13.7.3) that are used to compute the ENCRYPTED OR AUTHENTICATED DATA field (see 5.13.7.5) contents shall contain a CbCS credential (see table x14).

Table x14 — CbCS credential format

Bit / Byte	7	6	5	4	3	2	1	0
0	Reserved				CREDENTIAL FORMAT (1h)			
1	Reserved							
2	(MSB)			CREDENTIAL LENGTH (n-3)				
3								(LSB)
4	(MSB)			CAPABILITY LENGTH (k-5)				
5								(LSB)
6				CbCS capability descriptor (see 6.r.2.3)				
k								
k+1	(MSB)			CAPABILITY KEY LENGTH (n-k-4))				
k+4								(LSB)
k+5				CAPABILITY KEY				
n								

{{The CR PRSNT bit defined in 07-454r5 has been removed in this proposal because it is redundant with the length information in the ESP-SCSI descriptor.}}

The CREDENTIAL FORMAT field (see table x15) indicates the format of the credential.

Table x15 — Credential format values

Value	Description
0h	Reserved
1h	The format defined by this standard
2h - Fh	Reserved

{{Because the capability format is not in the capability descriptor, the information it contains is not available in the CbCS extension descriptor. This requires the enforcement manager to use the XCDB extension descriptor EXTENSION TYPE field as the indicator of capability format.}}

The CREDENTIAL LENGTH field indicates the number of bytes that follow in the credential including the capability length, the CbCS capability descriptor, the capability key length, and the capability key.

The CAPABILITY LENGTH field indicates the number of bytes that follow in the capability.

The contents of the CbCS capability descriptor are defined in 6.r.2.3.

The CAPABILITY KEY LENGTH field indicates the number of bytes that follow in the capability key.

The CAPABILITY KEY field contains an integrity check value (see 3.1.64) that is computed and used as described in 5.13.6.8.12.

6.r.2.3 CbCS capability descriptor

6.r.2.3.1 Overview

A CbCS capability descriptor (see table x16) specifies the commands that are allowed by the CbCS extension descriptor in which it appears.

Table x16 — CbCS capability descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESIGNATION TYPE				KEY VERSION			
1	CBCS METHOD							
2	(MSB)			CAPABILITY EXPIRATION TIME				
7								(LSB)
8	(MSB)			INTEGRITY CHECK VALUE ALGORITHM				
11								(LSB)
12				PERMISSIONS BIT MASK				
15								
16	(MSB)			POLICY ACCESS TAG				
19								(LSB)
20				DESIGNATION DESCRIPTOR				
57								
58				DISCRIMINATOR				
71								

The DESIGNATION TYPE field (see table x17) specifies the format of the Designation descriptor.

Table x17 — DESIGNATION TYPE field

Code	Description	DESIGNATION DESCRIPTOR field format reference
0h	Reserved	
1h	Logical unit designation descriptor	6.r.2.3.2
2h	MAM attribute designation descriptor	6.r.2.3.3
3h - Fh	Reserved	

The KEY VERSION field specifies which working key (see 5.13.6.8.11), is being used to compute the capability key (see 5.13.6.8.12) for this CbCS capability.

The CBCS METHOD field (see table x18) specifies the CbCS method used by this CbCS capability.

Table x18 — CBCS METHOD field

Code	CbCS method	Reference
00h	BASIC	5.13.6.8.8.2
01h	CAPKEY	5.13.6.8.8.3
02h - EFh	Reserved	
F0h - FEh	Vendor specific	
FFh	Reserved	

The CAPABILITY EXPIRATION TIME field specifies expiration time of this CbCS capability as the number of milliseconds that have elapsed since midnight, 1 January 1970 UT. If the CAPABILITY EXPIRATION TIME field is set to zero, this CbCS capability does not have an expiration time.

If the CAPABILITY EXPIRATION TIME field is not set to zero, then:

a) The clock maintained by the CbCS management device server (see 5.13.6.8.3) should be synchronized with the clock maintained by the enforcement manager (see 5.13.6.8.7). The method for synchronizing the clocks is outside the scope of this standard, however, the protocol should be implemented in a secure manner (e.g., it should not be possible for an adversary to set the clock in the SCSI device or in the secure CDB processor backwards to enable the reuse of expired CbCS credentials). The value in the enforcement manager's clock is available in the Current CbCS Parameters CbCS page (see 7.6.c.3.5) to assist in this synchronization;
b) The CbCS management device server should set the CAPABILITY EXPIRATION TIME field to a value that is at least an order of magnitude larger than the allowed deviation between the clocks.

The INTEGRITY CHECK VALUE ALGORITHM field specifies the algorithm used to compute the capability key and other integrity check values for this CbCS capability. The value in the INTEGRITY CHECK VALUE ALGORITHM field is selected from the codes that the Unchangeable CbCS Parameters CbCS page (see 7.6.c.3.3) lists as supported integrity check value algorithms.

The PERMISSIONS BIT MASK field (see table x19) specifies the permissions allowed by this CbCS capability. More than one permissions bit may be set. The relationship between commands and bits in the PERMISSIONS BIT MASK field is defined in for the commands defined by this standard and in the command standard (see 3.1.18) that defines commands for a specific device type.

Table x19 — PERMISSIONS BIT MASK field format

Bit / Byte	7	6	5	4	3	2	1	0
0	DATA READ	DATA WRITE	PARM READ	PARM WRITE	SEC MGMT	RESRV	MGMT	PHY ACC
1	Reserved							
2								
3	Restricted (see applicable command standard)							

A DATA READ bit set to zero indicates a command has no read permission for user data and protection information. A DATA READ bit set to one indicates a command has permission to read user data and protection information.

A DATA WRITE bit set to zero indicates a command has no write permission for user data and protection information. A DATA WRITE bit set to one indicates a command has permission to write user data and protection information.

A parameter data read (PARM READ) bit set to zero indicates a command has no parameter data read permission. A PARM READ bit set to one indicates a command has permission to read parameter data.

A parameter data write (PARM WRITE) bit set to zero indicates a command has no parameter data write permission. A PARM WRITE bit set to one indicates a command has permission to write parameter data.

A security management (SEC MGMT) bit set to zero indicates a command has no security management permission. A SEC MGMT bit set to one indicates a command has security management permission.

A reservation (RESRV) bit set to zero indicates a command has no persistent reservation permission. A RESRV bit set to one indicates a has permission to make or modify persistent reservations.

A management (MGMT) bit set to zero indicates a command has no storage management permission. A MGMT bit set to one indicates a command has storage management permission. Storage management is outside the scope of this standard.

A physical access (PHY ACC) bit set to zero indicates a command has no permission to affect physical access to the logical unit or volume. A PHY ACC bit set to one indicates a command has permission to affect physical access to the logical unit or volume (see SSC-3).

If the POLICY ACCESS TAG field contains a value other than zero, the policy access tag attribute of the logical unit (see 5.13.6.8.15) is compared to the POLICY ACCESS TAG field contents as part of validating the CbCS capability (see 5.13.6.8.13.2). If the POLICY ACCESS TAG field contains zero, then no comparison is made.

The DESIGNATION DESCRIPTOR field is used during the validation of the CbCS capability (see 5.13.6.8.13.2) to ensure that the command is being addressed to the correct logical unit or volume (see SSC-3). The format of the DESIGNATION DESCRIPTOR field is defined by the value in the DESIGNATION TYPE field as described in table x17.

If the CREDENTIAL REQUEST TYPE field in a RECEIVE CREDENTIAL command is set to 0001h (i.e., CbCS logical unit), then the DESIGNATION DESCRIPTOR field shall contain a logical unit designation descriptor that matches the DESIGNATION DESCRIPTOR field (see 6.r.1.2) in the CREDENTIAL REQUEST DESCRIPTOR field in the CDB. If the CREDENTIAL REQUEST TYPE field in a RECEIVE CREDENTIAL command is set to 0002h (i.e., CbCS logical unit and volume), then the DESIGNATION DESCRIPTOR field shall contain a MAM attribute designation descriptor that matches the MAM ATTRIBUTE field (see 6.r.1.3) in the CREDENTIAL REQUEST DESCRIPTOR field in the CDB.

The DISCRIMINATOR field provides uniqueness to the CbCS capability descriptor and may be used to limit the delegation or prevent leakage of the CbCS capability to other application clients. The CbCS management device server (see 5.13.6.8.3) shall not return the same CbCS capability descriptor to two secure CDB originators.

The enforcement manager (see 5.13.6.8.7) shall validate each CbCS capability descriptor it receives as described in 5.13.6.8.13.2.

6.r.2.3.2 Logical unit designation descriptor format

If the DESIGNATION TYPE field is set to 0001h (i.e., logical unit designation descriptor), then the format of the DESIGNATION DESCRIPTOR field is as shown in table x20.

Table x20 — Logical unit designation descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	DESIGNATION DESCRIPTOR							
19								
20	Reserved							
37								

The format of the DESIGNATION DESCRIPTOR field is defined in table 422 (see 7.7.3.1) with the following additional requirements:

a) The DESIGNATOR TYPE field shall contain 3h (i.e., NAA);
b) The ASSOCIATION field shall 00b (i.e., logical unit); and
c) The DESIGNATOR LENGTH field shall set to a value that is smaller than 17.

6.r.2.3.3 Volume designation descriptors

If the DESIGNATION TYPE field is set to 0002h (i.e., volume unit designation descriptor), then the format of the DESIGNATION DESCRIPTOR field is as shown in table x21.

Table x21 — Volume designation descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	MAM ATTRIBUTE							
36								
37	Reserved							

The format of the MAM ATTRIBUTE field is defined in table 310 (see 7.3.1) with the following additional requirements:

a) The MAM ATTRIBUTE field shall contain 0401h (i.e., MEDIUM SERIAL NUMBER); and
b) The attribute length field shall contain 0020h.

…

7.6 Security protocol parameters

…

7.6.c CbCS security protocol

{{All of 7.6.c is new. Editing markups suspended.}}

{{Changes related to the INC_512 bit are shown in 08-141r0 are replicated in 7.6.c without identifying annotations.}}

7.6.c.1 Overview

If the SECURITY PROTOCOL field in a SECURITY PROTOCOL IN command (see 6.30) is set to 07h, then the command specifies one of the CbCS pages (see 7.6.c.2) to be returned by the device sever. The information returned by a CbCS SECURITY PROTOCOL IN command indicates the CbCS operating parameters of:

a) The logical unit to which the CbCS SECURITY PROTOCOL IN command is addressed; or
b) The SCSI target device that contains the well-known logical unit to which the CbCS SECURITY PROTOCOL IN command is addressed.

If the SECURITY PROTOCOL field in a SECURITY PROTOCOL OUT command (see 6.31) is set to 07h, then the command specifies one of the CbCS pages (see 7.6.c.4) to be sent to the device sever. The instructions sent in a CbCS SECURITY PROTOCOL OUT command specify the CbCS operating parameters of:

a) The logical unit to which the CbCS SECURITY PROTOCOL OUT command is addressed; or
b) The SCSI target device that contains the well-known logical unit to which the CbCS SECURITY PROTOCOL OUT command is addressed.

7.6.c.2 CbCS SECURITY PROTOCOL IN CDB description

The CbCS SECURITY PROTOCOL IN CDB has the format defined in 6.30 with the additional requirements described in this subclause.

When the SECURITY PROTOCOL field is set to CbCS (i.e., 07h) in a SECURITY PROTOCOL IN command, the SECURITY PROTOCOL SPECIFIC field (see table x22) specifies the CbCS page to be returned in the parameter data (see 7.6.c.3). If the CBCS bit is set to one in the Extended INQUIRY Data VPD page (see 7.7.4), the CbCS SECURITY PROTOCOL IN command support requirements are shown in table x22.

Table x22 — SECURITY PROTOCOL SPECIFIC field for the CbCS SECURITY PROTOCOL IN command

Code [a]	CbCS page returned	Support	Reference
0000h	Supported CbCS SECURITY PROTOCOL IN Pages	Mandatory	7.6.c.3.1
0001h	Supported CbCS SECURITY PROTOCOL OUT Pages	Mandatory	7.6.c.3.2
0002h	Unchangeable CbCS Parameters	Mandatory	7.6.c.3.3
0003h – 003Eh	Reserved		
3Fh	Security Token	Optional [b]	7.6.c.3.4
0040h	Current CbCS Parameters	Mandatory	7.6.c.3.5
0041h – D00Fh	Reserved		
D010h	Set Master Key – Seed Exchange	Optional [b]	7.6.c.3.6
D011h – FFFFh	Reserved		

[a] If the SECURITY PROTOCOL SPECIFIC field contains a value that is less than D000h, then the working key specified by the KEY VERSION field in the CbCS capability descriptor (see 6.r.2.3) shall be used to compute the capability key (see 5.13.6.8.12). If the SECURITY PROTOCOL SPECIFIC field contains a value that is greater than or equal to D000h, then the authentication key component of the master key (see 5.13.6.8.11) shall be used to compute the capability key.

[b] Mandatory if the CAPKEY CbCS method (see 5.13.6.8.8.3) is supported.

{{Code assignments in table x22 have been modified to align them with code assignments in table x31.}}

If a CbCS SECURITY PROTOCOL IN command is received with the INC_512 bit set to one, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

7.6.c.3 CbCS SECURITY PROTOCOL IN parameter data

7.6.c.3.1 Supported CbCS SECURITY PROTOCOL IN Pages CbCS page

The Supported CbCS SECURITY PROTOCOL IN Pages CbCS page (see table x23) lists the CbCS pages that are supported for the (i.e., the values that are allowed in the SECURITY PROTOCOL SPECIFIC field in a) SECURITY PROTOCOL IN command (see 6.30).

Table x23 — Supported CbCS SECURITY PROTOCOL IN Pages CbCS page format

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)
Supported CbCS SECURITY PROTOCOL IN page list								
4	(MSB)			SUPPORTED CBCS SECURITY PROTOCOL IN PAGE [first] (0000h)				
5								(LSB)
				. . .				
n-1	(MSB)			SUPPORTED CBCS SECURITY PROTOCOL IN PAGE [last]				
n								(LSB)

The PAGE CODE field shall be set to 0000h to indicate that the Supported CbCS SECURITY PROTOCOL IN Pages CbCS page is being returned.

The page length field indicates the number of bytes that follow in the Supported CbCS SECURITY PROTOCOL IN Pages CbCS page.

Each SUPPORTED CBCS SECURITY PROTOCOL IN PAGE field indicates the page code (see table x22 in 7.6.c.2) of one CbCS page that is supported by the SECURITY PROTOCOL IN command when the SECURITY PROTOCOL field (see 6.30) is set to 07h (i.e., CbCS). The values in the SUPPORTED CBCS SECURITY PROTOCOL IN PAGE fields shall be returned in ascending order beginning with 0000h (i.e., this CbCS page).

7.6.c.3.2 Supported CbCS SECURITY PROTOCOL OUT pages CbCS page

The Supported CbCS SECURITY PROTOCOL OUT Pages CbCS page (see table x24) lists the CbCS pages that are supported for the (i.e., the values that are allowed in the SECURITY PROTOCOL SPECIFIC field in a) SECURITY PROTOCOL OUT command (see 6.31).

Table x24 — Supported CbCS SECURITY PROTOCOL OUT Pages CbCS page format

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)
Supported CbCS SECURITY PROTOCOL OUT page list								
4	(MSB)			SUPPORTED CBCS SECURITY PROTOCOL OUT PAGE [first]				
5								(LSB)
				. . .				
n-1	(MSB)			SUPPORTED CBCS SECURITY PROTOCOL OUT PAGE [last]				
n								(LSB)

The PAGE CODE field shall be set to 0001h to indicate that the Supported CbCS SECURITY PROTOCOL OUT Pages CbCS page is being returned.

The page length field indicates the number of bytes that follow in the Supported CbCS SECURITY PROTOCOL OUT Pages CbCS page.

Each SUPPORTED CBCS SECURITY PROTOCOL OUT PAGE field indicates the page code (see table x31 in 7.6.c.4) of one CbCS page that is supported by the SECURITY PROTOCOL OUT command when the SECURITY PROTOCOL field (see 6.31) is set to 07h (i.e., CbCS). The values in the SUPPORTED CBCS SECURITY PROTOCOL IN PAGE fields shall be returned in ascending order.

7.6.c.3.3 Unchangeable CbCS Parameters CbCS page

The Unchangeable CbCS Parameters CbCS page (see table x25) indicates the supported CbCS features and algorithms.

Table x25 — Unchangeable CbCS Parameters CbCS page format (part 1 of 2)

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0002h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	KEYS SUPPORT		MIN CBCS METHOD SUP		Reserved			
5	Reserved							
6	(MSB)							
7				SUPPORTED INTEGRITY CHECK VALUE ALGORITHM LIST LENGTH (c-7)				(LSB)
Supported integrity check value algorithms list								
8	(MSB)							
11				SUPPORTED INTEGRITY CHECK VALUE ALGORITHM [first]				(LSB)
				. . .				
c-3	(MSB)							
c				SUPPORTED INTEGRITY CHECK VALUE ALGORITHM [last]				(LSB)
c+1								
c+2				Reserved				
c+3	(MSB)							
c+4				SUPPORTED D-H ALGORITHM LIST LENGTH (d-c-4)				(LSB)
Supported Diffie-Hellman (D-H) algorithms list								
c+5	(MSB)							
c+8				SUPPORTED D-H ALGORITHM [first]				(LSB)
				. . .				
d-3	(MSB)							
d				SUPPORTED D-H ALGORITHM [last]				(LSB)
d+1	(MSB)							
d+2				SUPPORTED CBCS METHODS LIST LENGTH (n-d-2)				(LSB)

Table x25 — Unchangeable CbCS Parameters CbCS page format (part 2 of 2)

Bit / Byte	7	6	5	4	3	2	1	0
Supported CbCS methods list								
d+3	SUPPORTED CBCS METHOD [first]							
	⋮							
n	SUPPORTED CBCS METHOD [last]							

The PAGE CODE field shall be set to 0002h to indicate that the Unchangeable CbCS Parameters CbCS page is being returned.

The page length field indicates the number of bytes that follow in the Unchangeable CbCS Parameters CbCS page.

The KEYS SUPPORT field (see table x26) indicates the type of CbCS master keys and working keys supported.

Table x26 — KEYS SUPPORT field

Code	Description
00b	Reserved
01b	The SCSI target device supports single CbCS master key and a set of CbCS working keys (see 5.13.6.8.11) for the SCSI target device, but the logical units in the SCSI target device do not support CbCS master keys or working keys.
10b	The SCSI target device does not support CbCS master keys or working keys for the SCSI target device, but each logical unit in the SCSI target device supports a single CbCS master key and a set of CbCS working keys for that logical unit.
11b	The SCSI target device supports single CbCS master key and a set of CbCS working keys for the SCSI target device, and each logical unit in the SCSI target device supports a single CbCS master key and a set of CbCS working keys for that logical unit. Keys stored in the logical unit take precedence over keys stored in the SCSI target device (see 5.13.6.8.6).

The MIN CBCS METHOD SUP field (see table x27) indicates how the assignment of the minimum allowable CbCS method is supported.

Table x27 — MIN CBCS METHOD SUP field

Code	Description
00b	Reserved
01b	A single minimum allowed CbCS method (see 5.13.6.8.8) is assigned to all logical units in the SCSI target device, and the SECURITY PROTOCOL well known logical unit is implemented to control its value.
10b	Each logical unit in the SCSI target device is assigned its own minimum allowed CbCS method.
11b	Reserved

{{The 01b case was changed in r2 to eliminate the need for a SECURITY PROTOCOL well known logical unit based on the requirements written for the Set Minimum CbCS Method CbCS page (see 7.6.c.5.2).}}

The SUPPORTED INTEGRITY CHECK VALUE ALGORITHM LIST LENGTH field indicates the number of bytes that follow in the supported integrity check value algorithms list.

Each SUPPORTED INTEGRITY CHECK VALUE ALGORITHM field indicates one supported algorithm for computing CbCS integrity check values. The values in the SUPPORTED INTEGRITY CHECK VALUE ALGORITHM fields are selected from the codes that table 79 (see 5.13.8) lists as integrity checking (i.e., AUTH) algorithms, except for AUTH_COMBINED.

The SUPPORTED D-H ALGORITHM LIST LENGTH field indicates the number of bytes that follow in the supported Diffie-Hellman (D-H) algorithms list.

Each SUPPORTED D-H ALGORITHM field indicates one supported algorithm for constructing CbCS shared keys. The values in the SUPPORTED D-H ALGORITHM fields are selected from the codes that table 79 (see 5.13.8) lists as Diffie-Hellman algorithms with finite field D-H computations.

The SUPPORTED CBCS METHODS LIST LENGTH field indicates the number of bytes that follow in the supported CbCS methods list.

Each SUPPORTED CBCS METHODS field indicates one supported CbCS method (see 6.r.2.3). The values in the SUPPORTED CBCS METHODS fields are selected from the codes listed in table x18 (see 6.r.2.3).

7.6.c.3.4 Security Token CbCS page

The Security Token CbCS page (see table x28) indicates the value of the security token (see 5.13.6.8.10) for the I_T nexus on which the SECURITY PROTOCOL IN command was received.

Table x28 — Security Token CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (003Fh)				(LSB)
2	(MSB)							
3				PAGE LENGTH (n-3)				(LSB)
4	(MSB)							
n				SECURITY TOKEN				(LSB)

The PAGE CODE field shall be set to 003Fh to indicate that the Security Token CbCS page is being returned.

The page length field indicates the number of bytes that follow in the Security Token CbCS page.

The SECURITY TOKEN field shall contain the security token (see 5.13.6.8.10) for the I_T nexus on which the command was received.

7.6.c.3.5 Current CbCS Parameters CbCS page

The Current CbCS Parameters CbCS page (see table x29) indicates the current values for the CbCS parameters (see 5.13.6.8.15) used by the SCSI target device or logical unit as follows:

a) If the logical unit to which the SECURITY PROTOCOL IN command is addressed is the SECURITY PROTOCOL well known logical unit (see 8.5), then the contents of the Current CbCS Parameters CbCS page apply to the SCSI target device; or

b) If the logical unit to which the SECURITY PROTOCOL IN command is addressed is not the SECURITY PROTOCOL well known logical unit, then the contents of the Current CbCS Parameters CbCS page apply to the addressed logical unit.

Table x29 — Current CbCS Parameters CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0040h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (009Ah)				(LSB)
4								
6				Reserved				
7				MINIMUM ALLOWED CBCS METHOD				
8	(MSB)							
11				POLICY ACCESS TAG				(LSB)
12								
15				Reserved				
16	(MSB)							
23				MASTER KEY IDENTIFIER				(LSB)
24	(MSB)							
31				WORKING KEY 0 IDENTIFIER				(LSB)
32	(MSB)							
39				WORKING KEY 1 IDENTIFIER				(LSB)
				. . .				
144	(MSB)							
151				WORKING KEY 15 IDENTIFIER				(LSB)
152	(MSB)							
157				CLOCK				(LSB)

The PAGE CODE field shall be set to 0040h to indicate that the Current CbCS Parameters CbCS page is being returned.

The page length field indicates the number of bytes that follow in the Current CbCS Parameters CbCS page.

The contents of the MINIMUM ALLOWED CBCS METHOD field depend on the logical unit that returned the Current CbCS Parameters CbCS page as follows:

a) If the logical unit is not the SECURITY PROTCOL well known logical unit, then the MINIMUM ALLOWED CBCS METHOD field indicates the smallest value allowed in the CBCS METHOD field (see 6.r.2.3) of a capability descriptor processed by the enforcement manager (see 5.13.6.8.7) as described in 5.13.6.8.13.2 (i.e., the value of the minimum CbCS method CbCS parameter for the logical unit (see 5.13.6.8.15); or

b) If the SECURITY PROTOCOL well known logical unit is returning the Current CbCS Parameters CbCS page, then the MINIMUM ALLOWED CBCS METHOD field indicates the value that will be copied to the minimum CbCS method CbCS parameter of any dynamically created logical units (i.e., the initial minimum CbCS method CbCS parameter summarized in 5.13.6.8.15).

The value in the MINIMUM ALLOWED CBCS METHOD field is selected from those listed in table x18 (see 6.r.2.3).

The contents of the POLICY ACCESS TAG field depend on the logical unit that returned the Current CbCS Parameters CbCS page as follows:

a) If the logical unit is not the SECURITY PROTCOL well known logical unit, then the POLICY ACCESS TAG field indicates the value required in the POLICY ACCESS TAG field (see 6.r.2.3) of a capability descriptor processed by the enforcement manager (see 5.13.6.8.7) as described in 5.13.6.8.13.2 (i.e., the value of the policy access tag CbCS parameter for the logical unit (see 5.13.6.8.15); or
b) If the SECURITY PROTOCOL well known logical unit is returning the Current CbCS Parameters CbCS page, then the POLICY ACCESS TAG field indicates the value that will be copied to the policy access tag CbCS parameter of any dynamically created logical units (i.e., the initial policy access tag CbCS parameter summarized in 5.13.6.8.15).

The MASTER KEY IDENTIFIER field contains the current CbCS shared key identifier (see 5.13.6.8.11.2) for the master key (see 5.13.6.8.11).

The WORKING KEY 0 IDENTIFIER field, WORKING KEY 1 IDENTIFIER field, WORKING KEY 2 IDENTIFIER field, … and WORKING KEY 15 IDENTIFIER field contain the current CbCS shared key identifier (see 5.13.6.8.11.2) for the working keys (see 5.13.6.8.11).

The CLOCK field shall be set to the TIMESTAMP field format and value defined in 5.12.

7.6.c.3.6 Set Master Key – Seed Exchange CbCS page

The Set Master Key – Seed Exchange CbCS page (see table x30) in a SECURITY PROTOCOL IN command continues a CbCS master key update CCS (see 5.13.6.8.11.4) that was initiated by processing of a Set Master Key – Seed Exchange CbCS page in a SECURITY PROTOCOL OUT command (see 7.6.c.5.5), and completes a Diffie-Hellman key exchange protocol as part of that CCS.

Table x30 — Set Master Key – Seed Exchange CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (D010h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	(MSB)			D-H DATA				
n								(LSB)

The Diffie-Hellman (D-H) algorithm being used in the CbCS master key update CCS is determined by the contents of the D-H ALGORITHM field in the Set Master Key – Seed Exchange CbCS page in the SECURITY PROTOCOL

OUT command (see 7.6.c.5.5) that initiated the CCS. This Diffie-Hellman algorithm specifies the DH_generator value and DH_prime value to be used in the computation of the D-H DATA field as described in this subclause.

The PAGE CODE field set to D010h specifies that the Set Master Key – Seed Exchange CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Master Key – Seed Exchange CbCS page. If the PAGE LENGTH field shall be set to a value that is inconsistent with the Diffie-Hellman algorithm specified for the CbCS master key update CCS, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The contents of the D-H DATA field are computed as follows:

1) A random number, y, is generated having a value between 0 and DH_prime minus one observing the requirements in RFC 4086; and
2) The D-H DATA field is set to $DH\_generator^{y}$ modulo DH_prime, where the DH_generator and DH_prime values are identified by the value in the D-H ALGORITHM field.

As part of the successful completion of processing for the SECURITY PROTOCOL IN command transfer of the Set Master Key – Seed Exchange CbCS page the device server and application client store as part of the state maintained for CbCS master key update CCS (see 5.13.6.8.11.4):

a) The authentication key component of the next master key; and
b) The generation key component of the next master key.

{{See 07-454r5 for a proof that the $DH\_generator^{xy}$ computations shown below yield the same results.}}

The new master key is computed as follows:

1) An initial_seed value that is the value $DH\_generator^{xy}$ modulo DH_prime is computed as follows:
   A) The device server computes $(DH\_generator^{x}$ modulo DH_prime$)^{y}$, where $DH\_generator^{x}$ is the contents of the D-H DATA field in the SECURITY PROTOCOL OUT command and y is the random number generated by the device server; and
   B) The application client computes $(DH\_generator^{y}$ modulo DH_prime$)^{x}$, where $DH\_generator^{y}$ is the contents of the D-H DATA field in the SECURITY PROTOCOL IN command and x is the random number generated by the application client (see 7.6.c.5.5);
2) The generation key component of the new master key is computed using the integrity check value algorithm specified by the integrity check value algorithm field in the capability (see 6.r.2.3) in the CbCS extension descriptor (see 5.13.6.8.16) associated with the SECURITY PROTOCOL IN command that transferred the Set Master Key – Seed Exchange CbCS page. The following inputs are used with the specified integrity check value algorithm:
   A) The concatenation of the initial_seed value computed in step 1) and all of the bytes in the Device Identification VPD page (see 7.7.3) for the logical unit that is processing the CbCS master key update CCS (see 5.13.6.8.11.4) as the string for which the integrity check value is to be computed; and
   B) The generation key component of the current master key (see 5.13.6.8.11) for the logical unit that is processing the CbCS master key update CCS as the cryptographic key;
3) A modified_seed value is computed as follows:
   A) If the least significant bit of the initial_seed value is zero, then the modified_seed value is equal to the initial_seed value with the least significant bit set to one; and
   B) If the least significant bit of the initial_seed value is one, then the modified_seed value is equal to the initial_seed value with the least significant bit set to zero;

   and

4) The authentication key component of the new master key is computed using the integrity check value algorithm specified by the integrity check value algorithm field in the capability (see 6.r.2.3) in the CbCS extension descriptor (see 5.13.6.8.16) associated with the SECURITY PROTOCOL IN command that transferred the Set Master Key – Seed Exchange CbCS page. The following inputs are used with the specified integrity check value algorithm:
   A) The concatenation of the modified_seed value computed in step 3) and all of the bytes in the Device Identification VPD page for the logical unit that is processing the CbCS master key update CCS as the string for which the integrity check value is to be computed; and
   B) The generation key component of the current master key for the logical unit that is processing the CbCS master key update CCS as the cryptographic key.

7.6.c.4 CbCS SECURITY PROTOCOL OUT CDB description

The CbCS SECURITY PROTOCOL OUT CDB has the format defined in 6.31 with the additional requirements described in this subclause.

When the SECURITY PROTOCOL field is set to CbCS (i.e., 07h) in a SECURITY PROTOCOL OUT command, the SECURITY PROTOCOL SPECIFIC field (see table x31) specifies the CbCS page to be returned in the parameter data (see 7.6.c.5). If the CBCS bit is set to one in the Extended INQUIRY Data VPD page (see 7.7.4), the CbCS SECURITY PROTOCOL IN command support requirements are shown in table x31.

Table x31 — SECURITY PROTOCOL SPECIFIC field for the CbCS SECURITY PROTOCOL OUT command

Code [a]	CbCS page sent	Support	Reference
0000h – 0040h	Reserved		
0041h	Set Policy Access Tag	Optional	7.6.c.5.1
0042h	Set Minimum CbCS Method	Optional	7.6.c.5.2
0043h – CFFFh	Reserved		
D000h	Invalidate Key	Optional [b]	7.6.c.5.3
D001h	Set Key	Optional [b]	7.6.c.5.4
D003h – D00Fh	Reserved		
D010h	Set Master Key – Seed Exchange	Optional [b]	7.6.c.5.5
D011h	Set Master Key – Change Master Key	Optional [b]	7.6.c.5.6
D012h – FFFFh	Reserved		

[a] If the SECURITY PROTOCOL SPECIFIC field contains a value that is less than D000h, then the working key specified by the KEY VERSION field in the CbCS capability descriptor (see 6.r.2.3) shall be used to compute the capability key (see 5.13.6.8.12). If the SECURITY PROTOCOL SPECIFIC field contains a value that is greater than or equal to D000h, then the authentication key component of the master key (see 5.13.6.8.11) shall be used to compute the capability key.

[b] Mandatory if the CAPKEY CbCS method (see 5.13.6.8.8.3) is supported.

If a CbCS SECURITY PROTOCOL OUT command is received with the INC_512 bit set to one, the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

7.6.c.5 CbCS SECURITY PROTOCOL OUT parameter data

7.6.c.5.1 Set Policy Access Tag CbCS page

The Set Policy Access Tag CbCS page (see table x32) specifies a new policy access tag CbCS parameter value or a new initial policy access tag CbCS parameter value.

Table x32 — Set Policy Access Tag CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0041h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (0004h)				(LSB)
4	(MSB)							
7				POLICY ACCESS TAG				(LSB)

The PAGE CODE field set to 0041h specifies that the Set Policy Access Tag CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Policy Access Tag CbCS page. If the PAGE LENGTH field is set to a value that is smaller than four, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The CbCS parameter (see 5.13.6.8.15) that is set to the contents of the POLICY ACCESS TAG field depends on the logical unit that is processing the Set Policy Access Tag CbCS page as follows:

a) If the logical unit is not the SECURITY PROTCOL well known logical unit, then the contents of the POLICY ACCESS TAG field shall be placed in the policy access tag CbCS parameter for the logical unit; or
b) If the SECURITY PROTOCOL well known logical unit is processing the Set Policy Access Tag CbCS page, then the contents of the POLICY ACCESS TAG field shall be placed in the initial policy access tag CbCS parameter.

7.6.c.5.2 Set Minimum CbCS Method CbCS page

The Set Minimum CbCS Method CbCS page (see table x33) specifies a new minimum CbCS method CbCS parameter value or a new initial minimum CbCS method CbCS parameter value.

Table x33 — Set Minimum CbCS Method CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (0042h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (0002h)				(LSB)
4	Reserved							
5	MINIMUM ALLOWED CBCS METHOD							

The PAGE CODE field set to 0042h specifies that the Set Minimum CbCS Method CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Minimum CbCS Method CbCS page. If the PAGE LENGTH field is set to a value that is smaller than two, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The CbCS parameter or CbCS parameters (see 5.13.6.8.15) that are set to the contents of the MINIMUM ALLOWED CBCS METHOD field depends on the logical unit that is processing the Set Minimum CbCS Method CbCS page and the contents of the MIN CBCS METHOD SUP field in the Unchangeable CbCS Parameters CbCS page (see 7.6.c.3.3) as shown in table x34.

Table x34 — Minimum CbCS Method CbCS Parameter set

MIN CBCS METHOD SUP field	CbCS Parameter set based on the logical unit that processes the Set Minimum CbCS Method CbCS page	
	Not the SECURITY PROTOCOL well known logical unit	SECURITY PROTOCOL well known logical unit
00b	Reserved (see table x27)	
01b	The command is terminated with CHECK CONDITION status with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST	All minimum CbCS method CbCS parameters for all logical units in the SCSI target device, and the initial minimum CbCS method, if any
10b	The minimum CbCS method CbCS parameter for the logical unit that processes the Set Minimum CbCS Method CbCS page	The initial minimum CbCS method CbCS parameter
11b	Reserved (see table x27)	

If the MINIMUM ALLOWED CBCS METHOD field specifies a value that does not appear in the supported CbCS methods list in the Unchangeable CbCS Parameters CbCS page (see 7.6.c.3.3), then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

7.6.c.5.3 Invalidate Key CbCS page

The Invalidate Key CbCS page (see table x35) causes a working key (see 5.13.6.8.11) to be invalidated. After the successful processing of an Invalidate Key CbCS page, the working key with the specified key version shall not be valid for the purposes of enforcement manager (see 5.13.6.8.7) CbCS extension descriptor validation (see 5.13.6.8.13.2).

Table x35 — Invalidate Key CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1				PAGE CODE (D000h)				(LSB)
2	(MSB)							
3				PAGE LENGTH (0004h)				(LSB)
4								
6				Reserved				
7	Reserved				KEY VERSION			

The PAGE CODE field set to D000h specifies that the Invalidate Key CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Invalidate Key CbCS page. If the PAGE LENGTH field is set to a value that is smaller than four, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The KEY VERSION field specifies which working key (e.g., a KEY VERSION field set to four specifies that the working key whose Current CbCS Parameters CbCS page (see 7.6.c.3.5) key identifier is returned in the WORKING KEY 4 IDENTIFIER field) is to be invalidated as follows:

a) If the Invalidate Key CbCS page is processed by a logical unit that is not the SECURITY PROTOCOL well known logical unit, then the specified working key for that logical unit shall be invalidated; or
b) If the Invalidate Key CbCS page is processed by the SECURITY PROTOCOL well known logical unit, then the specified target-wide working key (see 5.13.6.8.11) shall be invalidated.

The CbCS shared key identifier (see 5.13.6.8.11.2) for the invalidated working key shall be set to FFFF FFFF FFFF FFFEh.

It shall not be an error to invalidate a working key that is already invalid.

7.6.c.5.4 Set Key CbCS page

The Set Key CbCS page (see table x36) causes a working key (see 5.13.6.8.11) to be set to a new value. After the successful processing of a Set Key CbCS page, the working key with the specified key version shall be valid for the purposes of enforcement manager (see 5.13.6.8.7) CbCS extension descriptor validation (see 5.13.6.8.13.2).

Table x36 — Set Key CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (D001h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (0020h)				
3								(LSB)
4				Reserved				
6								
7	Reserved				KEY VERSION			
8	(MSB)			KEY IDENTIFIER				
15								(LSB)
16	(MSB)			SEED				
35								(LSB)

The PAGE CODE field set to D001h specifies that the Set Key CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Key CbCS page. If the PAGE LENGTH field is set to a value that is other than 32, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The KEY VERSION field specifies which working key (e.g., a KEY VERSION field set to six specifies that the working key whose Current CbCS Parameters CbCS page (see 7.6.c.3.5) key identifier is returned in the WORKING KEY 6 IDENTIFIER field) is to be set as follows:

a) If the Set Key CbCS page is processed by a logical unit that is not the SECURITY PROTOCOL well known logical unit, then the specified working key for that logical unit shall be set; or
b) If the Set Key CbCS page is processed by the SECURITY PROTOCOL well known logical unit, then the specified target-wide working key (see 5.13.6.8.11) shall be set.

The KEY IDENTIFIER field specifies the value to which the CbCS shared key identifier (see 5.13.6.8.11.2) shall be set for the working key affected by the Set Key CbCS page. If the KEY IDENTIFIER field is set to a value that table x4 (see 5.13.6.8.11.2) describes as reserved in the CbCS pages that change CbCS shared key values, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The seed field contains a random number that is an input to the computation of the new working key value.

The value to which the specified working key is set shall be computed using the integrity check value algorithm specified by the INTEGRITY CHECK VALUE ALGORITHM field in the capability (see 6.r.2.3) in the CbCS extension

descriptor (see 5.13.6.8.16) associated with the SECURITY PROTOCOL OUT command that sent the Set Key CbCS page. The following inputs shall be used with the specified integrity check value algorithm:

a) The contents of the SEED field in the Set Key CbCS page as the string for which the integrity check value is to be computed; and
b) The generation key component of the master key (see 5.13.6.8.11) for the logical unit that is processing the Set Key CbCS page as the cryptographic key.

7.6.c.5.5 Set Master Key – Seed Exchange CbCS page

The Set Master Key – Seed Exchange CbCS page (see table x37) in a SECURITY PROTOCOL OUT command initiates a CbCS master key update CCS (see 5.13.6.8.11.4) and begins a Diffie-Hellman key exchange protocol as part of that CCS.

Table x37 — Set Master Key – Seed Exchange CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (D010h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4	(MSB)			D-H ALGORITHM				
7								(LSB)
8	(MSB)			D-H DATA LENGTH (n-11)				
11								(LSB)
12	(MSB)			D-H DATA				
n								(LSB)

The PAGE CODE field set to D010h specifies that the Set Master Key – Seed Exchange CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Master Key – Seed Exchange CbCS page. If the PAGE LENGTH field is set to a value that is smaller than ten, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The D-H ALGORITHM field specifies the Diffie-Hellman (D-H) algorithm to be used in the seed exchange protocol. The value in the D-H ALGORITHM field is selected from the codes that table 79 (see 5.13.8) lists as Diffie-Hellman algorithms with finite field D-H computations. The Diffie-Hellman algorithm selected specifies the DH_generator value and DH_prime value to be used in the computation of the D-H DATA field as described in this subclause.

If the value in the D-H ALGORITHM field does not appear in the Supported Diffie-Hellman algorithms list in the Unchangeable CbCS Parameters CbCS page (see 7.6.c.3.3), then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The D-H DATA LENGTH field specifies the number of bytes that follow in D-H DATA field. If the D-H DATA LENGTH field is set to a value that is inconsistent with the Diffie-Hellman algorithm specified by the D-H ALGORITHM field, then the

command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The contents of the D-H DATA field are computed as follows:

1) A random number, x, is generated having a value between 0 and DH_prime minus one observing the requirements in RFC 4086; and
2) The D-H DATA field is set to DH_generator^{x} modulo DH_prime, where the DH_generator and DH_prime values are identified by the value in the D-H ALGORITHM field.

7.6.c.5.6 Set Master Key – Change Master Key CbCS page

The Set Master Key – Change Master Key CbCS page (see table x38) concludes a CbCS master key update CCS (see 5.13.6.8.11.4) and changes the master key components to the values computed as part of processing completion (see 7.6.c.3.6) for the SECURITY PROTOCOL IN command that transferred the Set Master Key – Seed Exchange CbCS page.

Table x38 — Set Master Key – Change Master Key CbCS page format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PAGE CODE (D011h)				
1								(LSB)
2	(MSB)			PAGE LENGTH (n-3)				
3								(LSB)
4				Reserved				
7								
8	(MSB)			KEY IDENTIFIER				
15								(LSB)
16	(MSB)			APPLICATION CLIENT D-H DATA LENGTH (k-19)				
19								(LSB)
20	(MSB)			APPLICATION CLIENT D-H DATA				
k								(LSB)
k+1	(MSB)			DEVICE SERVER D-H DATA LENGTH (n-k-4)				
k+4								(LSB)
k+5	(MSB)			DEVICE SERVER D-H DATA				
n								(LSB)

The PAGE CODE field set to D011h specifies that the Set Master Key – Change Master Key CbCS page follows.

The PAGE LENGTH field specifies the number of bytes that follow in the Set Master Key – Change Master Key CbCS page. If the PAGE LENGTH field is set to a value that is smaller than 24, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The KEY IDENTIFIER field specifies the value to which the master key CbCS shared key identifier (see 5.13.6.8.11.2) shall be set. If the KEY IDENTIFIER field is set to a value that table x4 (see 5.13.6.8.11.2) describes as reserved in

the CbCS pages that change CbCS shared key values, then the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The APPLICATION CLIENT D-H DATA LENGTH field specifies the number of bytes that follow in the APPLICATION CLIENT D-H DATA field. If the contents of the APPLICATION CLIENT D-H DATA LENGTH field do not match the number of Diffie-Hellman (D-H) data bytes that the device server received in the SECURITY PROTOCOL OUT command that sent the Set Master Key – Seed Exchange CbCS page (see 7.6.c.5.5) and initiated the current CbCS master key update CCS (see 5.13.6.8.11.4), then master key shall not be modified and the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The APPLICATION CLIENT D-H DATA field contains Diffie-Hellman data field that was sent in the Set Master Key – Seed Exchange CbCS page that initiated the current CbCS master key update CCS. If the contents of the APPLICATION CLIENT D-H DATA field do not match Diffie-Hellman data that the device server received in the SECURITY PROTOCOL OUT command that sent the Set Master Key – Seed Exchange CbCS page that initiated the current CbCS master key update CCS (see 5.13.6.8.11.4), then master key shall not be modified and the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The DEVICE SERVER D-H DATA LENGTH field specifies the number of bytes that follow in the DEVICE SERVER D-H DATA field. If the contents of the DEVICE SERVER D-H DATA LENGTH field do not match the number of Diffie-Hellman data bytes that the device server returned in the SECURITY PROTOCOL IN command that returned the Set Master Key – Seed Exchange CbCS page (see 7.6.c.3.6) in the current CbCS master key update CCS, then master key shall not be modified and the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

The DEVICE SERVER D-H DATA field contains Diffie-Hellman data field that was returned in the Set Master Key – Seed Exchange CbCS page that the device server returned in response to a SECURITY PROTOCOL IN COMMAND as part of the current CbCS master key update CCS. If the contents of the DEVICE SERVER D-H DATA field do not match Diffie-Hellman data that the device server sent in the SECURITY PROTOCOL IN command that send the Set Master Key – Seed Exchange CbCS page, then master key shall not be modified and the command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN PARAMETER LIST.

…

7.7 Vital product data parameters

…

7.7.4 Extended INQUIRY Data VPD page

The Extended INQUIRY Data VPD page (see table 437) provides the application client with a means to obtain information about the logical unit.

Table 437 — Extended INQUIRY Data VPD page

Bit / Byte	7	6	5	4	3	2	1	0
0	PERIPHERAL QUALIFIER			PERIPHERAL DEVICE TYPE				
1	PAGE CODE (86h)							
2	Reserved							
3	PAGE LENGTH (3Ch)							
4	Reserved		SPT			GRD_CHK	APP_CHK	REF_CHK
5	Reserved		UASK_SUP	GROUP_SUP	PRIOR_SUP	HEADSUP	ORDSUP	SIMPSUP
6	Reserved				WU_SUP	CRD_SUP	NV_SUP	V_SUP
7	Reserved							LUICLR
8	Reserved							CBCS
9 ~~8~~ – 63	Reserved							

…

A capability-based command security (CBCS) bit set to one indicates that the logical unit supports the capability-based command security technique (see 5.13.6.8). A CBCS bit set to zero indicates that the logical unit does not support the capability-based command security technique.

7.7.5 Management Network Addresses VPD page

{{Several *pro forma* changes follow this note. The subclause and table numbering is reset at this point.}}

5.6 Reservations

…

5.6.1 Persistent Reservations overview

…

For each command, this standard or a command standard (see 3.1.18) defines the conditions that result in RESERVATION CONFLICT. Command standards define the conditions either in the device model or in the descriptions each of specific command.

Table 35 — SPC commands that are allowed in the presence of various reservations

Command	Addressed logical unit has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From not registered I_T nexus	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
…	…	…	…	…	…
RECEIVE COPY RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RECEIVE CREDENTIAL	Conflict	Conflict	Allowed	Conflict	Conflict
…					

…

6.1 Summary of commands for all device types

The operation codes for commands that apply to all device types when the MCHNGR bit is set to zero, the SCCS bit is set to zero, and the ENCSERV bit is set to zero in the standard INQUIRY data (see 6.4.2) are listed in table 75.

Table 75 — Commands for all device types

Command name	Operation code	Type	Reference
…	…	…	…
RECEIVE COPY RESULTS	84h	O	6.18
RECEIVE CREDENTIAL	7Fh/1800h [a]	O	6.r
Type Key: C = Command implementation is defined in the applicable command standard (see 3.1.18). M = Command implementation is mandatory. O = Command implementation is optional. Z = Command implementation is defined in a previous standard.			
[a] This command is defined by a combination of operation code and service action. The operation code value is shown preceding the slash and the service action value is shown after the slash.			

…

6.30 SECURITY PROTOCOL IN command

…

The SECURITY PROTOCOL field (see table 220) specifies which security protocol is being used.

Table 220 — SECURITY PROTOCOL field in SECURITY PROTOCOL IN command

Code	Description	Reference
00h	Security protocol information	7.6.1
01h - 06h	Defined by the TCG	3.1.169
07h	CbCS	7.6.c
~~07h~~ 08h - 1Fh	Reserved	
20h	Tape Data Encryption	SSC-3
21h	Data Encryption Configuration	TBD
22h - 3Fh	Reserved	
40h	SA Creation Capabilities	7.6.2
41h	IKEv2-SCSI	7.6.3
42h - ECh	Reserved	
EDh	SD Card TrustedFlash specification	3.1.137
EEh	Authentication in Host Attachments of Transient Storage Devices	IEEE 1667
EFh	ATA Device Server Password Security	SAT-2
F0h - FFh	Vendor Specific	

…

6.31 SECURITY PROTOCOL OUT command

…

The SECURITY PROTOCOL field (see table 222) specifies which security protocol is being used.

Table 222 — SECURITY PROTOCOL field in SECURITY PROTOCOL OUT command

Code	Description	Reference
00h	Reserved	
01h - 06h	Defined by the TCG	3.1.169
07h	CbCS	7.6.c
~~07h~~ 08h - 1Fh	Reserved	
20h	Tape Data Encryption	SSC-3
21h	Data Encryption Configuration	TBD
22h - 40h	Reserved	
41h	IKEv2-SCSI	7.6.3
42h - ECh	Reserved	
EDh	SD Card TrustedFlash specification	3.1.137
EEh	Authentication in Host Attachments of Transient Storage Devices	IEEE 1667
EFh	ATA Device Server Password Security	SAT-2
F0h - FFh	Vendor Specific	

…

D.3.5 Variable length CDB service action codes

The variable length CDB service action codes assigned by this standard are shown in table D.8.

Table D.8 — Variable Length CDB Service Action Codes Used by All Device Types

Service Action Code	Description
1800h	RECEIVE CREDENTIAL command
1801h - 1FFFh	Reserved
~~1800h - 1FFFh~~	~~Reserved~~