Date: 7 November 2007
To: T10 Technical Committee
From: Matt Ball (Quantum), David Black (EMC), and Ralph O. Weber (ENDL)
Subject: SPC-4: Establishing a Security Association using IKEv2

Introduction

This proposal provides a method, named IKEv2-SCSI, for creating a security association using Diffie-Hellman (DH) key establishment with nonces based on IETF RFC 4306 "IKEv2" and guidance from NIST SP 800-56A.

A security association provides the infrastructure necessary for sending encrypted messages with cryptographic integrity protection between the application client and device server, and allows end-point authentication that is linked to the cryptographic integrity protection in order to prevent man-in-the-middle attacks.

References

T10/SSC-3r3c	SCSI-3 Stream Commands.
T10/SPC-4r11	SCSI Primary Commands.
T10/06-225r5	Matt Ball, SSC-3: Key Entry using Encapsulating Security Payload (ESP).
NIST SP 800-56A	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography.
T11/06-157v3	Fibre Channel - Security Protocols (FC-SP)

Differences between IKEv2 and IKEv2-SCSI

The important differences between IKEv2 and IKEv2-SCSI include the following:

a) An SA created by the IKEv2-SCSI protocol is used to directly protect SCSI traffic. There is no concept of child SAs; this is based on a design assumption that SA usage will be infrequent in SCSI command streams;
b) The entity sending SCSI traffic determines what SA is used and what is to be protected via appropriate use of the SAI for the SA. All or only a portion of the parameter data may be protected. SCSI addresses are not involved in this determination, and hence IKEv2-SCSI does not provide address-based data origin authentication and hence does not use traffic selectors to associate security processing with SCSI traffic; this functionality is left to SCSI transport-specific security mechanisms, in part because SCSI addresses are transport-specific. SCSI command standards define the uses for SAs and the mechanisms for communicating the applicable SAIs between application clients and device servers;
c) Cryptographic algorithm negotiation has been replaced by a simplified mechanism based on a SCSI device capabilities design approach. The simplification includes removal of IKEv2's proposal concept; the application client chooses algorithms supported by the device server in accordance with the application client's policy and preferences; and
d) Significant portions of IKEv2 have been removed as inapplicable to SCSI. The removed functionality includes Traffic Selectors, NAT Traversal, Remote Configuration, and Compression.

In IKEv2 terminology, the application client is the IKEv2 initiator and the device server is the IKEv2 responder. A device server cannot initiate IKEv2-SCSI.

Revision History

r0 Initial revision with lots of help from Ralph Weber.
r1 Incorporate comments from discussion in November Las Vegas meeting. Major changes:
- Add timeout support (new STV payload). Timeouts are recommended (should) rather than mandatory (shall).
- Change sequencing support to talk about device server discarding state instead of mandatory timeouts.

- Changed most IKEv2-SCSI-specific ASC/ASCQs to new values.
- Require 16 kilobytes of parameter data support.
- Tweak Certificate Encoding field in Certificate Request payload so that it tells the device server whether or not a URL-based certificate format is acceptable to the application client.
- Make support for skipping authentication optional.
- Specify and explain what not to do with the PROGRESS INDICATION sense data.
- Add usage data to SCA payload
- Added the notify (Initial contact only) and delete payloads
- [Added a number of Editor's notes indicating significant additional work to be done ;-).]

r2 Incorporate comments from January Orlando meeting and additional design work. Major changes:
- Use separate IKEv2-SCSI SA Creation Capabilities payload in Device Server Capabilities phase. This removes the erroneous use of Usage Data in the Device Server Capabilities phase.
- Adopt certificate encoding recommendations from RFC 4718 and in addition prohibit Hash and URL certificate formats.
- Add new IANA-allocated values for crypto mechanisms. Remove GMAC as RFC 4543 does not define its use with IKEv2. Adjust vendor-specific ranges to match IANA private use ranges for IKEv2.
- Allow Fibre Channel names as identifiers (for FC-SP certificates).
- Tighten down specification of Delete to reflect removal of Child SAs and avoid problems - it has to be sent on the SA to be deleted because there are no child SAs. Add model clause to explain how Delete works.
- Factor out SA creation command sequencing into a separate subsection. This reduces the amount of text and covers a number of additional error cases.
- Add subsection on how to populate the fields of an SA.
- Renumber IKEv2-SCSI exchange types into IKEv2's private use range. Add additional explanation of IKEv2 header fields, including sequence association checks and errors.
- Lots of other edits and changes (e.g., to remove Editor's notes).

r3 Incorporate comments from March Memphis meeting. Major changes:
- Remove DSS digital signature support. Remove support for encryption without integrity. Finish aligning cryptographic algorithm identifiers with IANA registries for IKEv2.
- Terminology change to SA creation cryptographic command sequence, only allow one at a time per I_T_L Nexus (so device server only has to save one set of active parameters), but return NOT READY if another is attempted instead of aborting the original sequence.
- Adapt to USAGE changes made to SA proposal as part of approving it.
- Add key length values to encryption algorithm table. Add notes about extra salt bytes that CCM and GCM mode take from KEYMAT.

r4 Incorporate comments from April Houston interim meeting. Major changes:
- Change terminology from protocol "phases" to protocol "steps", add summary of protocol steps.
- Distinguish IKEv2-SCSI keys from SA keys for clarity.
- Modify (generally reduce) allowed cryptographic algorithms. GMAC cannot be added because IETF does not support GMAC usage in IKEv2.
- Add AUTH_NONE integrity support (no separate integrity algorithm) for use with AES_CCM and AES_GCM combined mode algorithms that provide integrity.
- Expand SA type (usage) to 2 bytes and reformat SCA payload accordingly. Did not add a second pair of nonces because IKEv2 (RFC 4306) uses the same nonces for the IKEv2 (SA) keys and the keys for the first child SA.

r5 Convert to FrameMaker and edit for T10 style

r6 Complete conversion to FrameMaker and incorporate comments from May CAP WG (minutes in 07-212), including the concept of tying the mandatory SA authentication algorithm to the USAGE_TYPE SA parameter.

r7 Incorporate comments received prior to the Colorado Springs CAP WG meeting
- Changed hard errors in IKE header processing to soft errors for the Authentication step to avoid denial of service attacks
- Changed nonces to be zero upon completion of SA creation. For security, the nonces are not to be reused after the SA keys are generated, and setting them to zero ensures that.

- Removed linkage between USAGE_TYPE and mandatory algorithms.
- Split cryptographic algorithm negotiations into SA Creation and Usage Type elements, with separate payloads for each.
- Clarified combined mode encryption and integrity checking key computations and algorithm usage.
- Corrected egregious errors in the Encrypted payload format definition.
- Added requirements for payloads based on CDB SECURITY PROTOCOL SPECIFIC field contents.
- Made numerous other changes requested by Paul Entzel and David Black.

r8 Made changes requested by the 14 August CAP Security conference call.

r9 Made the following changes requested by David Black:

- Corrected to typos in 5.13.4.9.2 (Generating shared keys when the Authentication step skipped): INGEG s/b INTEG.
- Changed all occurrences of 'cyphertext' to 'ciphertext'. No change bars for this change.
- Added a description of the inputs to the INTEGRITY CHECK VALUE field computation in 7.7.3.5.10.1.
- Corrected the 7.7.3.5.10.1 description of the encryption algorithm inputs to show than the order of AAD, IV, and plaintext matters.
- Clarified table 7.7.3.5.10.3 (Processing a received Encrypted payload) regarding which cases apply while a CCS is active and which apply after the CCS processing has finished.
- The key length column was removed from the table of SA authentication algorithms (see table x43) because there is no way to define fixed values for several of the key lengths and because the data in the column is no longer used.
- An output length column was added to the table of PRF algorithms (see table x37) and the SK_p? row in table x2 was modified to reference this new, correct information.
- Removed comment about bogus key lengths following table x43.

Made the following change requested by Matt Ball and others.

- Changed all instances of 'secure key (SK)' to 'shared key (SK)'. No change bars for this change.

Made the editor happy by adding structure formats for the Identification, Certificate, and Vendor ID payloads.

Updated 5.13.4.10.3 to account for the possibility that the encryption algorithm type is ENCR_NULL.

Updated 7.7.3.5.12 to prohibit ENCR_NULL as an encryption algorithm for SA creation.

Updated 7.7.3.6.2 to force errors to be returned for invalid combinations of ENCR_xxxx encryption algorithms and AUTH_COMBINED.

Rearranged a little of the information in 5.13.4.1 and tons of the information in 5.13.4.10 to untangle the accumulated shared-key-generation cruf, eliminate unnecessary complexity, and better represent the outline of what happens in the table of contents. These changes also provide the SCSI-ESP proposal (07-169) with a needed reference about the shared keys generated during SA creation.

Incorporated editorial and obviously correct comments from Rob Elliott and George Penokie.

Document number clocked to T10/07-437r0 due to too many revisions

r0 A bunch of DC_xxx SA parameter name typos which should be DS_xxx were fixed (no change bars were added for these).

The following September CAP working group recommended changes were made:

- An ECDSA acronym definition was added.
- Application client checking of cryptographic algorithm payloads returned by the device server was strengthened to a 'should'.
- Both occurrences of 'trust anchor' were decorated with RFC 4306 references.
- The IKEv2-SCSI requirements on pre-shared keys were moved to a separate subclause in the model.
- The chinese menu of cryptographic algorithm choices requested by the September CAP WG have been specified in list form in 5.13.4.7.
- The SK_xxx entries were moved from the acronyms list to the glossary.
- Several detailed descriptions of actions by an administrator were removed as requested.
- Several changes were made to clarify the relationship between SA parameter and the SPC-4 defined KDFs.
- The description of REQUEST SENSE processing when a security cryptographic operation is in progress was revised to more closely match the self-test equivalent, more or less as requested.
- Size specifications for the SECURITY PROTOCOL IN allocation length were deleted.

- The tables specifying next payload contents were reviewed and revised to conform with the restrictions discussed in the September CAP WG. As a result of the review, the specific changes differed from those suggested by CAP.
- The discussion of D-H exponential reuse was revised as requested.
- A D-H usage constraint that the proposal now implemented in the D-H algorithm identifier table was deleted from the text.
- The order of fields in the UT Cryptographic Algorithms payload was swapped to improve alignment and followed the SCSI convention of placing length fields immediately before the variable length fields that they describe.
- The 'IKEv2-SCSI parameter error categories' table was restructured to use table footnotes as requested.
- Numerous other changes involving fewer than two dozen words were made as requested.

The SA_AUTH cryptographic algorithm descriptor is replaced with SA_AUTH_OUT (which defines the authentication method in the Authentication step SECURITY PROTOCOL OUT command parameter data) and SA_AUTH_IN (which defines the authentication method in the Authentication step SECURITY PROTOCOL IN command). This eliminates the need for the ACCEPT and USE bits in the SA_AUTH algorithm descriptor and more directly aligns IKEv2-SCSI with SCSI concepts. However, it slightly increases the complexity of the tests that must be performed during the Key Exchange step and substantially increases the complexity of the text that describes the required tests.

r1 Modified description Notify payload in the model (see 5.13.4.2) as requested by e-mail from Kevin Butt and David Black. Also removed the document classification terms (i.e., Secret and Top Secret) from algorithms lists definitions in 5.13.4.7, as suggested by David Black.

r2 Converted the algorithms lists definitions in 5.13.4.7 to a table format as suggested by the 15 October conference call.

r3 Made the change requested by the October 17 telephone conference call. Made numerous changes requested by David Black and Kevin Butt.

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.

Proposed Changes in SPC-4 r10

Introduction

The SCSI Primary Commands - 4 (SPC-4) standard is divided into the following clauses and annexes:

Clause 1	is the scope.
Clause 2	enumerates the normative references that apply to this standard.
Clause 3	describes the definitions, symbols, and abbreviations used in this standard.
Clause 4	describes the conceptual relationship between this document and the SCSI-3 Architecture Model.
Clause 5	describes the command model for all SCSI devices.
Clause 6	defines the commands that may be implemented by any SCSI device.
Clause 7	defines the parameter data formats that may be implemented by any SCSI device.
Clause 8	defines the well known logical units that may be implemented by any SCSI device.
Annex A	identifies differences between the terminology used in this standard and previous versions of this standard. (informative)
Annex B	describes the PERSISTENT RESERVE OUT command features necessary to replace the reserve/release management method and provides guidance on how to perform a third party reservation using persistent reservations. (informative)
Annex C	identifies the differences between IKEv2 (see RFC 4306) and the IKEv2-SCSI SA creation protocol defined by this standard.
Annex ~~C~~D	lists code values in numeric order. (informative)

Annex ~~D~~E lists assigned vendor identifiers. (informative)

…

2.2 Normative References

…

ISO/IEC 14165-251, *Fibre Channel Framing and Signaling Interface (FC-FS) [ANSI INCITS 373-2003]*

ISO/IEC 14165-431, *Fibre Channel Security Protocols (FC-SP) [ANSI INCITS 426-2007]*

IEC 60027:2000, *Letter symbols to be used in electrical technology - Part 2: Telecommunications and electronics*

…

2.4 NIST References

~~Copies of the following approved NIST standards may be obtained through the National Institute of Standards and Technology (NIST) at http://csrc.nist.gov/publications/nistpubs/index.html.~~

~~NIST SP (Special Publication) 800-38C, *Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality*~~

{{The remainder of 2.4 is unchanged.}}

2.5 IETF References

{{add the following references to those already listed and maintain RFC number order}}

RFC 2104, *HMAC: Keyed-Hashing for Message Authentication*

RFC 2404, *The Use of HMAC-SHA-1-96 within ESP and AH*

RFC 2410, *The NULL Encryption Algorithm and Its Use With IPsec*

RFC 2437, *PKCS #1: RSA Cryptography Specifications Version 2.0*

RFC 3280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*

RFC 3447, *Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1*

RFC 3602, *The AES-CBC Cipher Algorithm and Its Use with IPsec*

RFC 3526, *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*

RFC 4106, *The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload*

RFC 4309, *Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload*

RFC 4434, *The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)*

RFC 4595, *Use of IKEv2 in the Fibre Channel Security Association Management Protocol*

RFC 4718, *IKEv2 Clarifications and Implementation Guidelines*

RFC 4753, *ECP Groups for IKE and IKEv2*

RFC 4754, *IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)*

RFC 4868, *Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec*

…

3.1 Definitions

{{insert the following in the proper alphabetical order.}}

3.1.a additional authenticated data (AAD): Bytes of data input to a combined mode encryption algorithm (e.g., AES-GCM) as part of integrity checking computations.

3.1.c cryptographic command sequence (CCS): A defined sequence of SECURITY PROTOCOL IN commands (see 6.29) and SECURITY PROTOCOL OUT commands (see 6.30) that realize the cryptographic protocol of a specified security operation (e.g., the SECURITY PROTOCOL IN commands and SECURITY PROTOCOL OUT commands in the Key Exchange step and Authentication step, if any, of an IKEv2-SCSI (see 3.1.f) SA (see 3.1.119) creation transaction).

3.1.f IKEv2-SCSI: Internet Key Exchange protocol version 2 for SCSI. See 5.13.4.

3.1.g IKEv2-SCSI keys: Shared keys (see 3.1.w) used to provide security for IKEv2-SCSI operations (e.g., creation and deletion of the SA). IKEv2-SCSI keys that are used after SA creation is complete are maintained in the MGMT_DATA SA parameter (see 3.1.103). The shared keys names used by IKEv2-SCSI are listed in 5.13.4.4.

3.1.h IKEv2-SCSI CCS: The CCS (see 3.1.c) that is the Key Exchange step and Authentication step, if any, of an IKEv2-SCSI (see 3.1.f) SA (see 3.1.119) creation transaction.

3.1.k pre-shared key: A shared key (see 3.1.w) that is established by a method outside the scope of this standard prior to the initiation of the function that requires its use. Requirement on pre-shared keys that are particular to this standard are described in 5.13.4.5. Some of the RFCs referenced by this standard use the name pre-shared secret for the same information that this standard calls a pre-shared key.

3.1.o SA creation transaction: Any sequence of SECURITY PROTOCOL IN commands (see 6.29) and SECURITY PROTOCOL OUT commands (see 6.30), including but not limited a CCS (see 3.1.c), that is used to create an SA between an application client and device server.

3.1.p SA generation: Computation and initialization of the SA parameter values (see 3.1.103) required to create an SA (see 3.1.119) that is performed separately by the application client and device server after all SCSI commands required to create an SA have been performed without error. See 5.13.4.11.

3.1.q SA keys: Shared keys (see 3.1.w) that are maintained in the USAGE_DATA SA parameter (see 3.1.103) and used to provide security for the operations that use the SA (e.g., encryption of command parameter data). The shared keys names used by IKEv2-SCSI are listed in 5.13.4.4.

3.1.r SA participant: An application client or device server that participates in the creation or use of an SA (see 3.1.119).

3.1.vh SK_ai: Shared Key (see 3.1.w) for IKEv2-SCSI Data-Out Buffer Encrypted payload integrity checking (see 5.13.4).

3.1.vi SK_ar: Shared Key (see 3.1.w) for IKEv2-SCSI Data-In Buffer Encrypted payload integrity checking (see 5.13.4).

3.1.vj SK_d: Shared Key (see 3.1.w) KDF input material use to generate the shared keys stored in the KEYMAT SA Parameter (see 3.1.103).

3.1.vk SK_ei: Shared Key (see 3.1.w) for IKEv2-SCSI Data-Out Buffer Encrypted payload encryption (see 5.13.4).

3.1.vl SK_er: Shared Key (see 3.1.w) for IKEv2-SCSI Data-In Buffer Encrypted payload encryption (see 5.13.4).

3.1.vm SK_pi: Shared Key (see 3.1.w) used to construct the IKEv2-SCSI Data-Out Buffer Authentication payload (see 5.13.4).

3.1.vn SK_pr: Shared Key (see 3.1.w) used to construct the IKEv2-SCSI Data-In Buffer Authentication payload (see 5.13.4).

3.1.w shared key (SK): A cryptographically protected secret (e.g., with more randomness (see RFC 4089) than a password) that is known only to a defined and limited set of entities (e.g., one application client and one device server). The shared keys names used by IKEv2-SCSI are listed in 5.13.4.4.

3.2 Symbols and acronyms

{{insert the following in the proper alphabetical order.}}

AAD	Additional Authenticated Data (see 7.7.3.5.10)
AES-GCM	Advanced Encryption Standard - Galois Counter Mode (see RFC 4106)
CCS	cryptographic command sequence (see 3.1.c)
ECP	Elliptic Curve group modulo Prime (see RFC 4753)
ECDSA	Elliptic Curve Digital Signature Algorithm (see RFC 4754)
MODP	Modular exponential
PKI	Public Key Infrastructure (see RFC 3280)
PRF	Pseudo-Random Function (see RFC 4306)
SK	Shared Key (see 3.1.w)
UT	Usage Type

…

5.6.3 Exceptions to SPC-2 RESERVE and RELEASE behavior

…

{{Insert the following new subclause in the Persistent Reservations model and renumber all subsequent subclauses.}}

5.6.4 Persistent reservations interactions with IKEv2-SCSI SA creation

If a PERSISTENT RESERVE OUT command is received while an IKEv2-SCSI CCS is in progress (see 5.13.4), the command shall be terminated with a CHECK CONDITION status, with the sense key NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS. The sense key specific additional sense data may be set as described in 5.13.5.

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

Editors Note 1 - ROW: David Black has asked if this requirement is overkill. Does the application client need to know that a potentially long-running SA Creation will not be affect by the reservation it is establishing?

…

5.13.2 Security associations

5.13.2.1 Principals of ~~security associations~~ SAs

…

5.13.2.2 Creating a security association

The SECURITY PROTOCOL IN command (see 6.29) and SECURITY PROTOCOL OUT command (see 6.30) security protocols shown in table 46 are used to create SAs. The process of creating an SA establishes the SA parameter (see 5.13.2.2) values as follows:

a) Initial values for:
   A) Both (i.e., application client and device server) sequence numbers set to zero; and
   ~~B) All KEYMAT bytes set to zero;~~
b) Unchanging values for the lifetime of the SA:
   A) Both SAIs;
   B) TIMEOUT;
   ~~C) Both nonces;~~
   D) KDF_ID;
   E) KEYMAT;
   F) USAGE_TYPE;
   G) USAGE_DATA; and
   H) MGMT_DATA;
   and
c) Values that are zero upon completion of SA creation:
   A) KEY_SEED; and
   B) Both nonces.

Table 46 — Security protocols ~~that~~ used to create SAs

Security Protocol Code	Description	Reference
~~TBD~~	~~TBD~~	~~TBD~~
zzh	SA creation capabilities	7.7.2
xxh	IKEv2-SCSI	5.13.4

5.13.3 Key derivation functions

…

5.13.4 Using IKEv2-SCSI to create a security association

{{All of 5.13.4 and 5.13.5 are new. Additions/deletions markups are not applied in these subclauses.}}

5.13.4.1 Overview

The IKEv2-SCSI protocol is a subset of the IKEv2 protocol (see RFC 4306) that this standard defines for use in the creation and maintenance of an SA (see 3.1.119).

An IKEv2-SCSI SA creation transaction (see 3.1.o) shall only be initiated by the application client.

The IKEv2-SCSI protocol creates the following pair of IKE SAs (see RFC 4306):

a) An SA that protects data sent from the application client to the device server; and
b) An SA that protects data sent from the device server to the application client.

An IKEv2-SCSI SA creation transaction consists of the following steps:

1) Device Server Capabilities step (see 5.13.4.7): The application client determines the device server's cryptographic capabilities;
2) Key Exchange step (see 5.13.4.8): The application client and device server:
   A) Perform a key exchange;
   B) Determine SAIs (see 3.1.120);
   C) Generate the shared keys used for SA management (e.g., SA creation and deletion) (see 5.13.4.10); and
   D) May complete the generation of the SA (see 5.13.4.11);
   and
3) Authentication step (see 5.13.4.9): Unless omitted by application client and device server negotiations in the previous steps:
   A) The application client and device server authenticate:
      a) Each other;
      b) The key exchange; and
      c) The capability selection;
      and
   B) Complete the generation of the SA (see 5.13.4.11).

The values in the SECURITY PROTOCOL field and the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL IN command (see 6.29) and SECURITY PROTOCOL OUT command (see 6.30) identify the step of the IKEv2-SCSI protocol (see 7.7.3.2).

The Key Exchange step and the Authentication step depend on the results from the Device Capabilities step in order to create an SA. During the IKEv2-SCSI Key Exchange step, the application client and device server perform independent computations to construct the following sets of shared keys:

a) Shared keys that are used by the IKEv2-SCSI Authentication step;
b) Shared keys that are used by the IKEv2-SCSI Authentication step and to delete the SA; and
c) Shared keys are used by SCSI UT operations that obtain security from the generated SA.

More details about these shared keys are provided in 5.13.4.4.

An application client may or may not:

a) Proceed to the Key Exchange step after the Device Server Capabilities step; or
b) Perform a separate Device Server Capabilities step for each IKEv2-SCSI SA creation transaction.

If the device server's capabilities have changed, the Authentication step returns an error, and the Key Exchange step may return an error.

Changes in the device server’s capabilities do not take effect until at least one application client has been notified of the new capabilities via the parameter data returned by the Device Server Capabilities step.

After a Device Capabilities step, the application client performs SA creation by sending a sequence of two or four IKEv2-SCSI commands over a single I_T_L nexus to the device server. The following commands constitute an IKEv2-SCSI CCS (see 3.1.h):

a) If the Authentication step is skipped (see 5.13.4.6):
   1) A Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2); and
   2) A Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3);
   or
b) If the Authentication step is performed:
   1) A Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2);
   2) A Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3);
   3) An Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2); and
   4) An Authentication step SECURITY PROTOCOL IN command (see 5.13.4.9.3).

The device server shall process each command in the IKEv2-SCSI CCS to completion before returning status. While a command in the IKEv2-SCSI CCS is being processed by the device server, the application client may use the REQUEST SENSE command (see 5.13.5) to ascertain the device server’s progress for the command.

When an error is encountered, the device server or application client may abandon the IKEv2-SCSI CCS before the SA is created (see 5.13.4.12).

The device server shall maintain state for the IKEv2-SCSI CCS on a given I_T_L nexus from the time the Key Exchange step SECURITY PROTOCOL OUT command is completed with GOOD status until one of the following occurs:

a) The IKEv2-SCSI CCS completes successfully;
b) The IKEv2-SCSI CCS is abandoned as described in 5.13.4.12;
c) The SA being created by the IKEv2-SCSI CCS is deleted as described in 5.13.4.13;

d) The number of seconds specified in the IKEV2-SCSI PROTOCOL TIMEOUT field of the IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14) in the Key Exchange step SECURITY PROTOCOL OUT parameter data elapses and none of the following commands have been received:
   A) The next command in the IKEv2-SCSI CCS; or
   B) A REQUEST SENSE command;
   or
e) One of the following event-related SCSI device conditions (see SAM-4) occurs:
   A) Power cycle;
   B) Hard reset;
   C) Logical unit reset; or
   D) I_T nexus loss.

If the device server receives a SECURITY PROTOCOL OUT or SECURITY PROTOCOL IN command with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) on an I_T_L nexus other than the one for which IKEv2-SCSI CCS state is being maintained, then:

a) An additional IKEv2-SCSI CCS may be started; or
b) The command may be terminated with CHECK CONDITION status, with the sense key set to ABORTED COMMAND, and the additional sense code set to CONFLICTING SA CREATION REQUEST.

{{CONFLICTING SA CREATION REQUEST is a new additional sense code.}}

Except for the PERSISTENT RESERVE OUT command (see 5.6.4) and the cases described in this subclause, a device server that is maintaining a IKEv2-SCSI CCS state on a particular I_T_L nexus shall not alter its processing of new commands received on that I_T_L nexus.

If all of the following conditions are true:

a) The device server includes the following algorithm descriptors in the IKEv2-SCSI SA Creation Capabilities payload (see 7.7.3.5.11) in the parameter data returned by the Device Server Capabilities step SECURITY PROTOCOL IN command (see 5.13.4.7):
   A) an SA_AUTH_OUT algorithm descriptor (see 7.7.3.6.6) with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE; and
   B) an SA_AUTH_IN algorithm descriptor (see 7.7.3.6.6) with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE;
   and
b) The application client sends the following algorithm descriptors to the device server in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2):
   A) an SA_AUTH_OUT algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE; and
   B) an SA_AUTH_IN algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE;

then:
a) The Authentication step is skipped;
b) The IKEv2-SCSI CCS consists of the two Key Exchange step commands;
c) The device server requires the IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13) to be present in the parameter data sent by the Key Exchange step SECURITY PROTOCOL OUT command;
d) The device server returns the IKEv2-SCSI UT Cryptographic Algorithms payload in the parameter data returned by the Key Exchange step SECURITY PROTOCOL IN command; and
e) SA creation occurs upon the completion of the Key Exchange step.

Proper operation of an IKEv2-SCSI CCS depends on SA_AUTH_NONE being used in both the Authentication step SECURITY PROTOCOL OUT command and the Authentication step SECUIRTY PROTOCOL IN command, or

SA_AUTH_NONE not being used by either command. Processing requirements placed on the Key Exchange step SECURITY PROTOCOL OUT command (see 7.7.3.6.6) ensure that this dependency is satisfied.

If no other errors are detected and any of the following conditions are true:

a) The device server does not include an SA_AUTH_OUT algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE in the IKEv2-SCSI SA Creation Capabilities payload in the parameter data returned by the Device Server Capabilities step SECURITY PROTOCOL IN command;
b) The device server does not include an SA_AUTH_IN algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE in the IKEv2-SCSI SA Creation Capabilities payload in the parameter data returned by the Device Server Capabilities step SECURITY PROTOCOL IN command; or
c) The application client does not set the ALGORITHM IDENTIFIER field to SA_AUTH_NONE in the SA_AUTH_OUT algorithm descriptor and the SA_AUTH_IN algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload sent to the device server in the Key Exchange step SECURITY PROTOCOL OUT command;

then:

a) The Authentication step is processed;
b) The IKEv2-SCSI CCS consists of the two Key Exchange step commands and two Authentication step commands;
c) The device server requires the IKEv2-SCSI UT Cryptographic Algorithms payload to be absent from the parameter data sent by the Key Exchange step SECURITY PROTOCOL OUT command;
d) The device server omits the IKEv2-SCSI UT Cryptographic Algorithms payload from the parameter data returned by the Key Exchange step SECURITY PROTOCOL IN command;
e) The device server requires the IKEv2-SCSI UT Cryptographic Algorithms payload to be present in the parameter data sent by the Authentication step SECURITY PROTOCOL OUT command;
f) The device server returns the IKEv2-SCSI UT Cryptographic Algorithms payload in the parameter data returned by the Authentication step SECURITY PROTOCOL IN command; and
g) SA creation occurs upon the completion of the Authentication step.

SA participants should perform the Authentication step unless man-in-the-middle attacks (see 5.13.1.4) are not of concern or are prevented by other means (e.g., physical security of the transport).

NOTE x1 - Omission of the Authentication step provides no defense against a man-in-the-middle adversary that is capable of modifying SCSI commands. Such an adversary is able to insert itself as an intermediary on the created SA without knowledge of the SA participants, thereby completely subverting the intended security. Omission of the Authentication step is only appropriate in environments where the absence of such adversaries is assured by other means (e.g., a direct physical connection between the systems on which the application client and device server or use of end-to-end security in the SCSI transport security such as FC-SP).

5.13.4.2 IKEv2-SCSI Protocol summary

This subclause summarizes the IKE-v2-SCSI payloads (see 7.7.3.5) that are exchanged between an application client and a device server during all steps of an IKEv2-SCSI SA creation transaction (see 3.1.o) using message diagrams. Each IKEv2-SCSI step (see 5.13.4.1) is shown in a separate figure. The contents of a payload (e.g., Key Exchange) may not be the same in both directions of transfer.

Figure x1 shows the Device Server Capabilities step (see 5.13.4.7). The Device Server Capabilities step consists of a SECURITY PROTOCOL IN command carrying an IKEv2-SCSI SA Creation Capabilities payload (see 7.7.3.5.11). The IKEv2-SCSI header is not used.

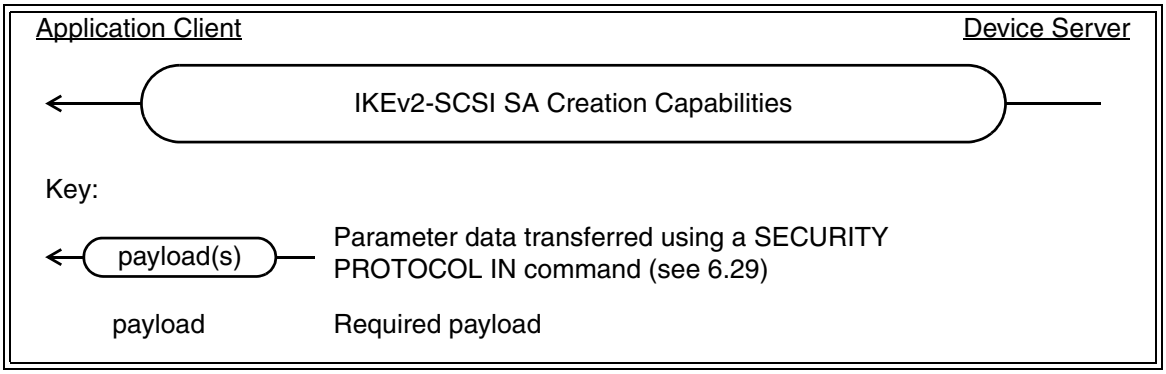

Figure x1 — IKEv2-SCSI Device Server Capabilities step

The IKEv2-SCSI SA Creation Capabilities payload indicates the device server's capabilities for SA creation.

Figure x2 shows the Key Exchange step (see 5.13.4.8). The Key Exchange step consists of a SECURITY PROTOCOL OUT command followed by a SECURITY PROTOCOL IN command.

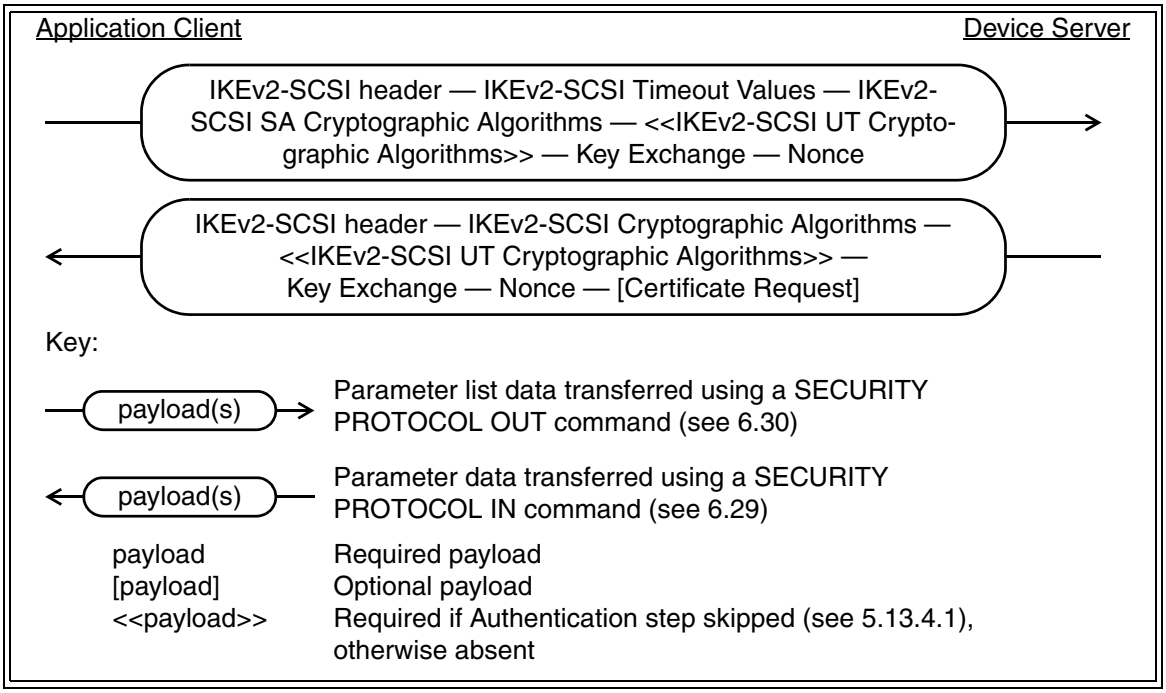

Figure x2 — IKEv2-SCSI Key Exchange step

The IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14) contains timeouts for SA creation and usage.

The IKEv2-SCSI SA Cryptographic Algorithms payloads (see 7.7.3.5.12) are used to select and agree on the cryptographic algorithms used for creating the SA.

If the Authentication step is skipped (see 5.13.4.1), the IKEv2-SCSI UT Cryptographic Algorithms payloads (see 7.7.3.5.13) are used to select and agree on usage of the SA and the cryptographic algorithms used by the created

SA. If the Authentication step is processed (i.e., not skipped), the SA usage and algorithms selection is performed there.

The Key Exchange payload (see 7.7.3.5.3) and Nonce payload (see 7.7.3.5.7) are part of the key and nonce exchanges that are used to generate SA keys.

The optional Certificate Request payload or payloads (see 7.7.3.5.5) enables the device server to request a certificate from the application client. If the Authentication step is being skipped (see 5.13.4.1), the device server shall not include any Certificate Request payloads in the parameter data. Use of the Certificate Request payload is described in 5.13.4.3.

Figure x3 shows the Authentication step (see 5.13.4.9). The Authentication step consists of a SECURITY PROTOCOL OUT command followed by a SECURITY PROTOCOL IN command.

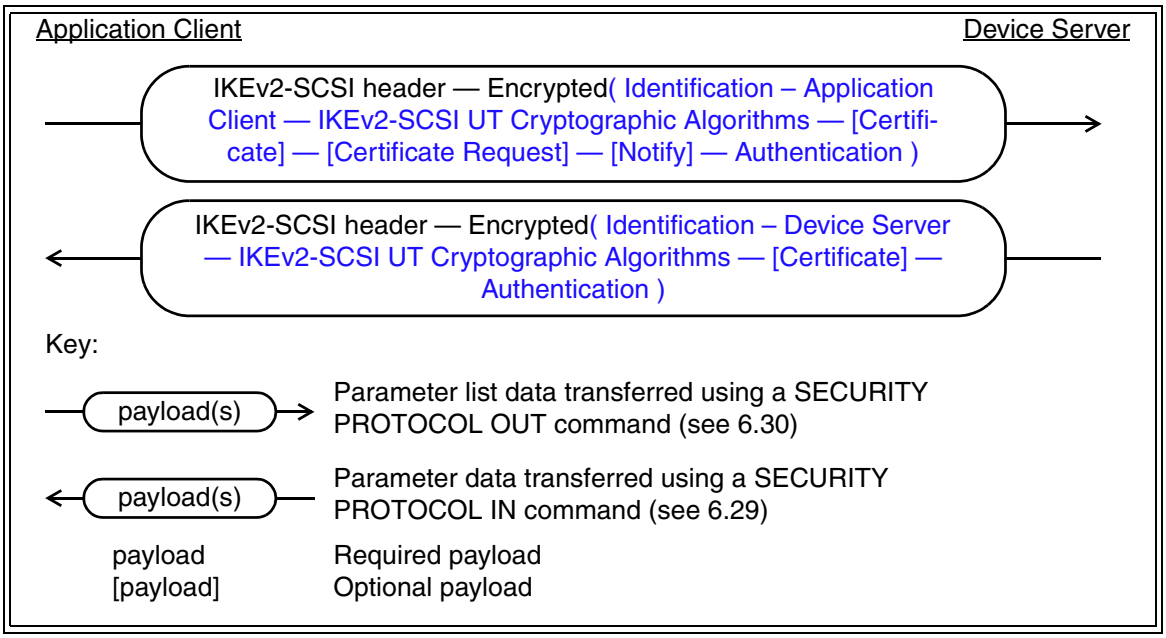

Figure x3 — IKEv2-SCSI Authentication step

{{The blue in figure x3 is intended for inclusion in SPC-4 and is used in conformance with the T10 Style Guide to assist viewers capable of rendering color to more easily see the encrypted payloads. It is not normative.}}

An Encrypted payload (see 7.7.3.5.10) contains all other Authentication step payloads that are protected using the cryptographic algorithms determined by the IKEv2-SCSI SA Cryptographic Algorithms payload descriptors in the Key Exchange step.

The Identification payloads (see 7.7.3.5.4) contain the identities to be authenticated. These identities are not required to be SCSI names or identifiers.

The IKEv2-SCSI UT Cryptographic Algorithms payloads (see 7.7.3.5.13) are used to select and agree on usage of the SA and the cryptographic algorithms used by the created SA.

The optional Certificate Request payload or payloads (see 7.7.3.5.5) allows an application client to request the delivery of a Certificate payload (see 7.7.3.5.5) in the parameter data for the Authentication step SECURITY PROTOCOL IN command (see 5.13.4.3).

The optional Notify payload (see 7.7.3.5.8) provides a means for the application client to inform the device server that this is the only SA being used between them, and that the device server should discard state for any other SAs created by the same application client.

The Authenticate payloads (see 7.7.3.5.6) authenticate not only the SA participants, but also the entire protocol sequence (e.g., the Authenticate payloads prevent a man-in-the-middle attack from succeeding).

Figure x4 shows the Delete operation (see 5.13.4.13). The Delete operation consists of a SECURITY PROTOCOL OUT command.

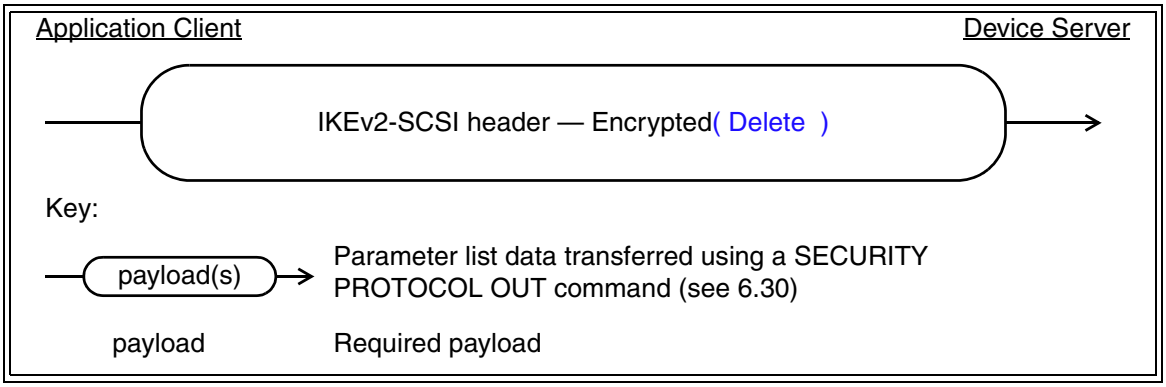

Figure x4 — IKEv2-SCSI Delete operation

{{The blue in figure x4 is intended for inclusion in SPC-4 and is used in conformance with the T10 Style Guide to assist viewers capable of rendering color to more easily see the encrypted payloads. It is not normative.}}

An Encrypted payload (see 7.7.3.5.10) contains the Delete payload that is protected using the cryptographic algorithms determined by the IKEv2-SCSI SA Cryptographic Algorithms payload descriptors in the Key Exchange step that was used to create the SA.

The Delete payload (see 7.7.3.5.9) specifies the SA to be deleted.

5.13.4.3 Handling of the Certificate Request payload and the Certificate payload

As detailed in this subclause, a Certificate Request payload (see 7.7.3.5.5) in one set of parameter data requests the delivery of a Certificate payload (see 7.7.3.5.5) in the next set of parameter data transferred. The purpose of this IKEv2-SCSI protocol construct is as follows:

a) Each SA participant is allowed to require the delivery of a Certificate payload by the other SA participant for use in authentication; and
b) Each Certificate Request payload indicates the trust anchors (see RFC 4306) list used by the device server or application client when PKI-based Authentication is being used with certificates that are not self signed (see RFC 3280).

The presence of one or more Certificate Request payloads in the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) parameter data indicates that the device server requires the application client to send a Certificate payload in the Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2).

The presence of one or more Certificate Request payloads in the Authentication step SECURITY PROTOCOL OUT command parameter list specifies that the application client requires the device server to send a Certificate payload in the Authentication step SECURITY PROTOCOL IN command (see 5.13.4.9.3).

If any Certificate payloads are included in the parameter data, the first Certificate payload shall contain the public key used to verify the Authentication payload. Additional Certificate payloads may be sent to assist in establishing a chain of trust from the certificate in the first payload to a trust anchor.

The application client and device server may use different authentication methods that require or do not require the use of Certificate payloads, and the presence or absence of Certificate Request payloads and Certificate payloads may vary in any of the commands described in this subclause.

5.13.4.4 Summary of IKEv2-SCSI shared keys nomenclature and shared key sizes

To facilitate use of the normative references made by this standard, the IKEv2-SCSI shared keys (see 3.1.w) are named as shown in table x1.

Table x1 — IKEv2-SCSI shared key names

Name	Description	SA parameter (see 3.1.103) that stores this shared key
Shared keys used only during Authentication step		
SK_pi	The shared key used in the Authentication payload (see 7.7.3.5.6) construction for the SECURITY PROTOCOL OUT parameter list in the Authentication step (see 5.13.4.9.2).	shall not be stored in any SA parameter
SK_pr	The shared key used in the Authentication payload construction for the SECURITY PROTOCOL IN parameter data in the Authentication step.	
Shared keys used during IKEv2-SCSI SA creation and deletion		
SK_ai	The shared key used to integrity check the Encrypted payload in the SECURITY PROTOCOL OUT parameter list in the: a) Authentication step; and b) IKEv2-SCSI Delete operation (see 5.13.4.13).	MGMT_DATA
SK_ar	The shared key used to integrity check the Encrypted payload (see 7.7.3.5.10) in the SECURITY PROTOCOL IN parameter data in the Authentication step (see 5.13.4.9.3).	
SK_ei	The shared key used to encrypt the Encrypted payload in the SECURITY PROTOCOL OUT parameter list in the: a) Authentication step; and b) IKEv2-SCSI Delete operation.	
SK_er	The shared key used to encrypt the Encrypted payload in the SECURITY PROTOCOL IN parameter data in the Authentication step.	
Shared keys used only after the IKEv2-SCSI SA has been created		
SK_d	The shared key material that is used as input to the KDF that generates the KEYMAT SA parameter (see 3.1.103) bytes for the SA.	KEY_SEED

The sizes of the shared keys are determined as shown in table x2.

Table x2 — IKEv2-SCSI shared key size determination

Name	Shared key size determination	
	Separate encryption and integrity checking [a, b]	**Combined mode encryption and integrity checking** [a, b]
SK_ai and SK_ar [c]	The shared key length shown in table x39 for the value in the ALGORITHM IDENTIFIER field of the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4) [b]	0
SK_er and SK_ei [c]	The shared key length shown in the KEY LENGTH field of the ENCR IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.2) [b]	The shared key length shown in the KEY LENGTH field of the ENCR IKEv2-SCSI cryptographic algorithm descriptor plus the number of salt bytes shown in table x35 (see 7.7.3.6.2) [b]
SK_pi and SK_pr [c]	The shared key length is equal to the PRF output length (see table x37) associated with the value in the ALGORITHM IDENTIFIER field of the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12)	
SK_d		

[a] The use of combined mode encryption and integrity checking is indicated by the AUTH_COMBINED value in the algorithm identifier field of the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4).

[b] For the shared keys used to create an SA, the algorithm descriptor is located in an IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12). For the shared keys used after the SA is created (i.e., the KEYMAT SA parameter), the algorithm descriptor is located in an IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13).

[c] To accommodate the need for two shared keys of the specified size, the shared key length shown is doubled for the purposes of shared key generation.

5.13.4.5 IKEv2-SCSI usage of pre-shared keys

If the Authentication payload (see 7.7.3.5.6) AUTHENTICATION DATA field contents are computed using pre-shared keys (e.g., if the applicable algorithm identifier is 00F9 0002h, Shared Key Message Integrity Code), then the following requirements apply in addition to those found in RFC 4306:

a) A pre-shared key shall be associated with one identity;
b) The same pre-shared key shall not be used to authenticate both an application client and a device server;
c) Use of the same pre-shared key for a group of application clients or a group of device servers is strongly discouraged, because it enables any member of the group to impersonate any other member;
d) The means for provisioning pre-shared keys are outside the scope of this standard;
e) The pre-shared keys may be provisioned as follows:
   A) At the time of manufacturing;
   B) During device or system initialization; or
   C) Any time thereafter;
f) The following requirements from RFC 4306 apply to the interfaces for provisioning pre-shared keys:
   A) ASCII strings of at least 64 bytes shall be supported;
   B) A null terminator shall not be added to any input before it is used as a pre-shared key;
   C) A hexadecimal ASCII encoding of the pre-shared key shall be supported; and

D) ASCII encodings other than hexadecimal may be supported. Support for any such encoding shall include specification of the algorithm for translating the encoding to a binary string as part of the interface;

and

g) Information about the size of the pre-shared key shall be stored at the same time that the pre-shared key is stored.

5.13.4.6 Constraints on skipping the Authentication step

In the Device Server Capabilities step (see 5.13.4.7), the parameter data returned by the SECURITY PROTOCOL IN command (see 7.7.2.3.2) contains the IKEv2-SCSI SA Creation Algorithms payload (see 7.7.3.5.11) that contains one or more SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptors (see 7.7.3.6.6) and one or more SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptors.

The device server permits the Authentication step to be omitted (see 5.13.4.1) if all of the following are true:

a) The ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE (see 7.7.3.6.6) in one of the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptors returned in the Device Server Capabilities step; and
b) The ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE in one of the SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptors returned in the Device Server Capabilities step.

The methods for configuring a device server to return SA_AUTH_NONE are outside the scope of this standard. Device servers shall not be manufactured to return SA_AUTH_NONE as an Authentication payload authentication algorithm type in the Device Server Capabilities step.

In the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2), the application client requests that the Authentication step be omitted by setting the ALGORITHM IDENTIFIER field to SA_AUTH_NONE in the SA_AUTH_OUT cryptographic algorithm descriptor and in the SA_AUTH_IN cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12).

To ensure adequate SA security, the application client should not select the SA_AUTH_NONE value as an Authentication payload authentication algorithm type unless:

a) An SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and an SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor from the Device Server Capabilities step indicates SA_AUTH_NONE availability; and
b) The application client is configured to omit the Authentication step.

NOTE x2 - When SA_AUTH_NONE is used, IKEv2-SCSI has no protection against any man-in-the-middle attacks. Enabling return of the SA_AUTH_NONE authentication algorithm type in the Device Capabilities step, and allowing an application client to select SA_AUTH_NONE in the Key Exchange step are administrative security policy decisions that absence of authentication is acceptable, and should only be made with a full understanding of the security consequences of the lack of authentication. Such decisions should only be made in situations where active attacks on IKEv2-SCSI are not of concern (e.g., direct attachment of a SCSI initiator device and a SCSI target device, or an end-to-end secure service delivery subsystem such as FCP over Fibre Channel secured by an end-to-end FC-SP SA).

5.13.4.7 Device Server Capabilities step

In the Device Server Capabilities step, the application client sends a SECURITY PROTOCOL IN command (see 6.29) with the SECURITY PROTOCOL field set to SA creation capabilities (i.e., zzh) and the SECURITY PROTOCOL SPECIFIC field set to 0101h.

The device server returns the SECURITY PROTOCOL IN parameter data specified by the SECURITY PROTOCOL SPECIFIC field (see 7.7.2.2) and the parameter data (see 7.7.2.3.2) contains an IKEv2-SCSI SA Creation Capabilities payload (see 7.7.3.5.11).

In the Device Server Capabilities step, the device server shall return parameter data containing the IKEv2-SCSI cryptographic algorithm descriptors (see 7.7.3.6) in at least one complete row shown in table x3.

Table x3 — Device Server Capabilities step parameter data requirements

IKEv2-SCSI cryptographic algorithm descriptor [a]					
ENCR (see 7.7.3.6.2)	**PRF (see 7.7.3.6.3)**	**INTEG (see 7.7.3.6.4)**	**D-H (see 7.7.3.6.5)**	**SA_AUTH_OUT (see 7.7.3.6.6)**	**SA_AUTH_IN (see 7.7.3.6.6)**
The ALGORITHM IDENTIFIER field set to 0001 000Ch (i.e., AES-CBC) and the KEY LENGTH field set to 0010h	The ALGORITHM IDENTIFIER field set to 0002 0005h (i.e., IKEv2-use based on SHA-256)	The ALGORITHM IDENTIFIER field set to 0003 000Ch (i.e., AUTH_HMAC_ SHA2_ 256_128)	The ALGORITHM IDENTIFIER field set to 0004 000Eh (i.e., 2 048-bit MODP group (finite field D-H))	The ALGORITHM IDENTIFIER field set to 00F9 0001h (i.e., RSA Digital Signature)	The ALGORITHM IDENTIFIER field set to 00F9 0001h (i.e., RSA Digital Signature)
The ALGORITHM IDENTIFIER field set to 0001 000Ch (i.e., AES-CBC) and the KEY LENGTH field set to 0020h	The ALGORITHM IDENTIFIER field set to 0002 0007h (i.e., IKEv2-use based on SHA-512)	The ALGORITHM IDENTIFIER field set to 0003 000Eh (i.e., AUTH_HMAC_ SHA2_ 512_256)	The ALGORITHM IDENTIFIER field set to 0004 0015h (i.e., 521-bit random ECP group)	The ALGORITHM IDENTIFIER field set to 00F9 000Bh (i.e., ECDSA with SHA-512 on the P-521 curve)	The ALGORITHM IDENTIFIER field set to 00F9 000Bh (i.e., ECDSA with SHA-512 on the P-521 curve)

[a] The IKEv2-SCSI cryptographic algorithm descriptors shown in all the columns in at least one row in this table shall be returned in the Device Server Capabilities step parameter data.

In the Device Server Capabilities step, the device server shall return parameter data containing one SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE and one SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE if any of the following are true:

a) The ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE in one of the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptors returned in the Device Server Capabilities step; or
b) The ALGORITHM IDENTIFIER field is set to SA_AUTH_NONE in one of the SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptors returned in the Device Server Capabilities step.

The device server capabilities returned in the SECURITY PROTOCOL IN parameter data may be changed at any time via interfaces that are outside the scope of this standard, however, such changes shall not take effect until at least one application client has been notified of the new capabilities via the parameter data returned by the Device Server Capabilities step SECURITY PROTOCOL IN command. Management applications may ensure that their device server capabilities changes to take effect by sending a Device Server Capabilities step SECURITY PROTOCOL IN command to the device server after the changes have been made.

When the device server capabilities change (i.e., upon completion of the processing for a Device Server Capabilities step SECURITY PROTOCOL IN command that reported changed information in its parameter data), the device server shall establish a unit attention condition for the initiator port associated with every I_T nexus except

the I_T nexus on which the Device Server Capabilities step SECURITY PROTOCOL IN command was received (see SAM-4), with the additional sense code set to SA CREATION CAPABILITIES DATA HAS CHANGED.

{{SA CREATION CAPABILITIES DATA HAS CHANGED is a new additional sense code.}}

The Device Server Capabilities step participates in the negotiation to skip the Authentication step as described in 5.13.4.6.

NOTE x3 - The Device Server Capabilities step has no IKEv2 exchange equivalent in RFC 4306. This step replaces most of IKEv2's negotiation by having the application client obtain the supported capabilities from the device server.

5.13.4.8 IKEv2-SCSI Key Exchange step

5.13.4.8.1 Overview

The Key Exchange step consists of a Diffie-Hellman key exchange with nonces (see RFC 4306) and is accomplished as follows:

1) A SECURITY PROTOCOL OUT command (see 5.13.4.8.2);
2) A SECURITY PROTOCOL IN command (see 5.13.4.8.3); and
3) Key exchange completion (see 5.13.4.8.4)

NOTE x4 - The Key Exchange step corresponds to the IKEv2 IKE_SA_INIT exchange in RFC 4306, except that determination of device server capabilities has been moved to the Device Server Capabilities step.

5.13.4.8.2 Key Exchange step SECURITY PROTOCOL OUT command

To send its key exchange message to the device server, the application client sends a SECURITY PROTOCOL OUT command (see 6.30) with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) and the SECURITY PROTOCOL SPECIFIC field set to 0102h. The parameter list consists of an IKEv2-SCSI header (see 7.7.3.4) and the following:

1) An IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14);
2) An IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12);
3) If the Authentication step is skipped (see 5.13.4.1), an IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13);
4) A Key Exchange payload (see 7.7.3.5.3); and
5) A Nonce payload (see 7.7.3.5.7).

The IKEv2-SCSI Timeout Values payload contains the inactivity timeouts that apply to this IKEv2-SCSI SA creation transaction (see 3.1.o) and the SA (see 3.1.119) that is created.

The IKEv2-SCSI SA Cryptographic Algorithms payload selects the cryptographic algorithms from among those returned in the Device Server Capabilities step (see 5.13.4.7) to be used in the creation of the SA.

If the Authentication step is skipped, the IKEv2-SCSI UT Cryptographic Algorithms payload contains the following information about the SA to be created:

a) The cryptographic algorithms selected by the application client from among those returned in the Device Server Capabilities step; and
b) The usage data (see 7.7.3.5.13) that is specific to the SA.

If the application client is unable to select a set of algorithms that are appropriate for the intended creation and usage of the SA, the application client should not perform the Key Exchange step to request the creation of an SA.

IKEv2-SCSI SA Cryptographic Algorithms payload error checking requirements that ensure a successful negotiation of SA creation algorithms are described in 7.7.3.5.12 and 7.7.3.6.

IKEv2-SCSI UT Cryptographic Algorithms payload error checking requirements that ensure a successful negotiation of SA creation algorithms are described in 7.7.3.5.13 and 7.7.3.6.

The Key Exchange payload contains the application client's Diffie-Hellman value.

The Nonce payload contains the application client's random nonce (see 3.1.95).

5.13.4.8.3 Key Exchange step SECURITY PROTOCOL IN command

If the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) completes with GOOD status, the application client sends a SECURITY PROTOCOL IN command (see 6.29) with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) and the SECURITY PROTOCOL SPECIFIC field set to 0102h to obtain the device server's key exchange message.

The parameter data returned by the device server in response to the SECURITY PROTOCOL IN command shall contain an IKEv2-SCSI header (see 7.7.3.4) and the following:

1) An IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12);
2) If the Authentication step is skipped (see 5.13.4.1), an IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13);
3) A Key Exchange payload (see 7.7.3.5.3);
4) A Nonce payload (see 7.7.3.5.7); and
5) Zero or more Certificate Request payloads (see 5.13.4.3).

As part of processing of the Key Exchange step SECURITY PROTOCOL IN command, the device server shall:

a) Associate the SECURITY PROTOCOL IN command to the last Key Exchange step SECURITY PROTOCOL OUT command received on the I_T_L nexus. If the device server is maintaining state for at least one IKEv2-SCSI CCS and the device server is unable to establish this association, then:
   A) The SECURITY PROTOCOL IN command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS; and
   B) The device server shall continue the IKEv2-SCSI CCS.

   If the device is not maintaining state for at least one IKEv2-SCSI CSS, then the SECURITY PROTOCOL IN 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.
b) Return the device server's SAI in the IKEv2-SCSI header IKE_SA DEVICE SERVER SAI field;
c) Return the IKEv2-SCSI SA Cryptographic Algorithms payload containing:
   A) The SA creation cryptographic algorithms supplied by the application client in the Key Exchange step SECURITY PROTOCOL OUT command parameter list; and
   B) The device server's SAI (see 3.1.120) in the SAI field (see 7.7.3.5.12);
d) If the Authentication step is skipped, return the IKEv2-SCSI UT Cryptographic Algorithms payload containing:
   A) The SA usage cryptographic algorithms supplied by the application client in the Key Exchange step SECURITY PROTOCOL OUT command parameter list; and
   B) The device server's SAI in the SAI field (see 7.7.3.5.13);
e) Return information about the completed the Diffie-Hellman exchange with the Key Exchange payload; and
f) Return device server's random nonce (see 3.1.95) in the Nonce payload.

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

If the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) completes with GOOD status, the application client should copy the device server's SAI from the SAI field in the IKEv2-SCSI SA Cryptographic Algorithms payload to the state it is maintaining for the IKEv2-SCSI CCS.

The application client should compare the other fields in the IKEv2-SCSI SA Cryptographic Algorithms payload and the IKEv2-SCSI UT Cryptographic Algorithms payload, if any, to the values sent in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2). If the application client detects differences in the contents of the payloads other than in the SAI field, the application client should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

5.13.4.8.4 Key Exchange step completion

Before completing the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) with GOOD status the device server shall complete the Key Exchange step as described in this subclause.

Upon receipt of GOOD status for the Key Exchange step SECURITY PROTOCOL IN command the application client should complete the Key Exchange step as described in this subclause.

If the Key Exchange step does not end with the IKEv2-SCSI CCS being abandoned (see 5.13.4.12), the contents of the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.6) the IKEv2-SCSI SA Cryptographic Algorithms payload specify how the shared key exchanged by the Key Exchange step SECURITY PROTOCOL OUT command and the Key Exchange step SECURITY PROTOCOL IN command is used to generate additional shared keys as follows:

a) If the ALGORITHM IDENTIFIER field in both descriptors contain SA_AUTH_NONE, then the SA participants generate the SA, including the generation of the shared keys used for SA management (e.g., SA creation and management) and the shared keys used by the created SA as defined in 5.13.4.11; or
b) If the ALGORITHM IDENTIFIER field in both descriptors contain a value other than SA_AUTH_NONE, then the SA participants generate shared keys (see 5.13.4.10.3) for the following:
   A) Seeding the Authentication step generation of the shared keys used by the created SA; and
   B) SA creation and management.

5.13.4.8.5 After the Key Exchange step

Processing of the IKEv2-SCSI CCS subsequent to completion of the Key Exchange step (see 5.13.4.8.4) depends on the contents of the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.6) the IKEv2-SCSI SA Cryptographic Algorithms payload as follows:

a) If the ALGORITHM IDENTIFIER field in both descriptors contain SA_AUTH_NONE, then processing of the IKEv2-SCSI CCS is finished and the generated SA is ready for use; or
b) If the ALGORITHM IDENTIFIER field in both descriptors contain a value other than SA_AUTH_NONE, then processing of the IKEv2-SCSI CCS continues as follows:
   A) The Authentication step is performed (see 5.13.4.9); and
   B) The SA participants generate the SA, including the generation of the shared keys used for SA management and the shared keys used by the created SA as defined in 5.13.4.11.

5.13.4.9 IKEv2-SCSI Authentication step

5.13.4.9.1 Overview

The Authentication step performs the following functions:

a) authenticates both the application client and the device server;
b) protects the previous steps of the protocol; and
c) cryptographically binds the authentication and the previous steps to the created SA.

The Authentication step is accomplished as follows:

1) A SECURITY PROTOCOL OUT command (see 5.13.4.9.2); and
2) A SECURITY PROTOCOL IN command (see 5.13.4.9.3).

The parameter data for both commands shall be encrypted and integrity protected using the algorithms and keys determined in the Key Exchange step (see 5.13.4.8.4).

NOTE x5 - The Authentication step corresponds to the IKEv2 IKE_AUTH exchange in RFC 4306.

5.13.4.9.2 Authentication step SECURITY PROTOCOL OUT command

To send its authentication information to the device server, the application client sends a SECURITY PROTOCOL OUT command (see 6.30) with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) and the SECURITY PROTOCOL SPECIFIC field set to 0103h. The parameter data consists of the IKEv2-SCSI header (see 7.7.3.4) and an Encrypted payload (see 7.7.3.5.10) that:

a) Is integrity checked and encrypted as follows:
   A) Using separate algorithms as follows:
      a) Integrity checked using the following:
         A) The algorithm specified by the INTEG IKEv2-SCSI algorithm descriptor (see 7.7.3.6.4) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12); and
         B) The SK_ai shared key (see 5.13.4.4);
      and
      b) Is Encrypted using the following:
         A) The algorithm specified by the ENCR IKEv2-SCSI algorithm descriptor (see 7.7.3.6.2) in the IKEv2-SCSI SA Cryptographic Algorithms payload; and
         B) The SK_ei shared key (see 5.13.4.4);
   or
   B) Using a single algorithm that uses the following (i.e., if the INTEG IKEv2-SCSI algorithm descriptor indicates AUTH_COMBINED):
      a) The algorithm specified by the ENCR IKEv2-SCSI algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload; and
      b) The SK_ei shared key with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4);
   and
b) Contains the following:
   1) An Identification – Application Client payload (see 7.7.3.5.4);
   2) An IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13);
   3) Zero or more Certificate payloads (see 5.13.4.3);
   4) Zero or more Certificate Request payloads (see 5.13.4.3);
   5) Zero or one Notify payload (see 7.7.3.5.8); and
   6) An Authentication payload (see 7.7.3.5.6).

Before performing any checks of data contained in the Encrypted payload, the device server shall validate the SECURITY PROTOCOL OUT command parameter data as follows:

a) The device server shall compare the IKE_SA APPLICATION CLIENT SAI field and the IKE_SA DEVICE SERVER SAI field in the IKEv2-SCSI header to the SAI values it is maintaining for the IKEv2-SCSI CCS, if any, being maintained for the I_T_L nexus on which the SECURITY PROTOCOL OUT command was received as described in 7.7.3.4; and
b) The device server shall decrypt and check the integrity of the Encrypted payload as described in 7.7.3.5.10.5.

Errors detected during the decryption and integrity checking of the Encrypted payload shall be handled as described in 7.7.3.8.2.

In the SECURITY PROTOCOL OUT command parameter list, the application client:

a) Sends information about the SA usage and cryptographic algorithms;
b) Sends its identity with the ID payload;
c) Sends information proving its knowledge of the secret corresponding to its identity in the Authentication payload; and
d) Integrity protects the Key Exchange step and the Authentication step using the Authentication payload.

The application client may use the Notify payload to send an initial contact notification to the device server. If sent, the initial contact notification specifies that the application client has no stored state for any SAs with the device server other than the SA that is being created.

In response to receipt of an initial contact notification, the device server should delete all other SAs that were authenticated with a SECURITY PROTOCOL OUT command that contained the same Identification - Application Client payload data as that which is present in the SECURITY PROTOCOL OUT command that the device server is processing.

If the device server deletes other SAs in response to an initial contact notification, it shall do so only after the successful completion of the Authentication step SECURITY PROTOCOL OUT command. If an error occurs during the Authentication SECURITY PROTOCOL OUT command, the device server shall ignore the initial contact notification.

If the device server is unable to proceed with SA creation for any reason (e.g., the verification of the Authentication payload fails), the SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to ABORTED COMMAND, and the additional sense code set to an appropriate value. The additional sense code AUTHENTICATION FAILED shall be used when verification of the Authentication payload fails, or when authentication fails for any other reason.

{{AUTHENTICATION FAILED is a new additional sense code}}

5.13.4.9.3 Authentication step SECURITY PROTOCOL IN command

To obtain the device server's authentication information, the application client sends a SECURITY PROTOCOL IN command (see 6.29) with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) and the SECURITY PROTOCOL SPECIFIC field set to 0103h. The parameter data consists of the IKEv2-SCSI header (see 7.7.3.4) and an Encrypted payload (see 7.7.3.5.10) that:

a) Is integrity checked using the following:
   A) The algorithm specified by the INTEG IKEv2-SCSI algorithm descriptor (see 7.7.3.6.4) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12); and
   B) The SK_ar shared key (see 5.13.4.4);

b) Is Encrypted using the following:
   A) The algorithm specified by the ENCR IKEv2-SCSI algorithm descriptor (see 7.7.3.6.2) in the IKEv2-SCSI SA Cryptographic Algorithms payload; and
   B) The SK_er shared key (see 5.13.4.4);
   and
c) Contains the following:
   1) An Identification – Device Server payload (see 7.7.3.5.4);
   2) An IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13);
   3) Zero or more Certificate payloads (see 5.13.4.3); and
   4) An Authentication payload (see 7.7.3.5.6).

In the SECURITY PROTOCOL IN parameter data, the device server:

a) Confirms information about the SA usage and cryptographic algorithms;
b) Sends its identity with the ID payload;
c) Authenticates its identity; and
d) Protects the integrity of the prior step messages using the Authentication payload.

Before returning GOOD status for the SECURITY PROTOCOL IN command, the device server shall generate the SA as described in 5.13.4.11.

The application client should verify the Authentication payload as described in 7.7.3.5.6. The Certificate payload(s) are used as part of this verification for PKI-based authentication. If the Authentication payload is verified and no other error occurs the application client should generate the SA as described in 5.13.4.11.

If the application client is unable to proceed with SA creation for any reason (e.g., the verification of the Authentication payload fails), the application client should:

a) Not use the SA for any additional activities; and
b) Notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

The application client should compare the other fields in the IKEv2-SCSI UT Cryptographic Algorithms payload to the values sent in the Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2). If the application client detects differences in the contents of the payloads other than in the SAI field, the application client should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

5.13.4.10 Generating shared keys

5.13.4.10.1 Overview

If the Authentication step is skipped (see 5.13.4.1), then shared key generation is performed as described in 5.13.4.10.2 and is summarized as follows:

a) The shared keys for SA management (e.g., SA creation and deletion) are generated at the same time as the shared keys used by the created SA; and
b) All the shared keys are generated during completion of the Key Exchange step (see 5.13.4.8.4).

If the Authentication step is processed (i.e., not skipped), then shared key generation is performed as described in 5.13.4.10.3 and is summarized as follows:

a) The shared keys for SA management (e.g., SA creation and deletion) are generated during the completion of the Key Exchange step (see 5.13.4.8.4); and
b) The shared keys used by the created SA are generated during SA generation (see 5.13.4.11).

Regardless of when the SA management shared keys (i.e., the shared keys used for SA creation and deletion) and shared keys used by the created SA are generated (see 5.13.4.4), the organization of the shared keys depends on the type of encryption and integrity checking algorithm being used as follows:

a) If an encryption algorithm that requires separate integrity checking is used, then separate shared keys are generated for each algorithm; or
b) If an encryption algorithm that includes integrity checking is used (i.e., if the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4) contains AUTH_COMBINED), then no shared keys are generated for the integrity checking algorithm but additional key material is generated to act as a salt (see table x35 in 7.7.3.6.2) for the combined mode encryption algorithm.

5.13.4.10.2 Generating shared keys when the Authentication step skipped

If the Authentication step is skipped (see 5.13.4.1), then the shared keys for SA management are generated at the same time as the shared keys for use by the created SA.

As part of completing the Key Exchange step (see 5.13.4.8.4), the SA participants generate all necessary shared keys as follows:

1) Generate SKEYSEED (see RFC 4306) as described in 5.13.4.10.4;
2) Generate the shared keys used for SA management as described in 5.13.4.10.5;
3) As part of generating the SA (see 5.13.4.11) (i.e., as part of completing the Key Exchange step as described in 5.13.4.8.4), generate the shared keys for use by the created SA as described in 5.13.4.10.6.

5.13.4.10.3 Generating shared keys when the Authentication step processed

If the Authentication step is not skipped (see 5.13.4.1), then:

1) The shared keys for SA management are generated during completion of the Key Exchange step (see 5.13.4.8.4) as follows;
   1) Generate SKEYSEED (see RFC 4306) as described in 5.13.4.10.4; and
   2) Generate the shared keys used for SA management as described in 5.13.4.10.5;
   and
2) The shared keys for use by the created SA are generated during SA generation (see 5.13.4.11), near the end of processing for the Authentication step (see 5.13.4.9) as described in 5.13.4.10.6.

5.13.4.10.4 Initializing shared key generation

5.13.4.10.4.1 Initializing for SA creation shared key generation

The SA parameters (see 3.1.103) are initialized for the KDF function used to generate SA creation shared keys as follows:

1) Generate the input to the PRF function by performing the last steps of the key exchange algorithm selected by the ALGORITHM IDENTIFIER field in the D-H IKEv2-SCSI algorithm descriptor (see 7.7.3.6.5) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) using at least the contents of one of the following fields as inputs to those last steps:
   A) The KEY EXCHANGE DATA field in the Key Exchange payload (see 7.7.3.5.3) in the Key Exchange SECURITY PROTOCOL OUT command (see 5.13.4.8.2) parameter data; and
   B) The KEY EXCHANGE DATA field in the Key Exchange payload in the Key Exchange SECURITY PROTOCOL IN command (see 5.13.4.8.3) parameter data;
2) Generate SKEYSEED (see RFC 4306) using the output from step 1) and the PRF selected by the ALGORITHM IDENTIFIER field in the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload;

3) Store the generated SKEYSEED value in the KEY_SEED SA parameter;
4) Store the contents of the IKE_SA APPLICATION CLIENT SAI field from the IKEv2-SCSI Header (see 7.7.3.4) from the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the AC_SAI SA parameter;
5) Store the contents of the IKE_SA DEVICE SERVER SAI field from the IKEv2-SCSI Header (see 7.7.3.4) from the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the DS_SAI SA parameter;
6) Store the contents of the NONCE DATA field from the Nonce payload (see 7.7.3.5.7) from the parameter data for the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) in the AC_NONCE SA parameter; and
7) Store the contents of the NONCE DATA field from the Nonce payload (see 7.7.3.5.7) from the parameter data for the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the DS_NONCE SA parameter.

5.13.4.10.4.2 Initializing for generation of shared keys used by the created SA

The SA parameters (see 3.1.103) are initialized for the KDF function used to generate the shared keys that will be used by the created SA as follows:

1) Store the SK_d value that was generated along with the other shared keys used in SA creation (see 5.13.4.10.5) in the KEY_SEED SA parameter;
2) Store the contents of the IKE_SA APPLICATION CLIENT SAI field from the IKEv2-SCSI Header (see 7.7.3.4) from the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the AC_SAI SA parameter;
3) Store the contents of the IKE_SA DEVICE SERVER SAI field from the IKEv2-SCSI Header (see 7.7.3.4) from the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the DS_SAI SA parameter;
4) Store the contents of the NONCE DATA field from the Nonce payload (see 7.7.3.5.7) from the parameter data for the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) in the AC_NONCE SA parameter; and
5) Store the contents of the NONCE DATA field from the Nonce payload (see 7.7.3.5.7) from the parameter data for the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the DS_NONCE SA parameter.

5.13.4.10.5 Generating shared keys used for SA management

The shared keys used for SA management (e.g., SA creation and deletion) are generated using the SKEYSEED generated during initialization (see 5.13.4.10.4) and the steps described in this subclause.

Which shared keys are generated for SA management depends on:

a) Whether the encryption algorithm includes integrity checking as indicated by the contents of the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4) of the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12), and
b) Whether the Authentication step is skipped (see 5.13.4.1).

Using the contents of the initialized SA parameters (see 5.13.4.10.4.1), the INTEG ALGORITHM IDENTIFIER field, and the KDF selected by the ALGORITHM IDENTIFIER field in the PRF IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload to generate the following shared keys (see 5.13.4.4) in the order shown:

a) If the INTEG ALGORITHM IDENTIFIER field does not contain AUTH_COMBINED, then generate the following shared keys (see 5.13.4.4):
   A) If the Authentication step is not skipped, generate the following shared keys:

1) SK_d;
2) SK_ai;
3) SK_ar;
4) SK_ei;
5) SK_er;
6) SK_pi; and
7) SK_pr;

or

B) If the Authentication step is skipped, then generate the following shared keys:
1) SK_d;
2) SK_ai;
3) SK_ar;
4) SK_ei; and
5) SK_er.

or

b) If the INTEG ALGORITHM IDENTIFIER field contains AUTH_COMBINED, then generate the following shared keys:
A) If the Authentication step is not skipped, generate the following shared keys:
1) SK_d;
2) SK_ei with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4);
3) SK_er with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4);
4) SK_pi; and
5) SK_pr;

or

B) If the Authentication step is skipped, then generate the following shared keys:
1) SK_d;
2) SK_ei with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4); and
3) SK_er with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4).

How to determine the sizes of the shared keys to be generated is summarized in table x2 (see 5.13.4.4).

5.13.4.10.6 Generating shared keys for use by the created SA

As part of completing the Authentication step and generating the SA (see 5.13.4.11), the SA participants initialize the SA parameters for performing a KDF (see 5.13.4.10.4.2), and use the KDF selected by the ALGORITHM IDENTIFIER field in the PRF IKEv2-SCSI cryptographic algorithm descriptor in the Key Exchange step IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) to generate the following shared keys (see 5.13.4.4) in the order shown and store them in the KEYMAT SA parameter:

a) If the ALGORITHM IDENTIFIER field in the ENCR IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.2) contains ENCR_NULL in the IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13), then generate the following shared keys:
1) SK_ai;
2) SK_ar;

and

b) If the ALGORITHM IDENTIFIER field in the ENCR IKEv2-SCSI cryptographic algorithm descriptor does not contain ENCR_NULL in the IKEv2-SCSI UT Cryptographic Algorithms payload, then generate the following shared keys:
A) If the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4) does not contain AUTH_COMBINED in the IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13), then generate the following shared keys:
1) SK_ai;
2) SK_ar;
3) SK_ei; and

4) SK_er;

or

B) If the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor contains AUTH_COMBINED in the IKEv2-SCSI UT Cryptographic Algorithms payload, then generate the following shared keys:
   1) SK_ei with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4); and
   2) SK_er with additional salt bytes as described in 5.13.4.10.1 and table x2 (see 5.13.4.4).

How to determine the sizes of the shared keys to be generated is summarized in table x2 (see 5.13.4.4).

NOTE x6 - The SKEYSEED described in this subclause is different from the SKEYSEED SA parameter (see 3.1.103).

5.13.4.11 IKEv2-SCSI SA generation

Depending on whether or not the Authentication step was skipped (see 5.13.4.1), the SA participants shall generate shared keys as described in 5.13.4.10.

The SA participants shall initialize the SA parameters (see 3.1.103) as follows:

1) KEYMAT shall be set as follows:
   A) If the Authentication step is skipped, KEYMAT shall be set as described in 5.13.4.10.2; or
   B) If the Authentication step is processed, KEYMAT shall be set as described in 5.13.4.10.3;

   and
2) Then, the other SA parameters shall be set as follows:
   A) AC_SAI shall be set to the value in the IKE_SA APPLICATION CLIENT SAI field in the IKEv2-SCSI header (see 7.7.3.4) in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2);
   B) DS_SAI shall be set to the value in the IKE_SA DEVICE SERVER SAI field in the IKEv2-SCSI header in the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3);
   C) TIMEOUT shall be set to the IKEV2-SCSI SA INACTIVITY TIMEOUT field in the IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14) in the Key Exchange step SECURITY PROTOCOL OUT command;
   D) KDF_ID shall be set to the value in the ALGORITHM IDENTIFIER field in the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12);
   E) AC_SQN shall be set to one;
   F) DS_SQN shall be set to one;
   G) AC_NONCE shall be set to zero;
   H) DS_NONCE shall be set to zero;
   I) KEY_SEED shall be set to zero;
   J) USAGE_TYPE shall be set to the value in the SA TYPE field in the IKEv2-SCSI UT Cryptographic Algorithms payload (see 7.7.3.5.13) in the Key Exchange step SECURITY PROTOCOL OUT command;
   K) USAGE_DATA shall contain at least the following values from of the usage data field in the IKEv2-SCSI UT Cryptographic Algorithms payload in the Key Exchange step SECURITY PROTOCOL OUT command:
      A) The USAGE DATA field;
      B) The ALGORITHM IDENTIFIER field (see 7.7.3.6) in the IKEv2-SCSI UT Cryptographic Algorithm descriptor (see 7.7.3.6) for the ENCR algorithm type;
      C) The KEY LENGTH field (see 7.7.3.6.2) in the ALGORITHM ATTRIBUTES field in the IKEv2-SCSI UT Cryptographic Algorithm descriptor for the ENCR algorithm type; and
      D) The ALGORITHM IDENTIFIER field (see 7.7.3.6) in the IKEv2-SCSI UT Cryptographic Algorithm descriptor for the INTEG algorithm type;

      and
   L) MGMT_DATA shall contain at least the following values:

a) From the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step SECURITY PROTOCOL OUT command:
   A) The ALGORITHM IDENTIFIER field (see 7.7.3.6) in the IKEv2-SCSI Cryptographic Algorithm descriptor (see 7.7.3.6) for the ENCR algorithm type;
   B) The KEY LENGTH field (see 7.7.3.6.2) in the ALGORITHM ATTRIBUTES field in the IKEv2-SCSI Cryptographic Algorithm descriptor for the ENCR algorithm type;
   C) The ALGORITHM IDENTIFIER field (see 7.7.3.6) in the IKEv2-SCSI Cryptographic Algorithm descriptor for the INTEG algorithm type;
b) From the shared keys generated for SA management (see 5.13.4.10.5), the following shared keys and salt bytes;
   A) The value of SK_ai, if any;
   B) The value of SK_ar, if any;
   C) The value of SK_ei, with additional salt bytes, if any; and
   D) The value of SK_er, with additional salt bytes, if any;
   and
c) The next value of the message id field in the IKEv2-SCSI header.

NOTE x7 - The inclusion of the algorithm identifiers and key length in MGMT_DATA SA parameter enables the SA to apply the same encryption and integrity algorithms that IKEv2-SCSI negotiated to future IKEv2-SCSI SECURITY PROTOCOL OUT commands.

5.13.4.12 Abandoning an IKEv2-SCSI CCS

The occurrence of errors in either the application client or the device server may require that an IKEv2-SCSI CCS (see 3.1.c) be abandoned.

A device server shall indicate that it has abandoned an IKEv2-SCSI CCS, if any, by terminating an IKEv2-SCSI CCS command (see 5.13.4.1) received on the I_T_L nexus for which the IKEv2-SCSI CCS state is being maintained with any combination of status and sense data other than those shown in table x4.

Table x4 — IKEv2-SCSI command terminations that do not abandon the CCS, if any

IKEv2-SCSI CCS Command	Status (Sense Key)	Additional Sense Code	Description
SECURITY PROTOCOL OUT SECURITY PROTOCOL IN	GOOD (n/a)	n/a	Indicates IKEv2-SCSI CCS is progressing as described in this standard
Key Exchange step SECURITY PROTOCOL OUT (see 5.13.4.8.2)	CHECK CONDITION (ABORTED COMMAND)	CONFLICTING SA CREATION REQUEST	At least one IKEv2-SCSI CCS is already active, and attempts to start another are blocked until the first CCS completes
SECURITY PROTOCOL OUT SECURITY PROTOCOL IN	CHECK CONDITION (NOT READY)	LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS	Device server is busy processing another command in the IKEv2-SCSI CCS associated with this I_T_L nexus or a different IKEv2-SCSI CCS associated with a different I_T_L nexus
SECURITY PROTOCOL IN	CHECK CONDITION (ILLEGAL REQUEST)	INVALID FIELD IN CDB	Incorrect SECURITY PROTOCOL IN CDB format
SECURITY PROTOCOL OUT		UNABLE TO DECRYPT PARAMETER LIST	Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2) for which the device server is unable to decrypt the Encrypted payload (see 7.7.3.5.10) or the integrity check fails
SECURITY PROTOCOL OUT		SA CREATION PARAMETER VALUE REJECTED	To adapt to possible denial of service attacks, a condition for which the optimal response includes an additional sense code of SA CREATION PARAMETER VALUE INVALID and the abandonment of the CCS is not causing the CCS to be abandoned

{{CONFLICTING SA CREATION REQUEST, LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS, UNABLE TO DECRYPT PARAMETER LIST, SA CREATION PARAMETER VALUE INVALID, and SA CREATION PARAMETER VALUE REJECTED are a new additional sense codes.}}

As part of abandoning an IKEv2-SCSI CCS, the device server shall:

a) Discard all maintained state (see 5.13.4.1); and

b) Prepare to allow a future Key Exchange step SECURITY PROTOCOL OUT command received on any I_T_L nexus to start a new IKEv2-SCSI CCS.

After a device server abandons an IKEv2-SCSI CCS, the device server shall respond to all new IKEv2-SCSI protocol commands as if an IKEv2-SCSI CCS had never been started.

An application client should not abandon an IKEv2-SCSI CCS when the next command in the CCS is a SECURITY PROTOCOL IN command. Instead, the application client should send the appropriate SECURITY PROTOCOL IN command and then abandon the IKEv2-SCSI CCS.

An application client should specify that it has abandoned an IKEv2-SCSI CCS by sending an IKEv2-SCSI Delete operation (see 5.13.4.13) with application client SAI and device server SAI information that matches that of the IKEv2-SCSI CCS being abandoned.

5.13.4.13 Deleting an IKEv2-SCSI SA

When an SA is deleted, both sets of SA parameters (see 5.13.2.2) are deleted as follows:

1) The application client uses the information in its SA parameters to prepare an IKEv2-SCSI Delete operation that requests deletion of the device server's SA parameters;
2) The application client deletes its SA parameters and any associated data;
3) The application client sends the IKEv2-SCSI Delete operation prepared in step 1) to the device server;
4) In response to the IKEv2-SCSI Delete operation, the device server deletes its SA parameters and any associated data.

The IKEv2-SCSI Delete operation is a SECURITY PROTOCOL OUT command with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., xxh) and the SECURITY PROTOCOL SPECIFIC field set to 0104h. The parameter data consists of the IKEv2-SCSI header (see 7.7.3.4) and an Encrypted payload (see 7.7.3.5.10) that contains the one Delete payload (see 7.7.3.5.9).

The Delete payload should conform to the requirements described in 7.7.3.5.9.

A valid Delete operation SECURITY PROTOCOL OUT command shall be processed regardless of whether or not IKEv2-SCSI CCS state is being maintained for the I_T_L nexus on which the command is received.

The application client SAI and device server SAI information may identify an IKEv2-SCSI CCS for which state is being maintained for the I_T_L nexus on which the command is received (see 7.7.3.5.10).

5.13.5 Security progress indication

The cryptographic calculations required by some security protocols are capable of consuming significant amounts of time in the device server. While a cryptographic security calculations are in progress unless an error has occurred, a device server shall respond to a REQUEST SENSE command by returning parameter data containing sense data with the sense key set to NOT READY and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS with the sense key specific bytes set for progress indication (i.e., a PROGRESS INDICATION field indicating the progress of the device server in performing the necessary cryptographic calculations).

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

The device server shall not use the progress indication to report the detailed progress of cryptographic computations that take a variable amount of time based on their inputs. The device server may use the progress indication to report synthetic progress that does not reveal the detailed progress of the computation (e.g., divide a constant

expected time for the computation by 10 and advance the progress indication in 10% increments based solely on the time).

The requirements in this subclause apply to implementations of Diffie-Hellman computations and operations involving any keys (e.g., RSA) that optimize operations on large numbers based on the values of inputs (e.g., a computational step may be skipped when a bit or set of bits in an input is zero). A progress indication that advances based on the computation structure (e.g., count of computational steps) may reveal the time taken by content-dependent portions of the computation, and reveal information about the inputs.

When cryptographic calculations are in progress, the sense data specified in this subclause shall be returned in response to a REQUEST SENSE command.

{{New model subclause text ends here. Additions/deletions markups resume.}}

5.13.6~~5.13.4~~ Security algorithm codes

Table 51 lists the security algorithm codes used in security protocol parameter data.

Table 51 — Security algorithm codes (page 1 of 2)

Code [a]	Description	Reference
Encryption algorithms		
0001 000Bh	ENCR_NULL	7.7.3.6.2
0001 000Ch	AES-CBC	RFC 3602
0001 0010h~~–[a]~~	AES-CCM with a 16 byte MAC	~~NIST SP 800-38C~~ RFC 4309
0001 0014h~~–[a]~~	AES-GCM with a 16 byte MAC	~~NIST SP 800-38D~~ RFC 4106
0000 0400h - 0000 FFFFh	Vendor specific	
PRF and KDF ~~Algorithms~~ algorithms [b]		
0002 0002h~~–[a]~~	IKEv2-use ~~based iterative HMAC KDF~~ based on SHA-1	~~5.13.3.3~~ ~~5.13.3.4~~ ~~5.13.3.3~~ ~~5.13.3.3~~ ~~5.13.3.3~~ table x37 (see 7.7.3.6.3)
0002 0004h~~–[a]~~	IKEv2-use ~~based iterative HMAC KDF~~ based on AES-128 in CBC mode	
0002 0005h~~–[a]~~	IKEv2-use ~~based iterative HMAC KDF~~ based on SHA-256	
~~0002 0006h –[a]~~	~~IKEv2-based iterative HMAC KDF based on SHA-384~~	
0002 0007h~~–[a]~~	IKEv2-use ~~based iterative HMAC KDF~~ based on SHA-512	
0002 0400h - 0002 FFFFh	Vendor specific	

[a] The lower order 16 bits of ~~this~~ these code values are assigned to match an IANA assigned value, if any, for an equivalent IKEv2 encryption algorithm (see 3.1.54) and values less than 100h in the high order 16 bits match the IANA assigned IKEv2 transform type (e.g.~~i.e.~~, 1 – Encryption Algorithms, 2 – PRFs and KDFs ~~Pseudo-random Functions~~).

[b] PRFs are equivalent to the prf() functions defined in RFC 4306. KDFs are equivalent to the prf+() functions defined in RFC 4306.

Table 51 — Security algorithm codes (page 2 of 2)

Code [a]	Description	Reference
Integrity checking (i.e., AUTH) algorithms		
0003 0002h	AUTH_HMAC_SHA1_96	RFC 2404
0003 000Ch	AUTH_HMAC_SHA2_256_128	RFC 4868
0003 000Eh	AUTH_HMAC_SHA2_512_256	RFC 4868
F003 0000h	AUTH_COMBINED	7.7.3.6.4
0003 0400h - 0003 FFFFh	Vendor specific	
Diffie-Hellman algorithms		
0004 000Eh	2 048-bit MODP group (finite field D-H)	RFC 3526
0004 000Fh	3 072-bit MODP group (finite field D-H)	RFC 3526
0004 0010h	4 096-bit MODP group (finite field D-H)	RFC 3526
0004 0013h	256-bit random ECP group	RFC 4753
0004 0015h	521-bit random ECP group	RFC 4753
0004 0400h - 0004 FFFFh	Vendor specific	
SA Authentication payload authentication algorithms		
00F9 0000h	SA_AUTH_NONE	7.7.3.6.6
00F9 0001h	RSA Digital Signature	RFC 4306
00F9 0002h	Shared Key Message Integrity Code	RFC 4306
00F9 0009h	ECDSA with SHA-256 on the P-256 curve	RFC 4754
00F9 000Bh	ECDSA with SHA-512 on the P-521 curve	RFC 4754
00F9 0400h - 00F9 FFFFh	Vendor specific	
Other algorithms		
0000 0000h – 0000 FFFFh	Restricted	IANA
~~0000 0400h – 0000 FFFFh~~	~~Vendor specific~~	
All others ~~values~~	Reserved	

[a] The lower order 16 bits of ~~this~~ these code values are assigned to match an IANA assigned value, if any, for an equivalent IKEv2 encryption algorithm (see 3.1.54) and values less than 100h in the high order 16 bits match the IANA assigned IKEv2 transform type (e.g.~~i.e.~~, 1 – Encryption Algorithms, 2 – PRFs and KDFs ~~Pseudo-random Functions~~).

[b] PRFs are equivalent to the prf() functions defined in RFC 4306. KDFs are equivalent to the prf+() functions defined in RFC 4306.

…

6.29 SECURITY PROTOCOL IN command

~~6.29.1 SECURITY PROTOCOL IN command description~~

The SECURITY PROTOCOL IN command (see table 193) is used to retrieve security protocol information (see 6.29.2) or the results of one or more SECURITY PROTOCOL OUT commands (see 6.30).

Table 193 — SECURITY PROTOCOL IN command
{{no changes in table 193 contents}}

The SECURITY PROTOCOL field (see table 194) specifies which security protocol is being used.

Table 194 — SECURITY PROTOCOL field in SECURITY PROTOCOL IN command

Code	Description	Reference
00h	Security protocol information	~~6.29.2~~ 7.7.1
01h - 06h	Defined by the TCG	3.1.140
07h - 1Fh	Reserved	
20h	Tape Data Encryption	SSC-3
21h	Data Encryption Configuration	TBD
{{insert two new rows (suggest 40h and 41h) and adjust reserved values accordingly}}		
zzh	SA Creation Capabilities	7.7.2
xxh	IKEv2-SCSI	7.7.3
22h - EDh	Reserved	
EEh	Authentication in Host Attachments of Transient Storage Devices	IEEE 1667
EFh	ATA Device Server Password Security	TBD
F0h - FFh	Vendor Specific	

Editors Note 2 - ROW: SECURITY PROTOCOL field code values 21h – 2Fh are tentatively reserved for SSC-x uses.

The contents of the SECURITY PROTOCOL SPECIFIC field depend on the protocol specified by the SECURITY PROTOCOL field (see table 194).

A 512 increment (INC_512) bit set to one specifies that the ALLOCATION LENGTH field (see 4.3.4.6) expresses the maximum number of bytes available to receive data in increments of 512 bytes (e.g., a value of one means 512 bytes, two means 1 024 bytes, etc.). Pad bytes may or may not be appended to meet this length. Pad bytes shall have a value of 00h. An INC_512 bit set to zero specifies that the ALLOCATION LENGTH field expresses the number of bytes to be transferred.

Indications of data overrun or underrun and the mechanism, if any, for processing retries depend on the protocol specified by the SECURITY PROTOCOL field (see table 194).

Any association between a previous SECURITY PROTOCOL OUT command and the data transferred by a SECURITY PROTOCOL IN command depends on the protocol specified by the SECURITY PROTOCOL field (see table 194). If the device server has no data to transfer (e.g., the results for any previous SECURITY PROTOCOL

OUT commands are not yet available), the device server may transfer data indicating it has no other data to transfer.

The format of the data transferred depends on the protocol specified by the SECURITY PROTOCOL field (see table 194).

The device server shall retain data resulting from a SECURITY PROTOCOL OUT command, if any, until one of the following events is processed:

a) Transfer of the data via a SECURITY PROTOCOL IN command from the same I_T_L nexus as defined by the protocol specified by the SECURITY PROTOCOL field (see table 194);
b) Logical unit reset (See SAM-4); or
c) I_T nexus loss (See SAM-4) associated with the I_T nexus that sent the SECURITY PROTOCOL OUT command.

If the data is lost due to one of these events the application client may send a new SECURITY PROTOCOL OUT command to retry the operation.

~~**6.29.2 Security protocol information description**~~

~~**6.29.2.1 Overview**~~

~~...~~ {{Move the entire contents of 6.29.2 to 7.7.1.}} ~~...~~

~~**6.29.2.4.3 Attribute certificate description**~~

~~RFC 3281 defines the certificate syntax for certificates consistent with X.509v2 Attribute Certificate Specification. Any further restrictions beyond the requirements of RFC 3281 are yet to be defined by T10.~~

6.30 SECURITY PROTOCOL OUT command

The SECURITY PROTOCOL OUT command (see table 198) is used to send data to the logical unit. The data sent specifies one or more operations to be performed by the logical unit. The format and function of the operations depends on the contents of the SECURITY PROTOCOL field (see table 199). Depending on the protocol specified by the SECURITY PROTOCOL field, the application client may use the SECURITY PROTOCOL IN command (see 6.29) to retrieve data derived from these operations.

Table 198 — SECURITY PROTOCOL OUT command
{{no changes in table 198 contents}}

The SECURITY PROTOCOL field (see table 199) specifies which security protocol is being used.

Table 199 — SECURITY PROTOCOL field in SECURITY PROTOCOL OUT command

Code	Description	Reference
00h	Reserved	
01h - 06h	Defined by the TCG	3.1.140
07h - 1Fh	Reserved	
20h	Tape Data Encryption	SSC-3
21h	Data Encryption Configuration	TBD
	{{insert one new row (suggest 41h) and adjust reserved values accordingly}}	
xxh	IKEv2-SCSI	7.7.3
22h - EDh	Reserved	
EEh	Authentication in Host Attachments of Transient Storage Devices	IEEE 1667
EFh	ATA Device Server Password Security	TBD
F0h - FFh	Vendor Specific	

Editors Note 3 - ROW: SECURITY PROTOCOL field code values 21h – 2Fh are tentatively reserved for SSC-x uses.

The contents of the SECURITY PROTOCOL SPECIFIC field depend on the protocol specified by the SECURITY PROTOCOL field (see table 199).

A 512 increment (INC_512) bit set to one specifies that the TRANSFER LENGTH field (see 4.3.4.4) expresses the number of bytes to be transferred in increments of 512 bytes (e.g., a value of one means 512 bytes, two means 1 024 bytes, etc.). Pad bytes shall be appended as needed to meet this requirement. Pad bytes shall have a value of 00h. A INC_512 bit set to zero specifies that the TRANSFER LENGTH field indicates the number of bytes to be transferred.

Any association between a SECURITY PROTOCOL OUT command and a subsequent SECURITY PROTOCOL IN command depends on the protocol specified by the SECURITY PROTOCOL field (see table 199). Each protocol shall define whether:

a) The device server shall complete the command with GOOD status as soon as it determines the data has been correctly received. An indication that the data has been processed is obtained by sending a SECURITY PROTOCOL IN command and receiving the results in the associated data transfer; or
b) The device server shall complete the command with GOOD status only after the data has been successfully processed and an associated SECURITY PROTOCOL IN command is not required.

The format of the data transferred depends on the protocol specified by the SECURITY PROTOCOL field (see table 199).

…

7.7 Security protocol parameters

7.7.1 Security protocol information description

{{Move the entire contents of 6.29.2 here}}

7.7.2 SA creation capabilities

{{All of 7.7.2 and 7.7.3 are new. Additions/deletions markups are not applied in these subclauses.}}

7.7.2.1 Overview

If the SECURITY PROTOCOL field in a SECURITY PROTOCOL IN command (see 6.29) is set to zzh, then the command returns information related to the SA creation (see 5.13.2.2) capabilities provided by the device server.

The SA creation capabilities protocol is independent of any other SA creation protocols. The device server shall not refuse to process an SA creation capabilities SECURITY PROTOCOL IN command, and this processing shall not affect the state maintained for any SA creation CCS (e.g., an IKEv2-SCSI CCS) on any I_T_L nexus. Except for those cases where an SA creation capabilities SECURITY PROTOCOL IN command reports changed SA creation capabilities, processing of the command shall not affect the concurrent processing of any commands that are part of an SA creation CCS.

If any SA creation protocols are supported, the SA creation capabilities protocol shall be supported as described in 7.7.2.

The SA creation capabilities SECURITY PROTOCOL IN CDB format is described in 7.7.2.2.

As shown in table x5 (see 7.7.2.2), the format of the parameter data returned by a SA creation capabilities SECURITY PROTOCOL IN command depends on the value in the SECURITY PROTOCOL SPECIFIC field in the CDB.

7.7.2.2 SA creation capabilities CDB description

The SA creation capabilities SECURITY PROTOCOL IN CDB has the format defined in 6.29 with the additional requirements described in this subclause.

When the SECURITY PROTOCOL field is set to SA creation capabilities (i.e., zzh) in a SECURITY PROTOCOL IN command, the SECURITY PROTOCOL SPECIFIC field (see table x5) identifies the SA creation protocol (see 5.13.2.2) for which the device server shall return capability information.

Table x5 — SECURITY PROTOCOL SPECIFIC field for the SA creation capabilities SECURITY PROTOCOL IN command

Code	Description	Parameter data format
0000h	Supported device server capabilities formats	7.7.2.3.1
0001h – 0100h	Reserved	
0101h	IKEv2-SCSI device server capabilities	7.7.2.3.2
0102h – EFFFh	Reserved	
F000h – FFFFh	Vendor Specific	

{{It is intended that the definition of a second SA Creation protocol be accompanied by the addition of a security protocol specific code (probably 0000h or 0102h). However, it is also possible to introduce new SA Creation protocols by modifying the data in the IKEv2-SCSI SA Creation payload (see 7.7.3.5.11).}}

If an SA creation capabilities SECURITY PROTOCOL IN command is received with the INC_512 bit is set to one, then the SECURITY PROTOCOL IN 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.

7.7.2.3 SA creation capabilities parameter data formats

7.7.2.3.1 Supported device server capabilities formats parameter data format

The supported device server capabilities formats parameter data (see table x6) indicates which device server capabilities parameter data formats are supported by the device server.

Table x6 — Supported device server capabilities formats parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PARAMETER DATA LENGTH (n-3)				(LSB)
	Supported capabilities parameter data formats							
4								
5				CAPABILITIES PARAMETER DATA FORMAT (first) (0000h)				
				. . .				
n-1								
n				CAPABILITIES PARAMETER DATA FORMAT (last)				

The PARAMETER DATA LENGTH field indicates the number of bytes that follow in the parameter data.

Each CAPABILITIES PARAMETER DATA FORMAT field in the supported capabilities parameter data formats list shall contain one of the SECURITY PROTOCOL SPECIFIC field values (see table x5) supported by the device server. The values shall be listed in ascending order starting with 0000h.

7.7.2.3.2 IKEv2-SCSI device server capabilities parameter data format

The IKEv2-SCSI device server capabilities parameter data (see table x7) indicates the IKEv2 transforms (i.e., key exchange protocols and authentication protocols) supported by the device server for IKEv2-SCSI.

Table x7 — IKEv2-SCSI device server capabilities parameter data

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PARAMETER DATA LENGTH (n-3)				(LSB)
4								
n				IKEv2-SCSI SA Creation Capabilities payload (see 7.7.3.5.11)				

The PARAMETER DATA LENGTH field indicates the number of bytes that follow in the parameter data.

The IKEv2-SCSI SA Creation Capabilities payload (see 7.7.3.5.11) indicates the algorithms supported by IKEv2-SCSI in the Key Exchange step (see 5.13.4.8) and Authentication step (see 5.13.4.9).

NOTE x8 - The primary content of the IKEv2-SCSI device server capabilities SA creation capabilities parameter data is an IKEv2-SCSI payload (see 7.7.3.5) because the IKEv2-SCSI SA Creation Capabilities payload is used by the device server and application client in the construction of the IKEv2-SCSI Authentication payload (see 7.7.3.5.6).

7.7.3 IKEv2-SCSI

7.7.3.1 Overview

If the SECURITY PROTOCOL field in a SECURITY PROTOCOL OUT command (see 6.30) or a SECURITY PROTOCOL IN command (see 6.29) is set to xxh, then the command is part of an IKEv2-SCSI CCS (see 5.13.4) and is used to transfer IKEv2-SCSI protocol information to or from the device server.

In an IKEv2-SCSI CCS, a defined sequence of SECURITY PROTOCOL OUT and SECURITY PROTOCOL IN commands are sent by the application client and processed by the device server as summarized in 5.13.4.1.

The IKEv2-SCSI SECURITY PROTOCOL OUT CDB format is described in 7.7.3.3.

The IKEv2-SCSI SECURITY PROTOCOL IN CDB format is described in 7.7.3.2.

The IKEv2-SCSI SECURITY PROTOCOL OUT command and the IKEv2-SCSI SECURITY PROTOCOL IN command use the same parameter data format and this format is described in 7.7.3.4. A significant component of the IKEv2-SCSI parameter data format is one or more IKE payloads and the format of IKE payloads is described in 7.7.3.5.

If the IKEv2-SCSI SA creation protocol is supported (see 7.7.1), the SA creation capabilities protocol (see 7.7.2) shall also be supported.

7.7.3.2 IKEv2-SCSI SECURITY PROTOCOL IN CDB description

The IKEv2-SCSI SECURITY PROTOCOL IN CDB has the format defined in 6.29 with the additional requirements described in this subclause.

When the SECURITY PROTOCOL field is set to IKEv2-SCSI (i.e., xxh) in a SECURITY PROTOCOL IN command, the SECURITY PROTOCOL SPECIFIC field (see table x8) identifies the IKEv2-SCSI step (see 5.13.4.1) that the device server is to process. If the IKEv2-SCSI SA creation protocol is supported (see 7.7.1), the SECURITY PROTOCOL IN command support requirements are shown in table x8.

Table x8 — SECURITY PROTOCOL SPECIFIC field for the IKEv2-SCSI SECURITY PROTOCOL IN command

Code	Description	Support	Reference	
			Usage	Data format
0000h – 00FFh	Restricted		RFC 4306	RFC 4306
0100h – 0101h	Reserved			
0102h	Key Exchange step	Mandatory	5.13.4.8.3	7.7.3.4
0103h	Authentication step	Mandatory	5.13.4.9.3	7.7.3.4
0104h – EFFFh	Reserved			
F000h – FFFFh	Vendor Specific			

If an IKEv2-SCSI SECURITY PROTOCOL IN command is received with the INC_512 bit is set to one while the device server is maintaining state for an IKEv2-SCSI CCS on the I_T_L nexus on which the command was received (see 5.13.4.1), then:

a) The SECURITY PROTOCOL IN command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS; and
b) The device server shall continue the IKEv2-SCSI CCS by preparing to receive another SECURITY PROTOCOL IN command or SECURITY PROTOCOL OUT command, as appropriate.

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

If an IKEv2-SCSI SECURITY PROTOCOL IN command is received with the INC_512 bit set to one while the device server is not maintaining state for an IKEv2-SCSI CCS (see 5.13.4.1), then the SECURITY PROTOCOL IN 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.

7.7.3.3 IKEv2-SCSI SECURITY PROTOCOL OUT CDB description

The IKEv2-SCSI SECURITY PROTOCOL OUT CDB has the format defined in 6.30 with the additional requirements described in this subclause.

When the SECURITY PROTOCOL field is set to IKEv2-SCSI (i.e., xxh) in a SECURITY PROTOCOL OUT command, the SECURITY PROTOCOL SPECIFIC field (see table x9) identifies the IKEv2-SCSI step (see 5.13.4.1) that the device server is to process. If the IKEv2-SCSI SA creation protocol is supported (see 7.7.1), the SECURITY PROTOCOL IN command support requirements are shown in table x9.

Table x9 — SECURITY PROTOCOL SPECIFIC field for the IKEv2-SCSI SECURITY PROTOCOL OUT command

Code	Description	Support	Reference	
			Usage	Data format
0000h – 00FFh	Restricted		RFC 4306	RFC 4306
0100h – 0101h	Reserved			
0102h	Key Exchange step	Mandatory	5.13.4.8.3	7.7.3.4
0103h	Authentication step	Mandatory	5.13.4.9.3	7.7.3.4
0104h	Delete operation	Mandatory	5.13.4.13	7.7.3.4
0105h – EFFFh	Reserved			
F000h – FFFFh	Vendor Specific			

If an IKEv2-SCSI SECURITY PROTOCOL OUT command is received with the INC_512 bit is set to one while the device server is maintaining state for an IKEv2-SCSI CCS on the I_T_L nexus on which the command was received (see 5.13.4.1), then:

a) The SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS; and
b) The device server shall continue the IKEv2-SCSI CCS by preparing to receive another SECURITY PROTOCOL OUT command or SECURITY PROTOCOL IN command, as appropriate.

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

If an IKEv2-SCSI SECURITY PROTOCOL OUT command is received with the INC_512 bit is set to one while the device server is not maintaining state for an IKEv2-SCSI CCS (see 5.13.4.1), then the SECURITY PROTOCOL OUT 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.

Any IKEv2-SCSI SECURITY PROTOCOL OUT command with an transfer length of up to 16 384 bytes shall not be terminated with an error due to the number of bytes to be transferred and processed.

7.7.3.4 IKEv2-SCSI parameter data format

Table x10 shows the parameter list format used by a SECURITY PROTOCOL OUT command and the parameter data format used by a SECURITY PROTOCOL IN command when the SECURITY PROTOCOL field is set to IKEv2-SCSI (i.e., xxh).

Table x10 — IKEv2-SCSI SECURITY PROTOCOL OUT and SECURITY PROTOCOL IN parameter data

Bit / Byte	7	6	5	4	3	2	1	0
IKEv2-SCSI header								
0	(MSB)			IKE_SA APPLICATION CLIENT SAI				
7								(LSB)
8	(MSB)			IKE_SA DEVICE SERVER SAI				
15								(LSB)
16	NEXT PAYLOAD							
17	MAJOR VERSION (2h)				MINOR VERSION			
18	EXCHANGE TYPE							
19	Reserved			INTTR	VERSION	RSPNS	Reserved	
20	(MSB)			MESSAGE ID				
23								(LSB)
24	(MSB)			IKE LENGTH (n+1)				
27								(LSB)
IKEv2-SCSI payloads								
28				First IKEv2-SCSI payload (see 7.7.3.5)				
				. . .				
n				Last IKEv2-SCSI payload (see 7.7.3.5)				

The IKE_SA APPLICATION CLIENT SAI field contains the value that will become the AC_SAI SA parameter (see 5.13.2.2) when the SA is generated (see 5.13.4.11). The AC_SAI is chosen by the application client to uniquely identify its representation of the SA that is being negotiated.

If the device server receives an IKEv2-SCSI header with the IKE_SA APPLICATION CLIENT SAI field set to zero, then the error shall be processed as described in 7.7.3.8.

To increase procedural integrity checking, the application client should compare the IKE_SA APPLICATION CLIENT SAI field contents in any SECURITY PROTOCOL IN parameter data it receives to the value that the application client is maintaining for the IKEv2-SCSI CCS. If the two values are not identical, the application client should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

Except in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2), the IKE_SA DEVICE SERVER SAI field contains the value that is destined to become the DS_SAI SA parameter when the SA is

generated. The DS_SAI is chosen by the device server in accordance with the requirements in 5.13.2.1 to uniquely identify its representation of the SA that is being negotiated. In the Key Exchange step SECURITY PROTOCOL OUT command the IKE_SA DEVICE SERVER SAI field is reserved.

To increase procedural integrity checking, the application client should compare the IKE_SA DEVICE SERVER SAI field contents in the Authentication step SECURITY PROTOCOL IN parameter data it receives to the value that the application client is maintaining for the IKEv2-SCSI CCS. If the two values are not identical, the application client should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

The device server shall compare the contents of the IKE_SA APPLICATION CLIENT SAI field and the IKE_SA DEVICE SERVER SAI field in the Authentication step SECURITY PROTOCOL OUT parameter list to the SAI values the device server is maintaining for the IKEv2-SCSI CCS on the I_T_L nexus on which the command was received. If the values do not match, then:

a) The SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to SA CREATION PARAMETER VALUE REJECTED; and
b) The device server shall continue the IKEv2-SCSI CCS by preparing to receive another Authentication step SECURITY PROTOCOL OUT command.

{{SA CREATION PARAMETER VALUE REJECTED is a new additional sense code.}}

The NEXT PAYLOAD field (see table x11) identifies the first IKEv2-SCSI payload that follows the IKEv2-SCSI header.

Table x11 — NEXT PAYLOAD field

Code	IKE Payload Name	Support requirements in SECURITY PROTOCOL …		Reference
		IN	OUT	
00h	No Next Payload	Mandatory		7.7.3.5.2
01h - 20h		Reserved		
21h	Security Association	Prohibited [a]		RFC 4306
22h	Key Exchange	Mandatory		7.7.3.5.3
23h	Identification – Application Client	Prohibited	Mandatory	7.7.3.5.4
24h	Identification – Device Server	Mandatory	Prohibited	7.7.3.5.4
25h	Certificate	Optional		7.7.3.5.5
26h	Certificate Request	Optional		7.7.3.5.5
27h	Authentication	Mandatory		7.7.3.5.6
28h	Nonce	Mandatory		7.7.3.5.7
29h	Notify	Prohibited	Mandatory	7.7.3.5.8
2Ah	Delete	Prohibited	Mandatory	7.7.3.5.9
2Bh	Vendor ID	Prohibited		RFC 4306

[a] The Security Association payload type value is not used in IKEv2-SCSI. The IKEv2-SCSI SA Cryptographic Algorithms payload (i.e., 81h) and IKEv2-SCSI UT Cryptographic Algorithms payload (i.e., 82h) are used instead.

Table x11 — NEXT PAYLOAD field

Code	IKE Payload Name	Support requirements in SECURITY PROTOCOL …		Reference
		IN	OUT	
2Ch	Traffic Selector – Application Client	Prohibited		RFC 4306
2Dh	Traffic Selector – Device Server	Prohibited		RFC 4306
2Eh	Encrypted	Mandatory		7.7.3.5.10
2Fh	Configuration	Prohibited		RFC 4306
30h	Extensible Authentication	Prohibited		RFC 4306
31h - 7Fh		Restricted		RFC 4306
80h	IKEv2-SCSI SA Creation Capabilities	Mandatory		7.7.3.5.11
81h	IKEv2-SCSI SA Cryptographic Algorithms	Mandatory		7.7.3.5.12
82h	IKEv2-SCSI UT Cryptographic Algorithms	Mandatory		7.7.3.5.13
83h	IKEv2-SCSI Timeout Values	Mandatory		7.7.3.5.14
84h - BFh		Reserved		
C0h - FFh	Vendor Specific			

[a] The Security Association payload type value is not used in IKEv2-SCSI. The IKEv2-SCSI SA Cryptographic Algorithms payload (i.e., 81h) and IKEv2-SCSI UT Cryptographic Algorithms payload (i.e., 82h) are used instead.

The MAJOR VERSION field shall contain the value 2h. If a device server receives an IKE header with a MAJOR VERSION field containing a value other than 2h, then the error shall be processed as described in 7.7.3.8.

The MINOR VERSION field is reserved.

The EXCHANGE TYPE field is reserved.

The initiator (INTTR) bit shall be set to:

a) One for SECURITY PROTOCOL OUT commands; and
b) Zero for SECURITY PROTOCOL IN commands.

If a device server receives an IKEv2-SCSI header with the INTTR bit set to zero, then the error shall be processed as described in 7.7.3.8.

If an application client receives an IKEv2-SCSI header with the INTTR bit set to one, it should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

The VERSION bit is reserved.

The response (RSPNS) bit shall be set to:

a) Zero for SECURITY PROTOCOL OUT commands; and
b) One for SECURITY PROTOCOL IN commands.

If a device server receives an IKEv2-SCSI header with the RSPNS bit set to one, then the error shall be processed as described in 7.7.3.8.

If an application client receives an IKEv2-SCSI header with the RSPNS bit set to zero, it should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

The MESSAGE ID field (see table x12) identifies the function of the parameter data.

Table x12 — MESSAGE ID field

Code	Description
0000h	Key Exchange step a SECURITY PROTOCOL OUT command or SECURITY PROTOCOL IN command
0001h	Authentication step a SECURITY PROTOCOL OUT command or SECURITY PROTOCOL IN command
0002h	Delete operation in a SECURITY PROTOCOL OUT command
0003h - FFFFh	Reserved

If the device server receives a SECURITY PROTOCOL OUT command with an invalid MESSAGE ID field in its IKEv2-SCSI header, then the error shall be processed as described in 7.7.3.8.

If the application client receives an invalid message id field in the parameter data for a SECURITY PROTOCOL IN command, the application client should abandon the IKEv2-SCSI CCS and notify the device server that the IKEv2-SCSI CCS is being abandoned as described in 5.13.4.12.

The IKE LENGTH field contains the total number of bytes in the parameter data, including the IKEv2-SCSI header and all the IKEv2-SCSI payloads.

NOTE x9 - The contents of the IKE LENGTH field differ from those found in most SCSI length fields, however, they are consistent with the IKEv2 usage (see RFC 4306).

Each IKEv2-SCSI payload (see 7.7.3.5) contains specific data related to the operation being performed. A specific combination of IKEv2-SCSI payloads is specified for each operation (e.g., Key Exchange) as summarized in 5.13.4.2. The Encryption payload (see 7.7.3.6.2) nests one set of IKEv2-SCSI payloads inside another.

Based on the contents of the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL OUT CDB, the device server shall:

a) Validate the contents of the NEXT PAYLOAD field in the IKEv2-SCSI header as shown in table x13, before performing any decryption or integrity checking; and
b) Validate that the number of instances of each NEXT PAYLOAD value shown in table x13 occur in the parameter data, after the encrypted data, in any, is decrypted and integrity checked (i.e., if the NEXT PAYLOAD field in the IKEv2-SCSI contains 2Eh, after the Encrypted payload is decrypted and integrity checked).

If the NEXT PAYLOAD fields in the unencrypted SECURITY PROTOCOL OUT parameter data do not meet the minimum requirements shown in table x13 for the listed SECURITY PROTOCOL SPECIFIC field contents, then the error shall be processed as described in 7.7.3.8.

Based on the contents of the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL IN CDB, the device server shall:

a) Place one of the NEXT PAYLOAD values allowed by table x13 in the NEXT PAYLOAD field in the IKEv2-SCSI header; and
b) Include that the number of instances of each NEXT PAYLOAD value shown in table x13 occur in the parameter data either before or after encryption, if applicable (i.e., if the NEXT PAYLOAD field in the IKEv2-SCSI contains 2Eh, before the contents of the Encrypted payload are encrypted and integrity check value is computed).

Table x13 — NEXT PAYLOAD field values in SECURITY PROTOCOL OUT/IN parameter data (page 1 of 3)

NEXT PAYLOAD field value	NEXT PAYLOAD value allowed in IKEv2-SCSI header	Number of NEXT PAYLOAD fields allowed to contain value
SECURITY PROTOCOL OUT command with SECURITY PROTOCOL SPECIFIC field set to 0102h (i.e., Key Exchange step)		**Authentication step not skipped (see 5.13.4.1)**
84h (i.e., IKEv2-SCSI Timeout Values)	Yes	1
81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms)	Yes	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	No	0
22h (i.e., Key Exchange)	Yes	1
28h (i.e., Nonce)	Yes	1
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0
SECURITY PROTOCOL OUT command with SECURITY PROTOCOL SPECIFIC field set to 0102h (i.e., Key Exchange step)		**Authentication step skipped (see 5.13.4.1)**
84h (i.e., IKEv2-SCSI Timeout Values)	Yes	1
81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms)	Yes	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	Yes	1
22h (i.e., Key Exchange)	Yes	1
28h (i.e., Nonce)	Yes	1
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0

Table x13 — NEXT PAYLOAD field values in SECURITY PROTOCOL OUT/IN parameter data (page 2 of 3)

NEXT PAYLOAD field value	NEXT PAYLOAD value allowed in IKEv2-SCSI header	Number of NEXT PAYLOAD fields allowed to contain value
SECURITY PROTOCOL IN command with SECURITY PROTOCOL SPECIFIC field set to 0102h (i.e., Key Exchange step)		**Authentication step not skipped (see 5.13.4.1)**
84h (i.e., IKEv2-SCSI Timeout Values)	Yes	1
81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms)	Yes	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	No	0
22h (i.e., Key Exchange)	Yes	1
28h (i.e., Nonce)	Yes	1
26h (i.e., Certificate Request)	Yes	0 or more
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0
SECURITY PROTOCOL IN command with SECURITY PROTOCOL SPECIFIC field set to 0102h (i.e., Key Exchange step)		**Authentication step skipped (see 5.13.4.1)**
84h (i.e., IKEv2-SCSI Timeout Values)	Yes	1
81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms)	Yes	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	Yes	1
22h (i.e., Key Exchange)	Yes	1
28h (i.e., Nonce)	Yes	1
26h (i.e., Certificate Request)	Yes	0 or more
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0
SECURITY PROTOCOL OUT command with SECURITY PROTOCOL SPECIFIC field set to 0103h (i.e., Authentication step)		
2Eh (i.e., Encrypted)	Yes	1
23h (i.e., Identification – Application Client)	No	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	No	1
25h (i.e., Certificate)	No	0 or more
26h (i.e., Certificate Request)	No	0 or more
29h (i.e., Notify)	No	0 or 1
27h (i.e., Authentication)	No	1
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0

Table x13 — NEXT PAYLOAD field values in SECURITY PROTOCOL OUT/IN parameter data (page 3 of 3)

NEXT PAYLOAD field value	NEXT PAYLOAD value allowed in IKEv2-SCSI header	Number of NEXT PAYLOAD fields allowed to contain value
SECURITY PROTOCOL IN command with SECURITY PROTOCOL SPECIFIC field set to 0103h (i.e., Authentication step)		
2Eh (i.e., Encrypted)	Yes	1
23h (i.e., Identification – Device Server)	No	1
82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms)	No	1
25h (i.e., Certificate)	No	0 or more
27h (i.e., Authentication)	No	1
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0
SECURITY PROTOCOL OUT command with SECURITY PROTOCOL SPECIFIC field set to 0104h (i.e., Delete operation)		
2Eh (i.e., Encrypted)	Yes	1
2Ah (i.e., Delete)	No	1
00h (i.e., No Next Payload)	No	1
All other next payload codes	No	0

7.7.3.5 IKEv2-SCSI payloads

7.7.3.5.1 IKEv2-SCSI payload format

Each IKEv2-SCSI payload (see table x14) is composed of a header and data that is specific to the payload type.

Table x14 — IKEv2-SCSI payload format

Bit / Byte	7	6	5	4	3	2	1	0
IKEv2-SCSI payload header								
0	NEXT PAYLOAD							
1	CRIT	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
IKEv2-SCSI payload-specific data								
4								
n								

The NEXT PAYLOAD field identifies the IKEv2-SCSI payload that follows this IKEv2-SCSI payload using one of the code values shown in table x11 (see 7.7.3.4).

~~Based on the contents of the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL OUT CDB, the device server shall validate the contents of the NEXT PAYLOAD field in each decrypted or unencrypted IKEv2-SCSI payload header as shown in table x15.~~

Table x15 — NEXT PAYLOAD field contents in decrypted SECURITY PROTOCOL OUT payload headers

SECURITY PROTOCOL SPECIFIC field contents	Required NEXT PAYLOAD field contents in a decrypted or unencrypted IKEv2-SCSI payload header
0102h (i.e., Key Exchange step)	One of the following: a) 84h (i.e., IKEv2-SCSI Timeout Values), b) 81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms), c) 82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms), [a] d) 22h (i.e., Key Exchange), e) 28h (i.e., Nonce), or f) 00h (i.e., No Next Payload)
0103h (i.e., Authentication step)	One of the following: a) 82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms), b) 23h (i.e., Identification – Application Client), c) 25h (i.e., Certificate), d) 26h (i.e., Certificate Request), e) 29h (i.e., Notify), or f) 27h (i.e., Authentication)
0104h (i.e., Delete operation)	2Ah (i.e., Delete)

[a] Allowed only if the Authentication step is skipped (see 5.13.4.1). If the Authentication step is not skipped, the presence of 82h in the NEXT PAYLOAD field in an IKEv2-SCSI payload header shall be processed as an error.

~~If the NEXT PAYLOAD field in a decrypted or unencrypted IKEv2-SCSI payload header does not contain one the values shown in table x15 for the listed SECURITY PROTOCOL SPECIFIC field contents, then the error shall be processed as described in 7.7.3.8.~~

~~Based on the contents of the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL OUT CDB and whether the Authentication step is skipped (see 5.13.4.6), the device server shall verify that all of the next payload values shown in table x16 appear somewhere in the parameter data.~~

Table x16 — Mandatory next payload values

SECURITY PROTOCOL SPECIFIC field contents	Authentication step skipped	Mandatory next payload values
0102h (i.e., Key Exchange step)	No	84h (i.e., IKEv2-SCSI Timeout Values) 81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms) 22h (i.e., Key Exchange) 28h (i.e., Nonce) 00h (i.e., No Next Payload)
	Yes	84h (i.e., IKEv2-SCSI Timeout Values) 81h (i.e., IKEv2-SCSI SA Cryptographic Algorithms) 82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms) 22h (i.e., Key Exchange) 28h (i.e., Nonce) 00h (i.e., No Next Payload)
0103h (i.e., Authentication step)	n/a	82h (i.e., IKEv2-SCSI UT Cryptographic Algorithms) 23h (i.e., Identification – Application Client) 27h (i.e., Authentication) 00h (i.e., No Next Payload)
0104h (i.e., Delete operation)	n/a	00h (i.e., No Next Payload)

~~If the unencrypted and encrypted SECURITY PROTOCOL OUT command parameter data does not contain all the next payload values shown in table x16 for the listed SECURITY PROTOCOL SPECIFIC field contents, then the error shall be processed as described in 7.7.3.8.~~

~~If the unencrypted and encrypted SECURITY PROTOCOL OUT command parameter data contains the same next payload value more than once, then the error shall be processed as described in 7.7.3.8.~~

If a device server receives an IKEv2-SCSI payload that it does not recognize (e.g., an IKEv2-SCSI payload identified by a next payload value of 01h) with the critical (CRIT) bit set to one, then the error shall be processed as described in 7.7.3.8.

If an application client receives an IKEv2-SCSI payload that it does not recognize with the CRIT bit set to one, then the application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

If an application client or device server receives an IKEv2-SCSI payload that it does not recognize with the CRIT bit set to zero, then it should use the NEXT PAYLOAD field and the IKE PAYLOAD LENGTH field to skip processing of the unrecognized IKEv2-SCSI payload and continue processing at the next IKEv2-SCSI payload.

The IKE PAYLOAD LENGTH field contains the total number of bytes in the payload, including the IKEv2-SCSI payload header. The value in the IKE PAYLOAD LENGTH field need not be a multiple of two or four (i.e., no byte alignment is maintained among IKEv2-SCSI payloads).

NOTE x10 - The contents of the IKE PAYLOAD LENGTH field differ from those found in most SCSI length fields, however, they are consistent with the IKEv2 usage (see RFC 4306).

The format and contents of the IKEv2-SCSI payload-specific data depends on the value in the NEXT PAYLOAD field of:

a) The IKEv2-SCSI header (see 7.7.3.4), if this is the first IKEv2-SCSI payload in the parameter data; or
b) The previous IKEv2-SCSI payload, in all other cases.

7.7.3.5.2 No Next payload

A NEXT PAYLOAD field that is set to 00h (i.e., No Next payload) specifies that no more IKEv2-SCSI payloads follow the current payload. The IKEv2-SCSI No Next payload contains no bytes and has no format.

7.7.3.5.3 Key Exchange payload

The Key Exchange payload (see table x17) transfers Diffie-Hellman shared key exchange data between an application client and a device server or vice versa.

Table x17 — Key Exchange payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	(MSB)	DIFFIE-HELLMAN GROUP NUMBER						
5								(LSB)
6		Reserved						
7								
8	KEY EXCHANGE DATA							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Key Exchange payload.

The DIFFIE-HELLMAN GROUP NUMBER field contains the least significant 16 bits from ALGORITHM IDENTIFIER field in the D-H IKEv2-SCSI algorithm descriptor (see 7.7.3.6.5) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12).

The KEY EXCHANGE DATA field contains the sender's Diffie-Hellman public value for this key exchange. The format of key exchange data is as specified in the reference cited in that registry for the value used.

When a prime modulus (i.e., mod p) Diffie-Hellman group is used, the length of the Diffie-Hellman public value shall be equal to the length of the prime modulus over which the exponentiation was performed as shown in table x41 (see 7.7.3.6.5). Zero bits shall be prepended to the KEY EXCHANGE DATA field if necessary.

Diffie-Hellman exponential reuse and reuse of analogous Diffie-Hellman public values for Diffie-Hellman mechanisms not based on exponentiation should not be performed, but if performed it shall be constrained (e.g., requirements regarding when Diffie-Hellman information is discarded) as specified in RFC 4306. The freshness and

randomness of the random nonces are critical to the security of IKEv2-SCSI when Diffie-Hellman exponentials and public values are reused (see RFC 4306).

7.7.3.5.4 Identification – Application Client payload and Identification – Device Server payload

The Identification – Application Client payload (see table x18) transfers identification information from the application client to the device server. The Identification – Device Server payload (see table x18) transfers identification information from the device server to the application client.

Table x18 — Identification payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	ID TYPE							
5	Reserved							
7								
8	IDENTIFICATION DATA							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Identification – Application Client payload and the Identification – Device Server payload.

The ID TYPE field describes the contents of the IDENTIFICATION DATA field and shall contain one of the named values shown in table x19.

Table x19 — ID TYPE field

Code	Name [a]	Contents of the IDENTIFICATION DATA field
09h	ID_DER_ASN1_DN	The value of a certificate subject field (see RFC 3280)
0Ah	ID_DER_ASN1_GN	The value of a name contained in a Subject Alternative Name (i.e., SubjectAltName) certificate extension (see RFC 3280)
0Bh	ID_FC_NAME	FC-SP certificates that certify a Fibre Channel name as an identity to be used (see RFC 4595 and FC-SP)
0Ch	ID_KEY_ID	Arbitrary identity data (e.g., SCSI port names, SCSI device names)
all other values	Prohibited	

[a] See RFC 4306 and RFC 4595 for the code value assignments for these names and for additional information about the associated identification data.

The contents of the IDENTIFICATION DATA field depend on the value in the ID TYPE field.

When the Certificate payload is included in the parameter data, the identity in the Identification – Application Client payload or Identification – Device Server payload is not required to match anything in the Certificate payload (see RFC 4306). A mechanism out side the scope of this standard shall be provided to configure any application client or device server to require a match between the identity in an Identification payload and the subject name or subject alternative name in a Certificate payload.

If a device server receives an Identification – Application Client payload that does not conform to the requirements in RFC 4306 or the requirements in this subclause, then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

If an application client receives an Identification – Device Server payload that does not conform to the requirements in RFC 4306 or the requirements in this subclause, then the application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

7.7.3.5.5 Certificate payload and Certificate Request payload

The Certificate Request payload (see table x20) allows an application client or device server to request the use of certificates as part of identity authentication and to name one or more trust anchors (see RFC 4306) for the certificate verification process. The Certificate payload (see table x20) delivers a requested identity authentication certificate. The protocol for using Certificate Request payloads and Certificate payloads is described in 5.13.4.3.

Table x20 — Certificate Request payload and Certificate payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	CERTIFICATE ENCODING							
5	CERTIFICATE DATA							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Certificate Request payload and the Certificate payload.

The CERTIFICATE ENCODING field describes the contents of the CERTIFICATE DATA field and shall contain one of the named values shown in table x21.

Table x21 — CERTIFICATE ENCODING field

Code	Description	Reference
00h	Reserved	
01h-03h	Prohibited	Annex C
04h	X.509 Certificate - Signature	RFC 4306
05h-0Ah	Prohibited	Annex C
0Bh	Raw RSA Key	RFC 4306 and RFC 4718
0Ch-0Dh	Prohibited	Annex C
0Eh-C8h	Restricted	IANA
C9h-FFh	Reserved	

The contents of the CERTIFICATE DATA field depend on the value in the CERTIFICATE ENCODING field.

The relationship between the Certificate payload and the Identification payload is described in 7.7.3.5.4.

7.7.3.5.6 Authentication payload

The Authentication payload (see table x22) allows the application client and a device server to verify that the data transfers in their IKEv2-SCSI CCS have not be compromised by a man-in-the-middle attack (see 5.13.1.4).

Table x22 — Authentication payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	AUTH METHOD							
5	Reserved							
7								
8	AUTHENTICATION DATA							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Authentication payload.

The AUTH METHOD field indicates the authentication algorithm to be applied to this Authentication payload. The AUTH METHOD field contains the least significant eight bits of the ALGORITHM IDENTIFIER field in an SA_AUTH_OUT or an SA_AUTH_IN algorithm descriptor (see 7.7.3.6.6) from the Key Exchange step (see 5.13.4.8).

If the contents of the AUTH METHOD field in the parameter data for the Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2) do not match the least significant eight bits in the ALGORITHM IDENTIFIER field in the

SA_AUTH_OUT algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step, then:

a) The SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to ABORTED COMMAND and the additional sense code set to AUTHENTICATION FAILED; and
b) The device server shall abandon the IKEv2-SCSI CCS (see 5.13.4.12).

If the contents of the AUTH METHOD field in the parameter data for the Authentication step SECURITY PROTOCOL IN command (see 5.13.4.9.3) do not match the least significant eight bits in the ALGORITHM IDENTIFIER field in the SA_AUTH_IN algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload in the Key Exchange step, then application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

In the Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2) parameter list, the AUTHENTICATION DATA field contains the result of applying the algorithm specified by the ALGORITHM IDENTIFIER field in the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) as described in table x43 (see 7.7.3.6.6) and this subclause to the following concatenation of bytes:

1) All the bytes in the Data-In Buffer returned by the Device Server Capabilities step (see 5.13.4.7) SECURITY PROTOCOL IN command;
2) All the bytes in the Data-Out Buffer sent by the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) in the same IKEv2-SCSI CCS for which GOOD status was returned;
3) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Nonce payload (see 7.7.3.5.7) that was received in the Key Exchange step SECURITY PROTOCOL IN command in the same IKEv2-SCSI CCS; and
4) All the bytes produced by applying the PRF selected by the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) to the following inputs:
    1) The SK_pi shared key (see 5.13.4.4); and
    2) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Identification – Application Client payload (see 7.7.3.5.4).

When processing the Authentication step SECURITY PROTOCOL OUT command, the device server shall compute the expected contents of the AUTHENTICATION DATA field by applying the algorithm specified by the ALGORITHM IDENTIFIER field in the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) as described in table x43 (see 7.7.3.6.6) and this subclause to the following concatenation of bytes:

1) All the bytes in the Data-In Buffer that the device server returned to any application client in response the last received the Device Server Capabilities step SECURITY PROTOCOL IN command;
2) All the bytes in the Data-Out Buffer received in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) in the same IKEv2-SCSI CCS for which GOOD status was returned;
3) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Nonce payload (see 7.7.3.5.7) sent in the Key Exchange step SECURITY PROTOCOL IN command in the same IKEv2-SCSI CCS; and
4) All the bytes produced by applying the PRF selected by the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) to the following inputs:
    1) The SK_pi shared key (see 5.13.4.4); and
    2) All the bytes received in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Identification – Application Client payload (see 7.7.3.5.4) of the parameter list being processed.

If the expected contents of the AUTHENTICATION DATA field do not match actual contents of the AUTHENTICATION DATA field, then:

a) The SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to ABORTED COMMAND and the additional sense code set to AUTHENTICATION FAILED; and
b) The device server shall abandon the IKEv2-SCSI CCS (see 5.13.4.12).

{{AUTHENTICATION FAILED is a new additional sense code.}}

For the Authentication step SECURITY PROTOCOL IN command (see 5.13.4.9.3) parameter list, the device server shall compute the AUTHENTICATION DATA field contains by applying the algorithm specified by the ALGORITHM IDENTIFIER field in the SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) as described in table x43 (see 7.7.3.6.6) and this subclause to the following concatenation of bytes:

1) All the bytes in the Data-In Buffer that the device server returned to any application client in response the last received the Device Server Capabilities step SECURITY PROTOCOL IN command;
2) All the bytes in the Data-In Buffer sent by the most recent Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the same IKEv2-SCSI CCS for which GOOD status was returned;
3) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Nonce payload (see 7.7.3.5.7) received in the Key Exchange step SECURITY PROTOCOL OUT command in the same IKEv2-SCSI CCS; and
4) All the bytes produced by applying the PRF selected by the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) to the following inputs:
    1) The SK_pr shared key (see 5.13.4.4); and
    2) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Identification – Device Server payload (see 7.7.3.5.4).

After GOOD status is received for the Authentication step SECURITY PROTOCOL IN command, the application client should compute the expected contents of the AUTHENTICATION DATA field by applying the algorithm specified by the ALGORITHM IDENTIFIER field in the SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) as described in table x43 (see 7.7.3.6.6) and this subclause to the following concatenation of bytes:

1) All the bytes in the Data-In Buffer returned by the Device Server Capabilities step (see 5.13.4.7) SECURITY PROTOCOL IN command;
2) All the bytes in the Data-In Buffer returned by the most recent Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) in the same IKEv2-SCSI CCS for which GOOD status was returned;
3) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Nonce payload (see 7.7.3.5.7) sent in the Key Exchange step SECURITY PROTOCOL OUT command in the same IKEv2-SCSI CCS; and
4) All the bytes produced by applying the PRF selected by the PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) to the following inputs:
    1) The SK_pr shared key (see 5.13.4.4); and
    2) All the bytes in the IKEv2-SCSI payload-specific data part (see 7.7.3.5.1) of the Identification – Device Server payload (see 7.7.3.5.4) received in the SECURITY PROTOCOL IN parameter data.

If the expected contents of the AUTHENTICATION DATA field do not match actual contents of the AUTHENTICATION DATA field, then application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

If the AUTH METHOD field contains 01h (i.e., RSA Digital Signature) the RSA digital signature shall be encoded with the EMSA-PKCS1-v1_5 signature encoding method as specified in RFC 2437 (see RFC 4718).

7.7.3.5.7 Nonce payload

The Nonce payload (see table x23) transfers one random nonce (see 3.1.95) from the application client to the device server or from the device server to the application client.

Table x23 — Nonce payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	NONCE DATA							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Nonce payload.

In the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) the NONCE DATA field contains the application client's random nonce.

In the Key Exchange step SECURITY PROTOCOL IN command (see 5.13.4.8.3) the NONCE DATA field contains the device server's random nonce.

The requirements that RFC 4306 places on the nonce data shall apply to this standard.

7.7.3.5.8 Notify payload

This standard uses the Notify payload (see table x24) only to provide initial contact notification from the application client to the device server.

Table x24 — Notify payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (10h)						
3								(LSB)
4	PROTOCOL ID (01h)							
5	SAI SIZE (08h)							
6	(MSB)	NOTIFY MESSAGE TYPE (4000h)						
7								(LSB)
8	SAI							
15								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Notify payload.

The PROTOCOL ID field contains one. If the device server receives a PROTOCOL ID field set to a value other than one, then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

The SAI SIZE field contains eight. If the device server receives a value other than eight in the SAI SIZE field, then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

The NOTIFY MESSAGE TYPE field contains 16 384 (i.e., INITIAL_CONTACT). If the device server receives a value other than 16 384 in the NOTIFY MESSAGE TYPE field, then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

The SAI field contains the device server's SAI. If the contents of the SAI field are not identical to the contents of the IKE_SA DEVICE SERVER SAI field in the IKEv2-SCSI header (see 7.7.3.4), then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

Unless an error is detected, the device server shall process the Notify payload as described in 5.13.4.9.2.

7.7.3.5.9 Delete payload

The Delete payload (see table x25) requests the deletion of an existing SA or the abandonment of an IKEv2-SCSI CCS that is in progress.

Table x25 — Delete payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (18h)						
3								(LSB)
4	PROTOCOL ID (01h)							
5	SAI SIZE (08h)							
6	(MSB)	NUMBER OF SAIS (0002h)						
7								(LSB)
8	AC_SAI							
15								
16	DS_SAI							
23								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Delete payload.

The PROTOCOL ID field contains one. If the device server receives a PROTOCOL ID field set to a value other than one, then the error shall be processed as described in 7.7.3.8.3.

The SAI SIZE field contains eight. If the device server receives a value other than eight in the SAI SIZE field, then the error shall be processed as described in 7.7.3.8.3.

The AC_SAI field contains the AC_SAI SA parameter value (see 3.1.103) for the SA to be deleted. If the contents of the AC_SAI field do not match the contents of the IKE_SA APPLICATION CLIENT SAI field in the IKEv2-SCSI header (see 7.7.3.4), then the error shall be processed as described in 7.7.3.8.3.

The DS_SAI field contains the DS_SAI SA parameter value (see 3.1.103) for the SA to be deleted. If the contents of the DS_SAI field do not match the contents of the IKE_SA DEVICE SERVER SAI field in the IKEv2-SCSI header (see 7.7.3.4), then the error shall be processed as described in 7.7.3.8.3.

If the device server is maintaining SA parameters for which the AC_SAI matches the contents of the AC_SAI field and the DS_SAI matches the contents of the DS_SAI field, that set of SA parameters shall be deleted.

If the device server is maintaining state for an IKEv2-SCSI CCS on the I_T_L nexus on which the command was received, the IKEv2-SCSI CCS shall be abandoned (see 5.13.4.12) if:

a) The contents of the AC_SAI field match the application client's SAI in the maintained state; and
b) The contents of the DS_SAI field match the device server's SAI in the maintained state.

7.7.3.5.10 Encrypted payload

7.7.3.5.10.1 Introduction

The Encrypted payload transfers (see table x26) one or more other IKEv2-SCSI payloads that are encrypted and integrity checked from the application client to the device server and vice versa.

If IKEv2-SCSI parameter data contains the Encrypted payload, then the Encrypted payload is the first payload in the parameter (i.e., the NEXT PAYLOAD field in the IKEv2-SCSI header (see 7.7.3.5.1) contains 2Eh). Since the NEXT PAYLOAD field in an Encrypted payload identifies the first payload in the CIPHERTEXT field, there is no way to identify a payload following the Encrypted payload and none are allowed.

Table x26 — Encrypted payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	INITIALIZATION VECTOR							
i-1								
i	CIPHERTEXT							
k-1								
k	INTEGRITY CHECK VALUE							
n								

The NEXT PAYLOAD field identifies the first IKEv2-SCSI payload in the CIPHERTEXT field using the coded values shown in table x11 (see 7.7.3.5.1).

The CRIT bit and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the Encrypted payload.

The IKE PAYLOAD LENGTH field contains the number of bytes that follow in the Encrypted payload. The number of bytes in the CIPHERTEXT field is equal to the number of bytes in the plaintext (see table x27).

The INITIALIZATION VECTOR field contains the initialization vector encryption algorithm input value. The size of the initialization vector is defined by the encryption algorithm.

The CIPHERTEXT field contains the result of processing the encryption algorithm specified by the ALGORITHM IDENTIFIER field in the ENCR IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.2) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) using the inputs:

a) A byte string composed of:
   1) The AAD, if any, described in 7.7.3.5.10.2;
   2) The contents of the INITIALIZATION VECTOR field (see table x26); and
   3) The plaintext data shown in table x27;
   and

b) The shared key value, and salt value (see table x35 in 7.7.3.6.2) if any, from one of the following shared keys:
   A) If the Encrypted payload appears in SECURITY PROTOCOL OUT command parameter data, the SK_ei shared key (see 5.13.4.4); or
   B) If the Encrypted payload appears in SECURITY PROTOCOL IN command parameter data, the SK_er shared key (see 5.13.4.4).

The INTEGRITY CHECK VALUE field contains the integrity check value that is output by integrity algorithm or encryption algorithm (i.e., if the integrity algorithm is AUTH_COMBINED the encryption algorithm includes integrity checking capabilities). The size of the integrity check value is defined by the integrity algorithm or encryption algorithm.

If the integrity algorithm specified by the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4) in the IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) in the Key Exchange step (see 5.13.4.8) is AUTH_COMBINED, then the inputs to and computation of the INTEGRITY CHECK VALUE field are the same as the inputs to and computation of the CIPHERTEXT field.

If the integrity algorithm specified by the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor in the IKEv2-SCSI SA Cryptographic Algorithms payload in the Key Exchange step is not AUTH_COMBINED, then the contents of the INTEGRITY CHECK VALUE field are computed by processing the integrity check algorithm specified by the ALGORITHM IDENTIFIER field in the INTEG IKEv2-SCSI cryptographic algorithm descriptor using the following inputs:

a) A byte string composed of:
   1) The AAD, if any, described in 7.7.3.5.10.2;
   2) The contents of the INITIALIZATION VECTOR field (see table x26); and
   3) The plaintext data shown in table x27;
   and
b) The shared key value from one of the following shared keys:
   A) If the Encrypted payload appears in SECURITY PROTOCOL OUT command parameter data, the SK_ai shared key (see 5.13.4.4); or
   B) If the Encrypted payload appears in SECURITY PROTOCOL IN command parameter data, the SK_ar shared key (see 5.13.4.4).

When computing the encrypted CIPHERTEXT field contents for an Encrypted payload, the plaintext shown in table x27 is used.

Table x27 — Plaintext format for Encrypted payload CIPHERTEXT field

Bit / Byte	7	6	5	4	3	2	1	0
IKEv2-SCSI payloads								
0	First IKEv2-SCSI payload (see 7.7.3.5)							
	. . .							
	Last IKEv2-SCSI payload (see 7.7.3.5)							
n								
p	PADDING BYTES (if any)							
k-1								
k	PAD LENGTH (k-p)							

Each IKEv2-SCSI payload (see 7.7.3.5) contains specific data related to the operation being performed. A specific combination of IKEv2-SCSI payloads is needed for each operation (e.g., Authentication) as summarized in 5.13.4.2.

The PADDING BYTES field contains zero to 255 bytes. The number of padding bytes is:

a) Defined by the encryption algorithm; or
b) Any number of bytes that causes the length of all plaintext bytes (i.e., l+2) to be a whole multiple of the cipher block size for the encryption algorithm being used.

The contents of the padding bytes are:

a) Defined by the encryption algorithm; or
b) If the encryption algorithm does not define the padding bytes contents, a series of one byte binary values starting at one and incrementing by one in each successive byte (i.e., 01h in the first padding byte, 02h in the second padding byte, etc.).

If the encryption algorithm does not place requirements on the contents of the padding bytes (i.e., option b) is in effect), then after decryption the contents of the padding bytes shall be verified to match the series of one byte binary values described in this subclause. If this verification is not successful in a device server, the error shall be processed as described in 7.7.3.8.2. If this verification is not successful in an application client, the decrypted data should be ignored.

The PAD LENGTH field contains the number of bytes in the PADDING BYTES field.

7.7.3.5.10.2 AAD for combined mode encryption algorithms

For IKEv2-SCSI, encryption algorithms that also provide integrity checking require AAD as an input to their encryption and decryption functions. The AAD defined by this standard is as follows:

1) All the bytes in the IKEv2-SCSI header (see 7.7.3.4); and
2) All the bytes in the IKEv2-SCSI payload header (see 7.7.3.5.1) of the Encrypted payload (see 7.7.3.5.10).

Encryption algorithms that do not provide integrity checking do not use AAD.

7.7.3.5.10.3 Processing a received Encrypted payload

Before performing any checks of data contained in the CIPHERTEXT field, the contents of the CIPHERTEXT field shall be decrypted and integrity checked based on the contents of the IKE_SA APPLICATION CLIENT SAI field and the IKE_SA DEVICE SERVER SAI field in the IKEv2-SCSI header (see 7.7.3.4) as described in this subclause.

Device server processing of an Encrypted payload in a SECURITY PROTOCOL OUT command parameter list shall proceed as follows:

a) If the following are true:
   A) The SECURITY PROTOCOL SPECIFIC field in the CDB contains 0104h (i.e., Delete operation);
   B) The contents of the IKE_SA APPLICATION CLIENT SAI field match the AC_SAI SA parameter (see 3.1.103) in a generated SA for which IKEv2-SCSI CCS processing has been completed; and
   C) The IKE_SA DEVICE SERVER SAI field match the DS_SAI SA parameter in the same generated SA;
   then the contents of the MGMT_DATA SA parameter shall be used to decrypt and integrity check the Encrypted payload as described in 7.7.3.5.10.4;
b) If the following are true:
   A) If the device server is maintaining state for an IKEv2-SCSI CCS on the I_T_L nexus on which the command was received;

B) The IKEv2-SCSI CCS has completed the Key Exchange step;
C) The SECURITY PROTOCOL SPECIFIC field in the CDB contains 0103h (i.e., Authentication step) or 0104h (i.e., Delete operation);
D) The contents of the IKE_SA APPLICATION CLIENT SAI field match the application client's SAI in the maintained IKEv2-SCSI CCS state; and
E) The contents of the IKE_SA DEVICE SERVER SAI field match the device server's SAI in the maintained IKEv2-SCSI CCS state;

then the payload shall be decrypted as described in 7.7.3.5.10.5; or

c) If the conditions described in a) and b) are not met or if an error is detected while decrypting or integrity checking the CIPHERTEXT field contents, then the error shall be processed as described in 7.7.3.8.2.

If an application client receives an Encrypted payload in a SECURITY PROTOCOL IN command parameter list, then the payload shall be decrypted and integrity checked as described in 7.7.3.5.10.6.

7.7.3.5.10.4 Decrypting an Encrypted payload that is not part of an IKEv2-SCSI CCS

Before performing any checks of data contained in the Encrypted payload received in a SECURITY PROTCOL OUT command parameter list (see 5.13.4.9.2) for a generated SA, the device server shall decrypt and check the integrity of the Encrypted payload as follows:

1) Decrypt the CIPHERTEXT field of the Encrypted payload using the contents of the INITIALIZATION VECTOR field in the Encrypted payload (see 7.7.3.5.10.1), the AAD described in 7.7.3.5.10.2, and following information from the MGMT_DATA_SA parameter (see 5.13.4.11):
   A) Encryption algorithm;
   B) Key length; and
   C) SK_ei shared key (see 5.13.4.4);
2) Integrity check the decrypted data using the integrity algorithm specified in the MGMT_DATA SA parameter as follows:
   A) If the integrity algorithm is not AUTH_COMBINED, then compute an integrity check value for the decrypted data using the specified integrity algorithm and SK_ai shared key (see 5.13.4.4) stored in the MGMT_DATA SA parameter; or
   B) If the integrity algorithm is AUTH_COMBINED, then compute an integrity check value for the decrypted data in combination with decrypting the CIPHERTEXT field in step 1);

   and
3) Verify that the computed integrity check value matches the one contained in the INTEGRITY CHECK VALUE field in the Encrypted payload (see 7.7.3.5.10.1).

If the integrity checking of the Encrypted payload fails, then the SECURITY PROTOCOL OUT command shall be terminated with CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to UNABLE TO DECRYPT PARAMETER LIST. The IKEv2-SCSI CCS state being maintained for the I_T_L nexus on which the command was received, if any, shall not be affected by this error.

{{UNABLE TO DECRYPT PARAMETER LIST is a new additional sense code.}}

7.7.3.5.10.5 Decrypting an Encrypted payload that is part of an IKEv2-SCSI CCS

Before performing any checks of data contained in the Encrypted payload received in an Authentication step or Delete operation SECURITY PROTOCOL OUT parameter list (see 5.13.4.9.2), the device server shall decrypt and check the integrity of the Encrypted payload as follows:

1) Decrypt the CIPHERTEXT field of the Encrypted payload using the contents of the INITIALIZATION VECTOR field in the Encrypted payload (see 7.7.3.5.10.1), the AAD described in 7.7.3.5.10.2, and following information from the MGMT_DATA_SA parameter (see 5.13.4.11):
   A) Encryption algorithm;

B) Key length; and
C) SK_ei shared key (see 5.13.4.4);

2) Integrity check the decrypted data using the integrity algorithm specified in the MGMT_DATA SA parameter as follows:
   A) If the integrity algorithm is not AUTH_COMBINED, then compute an integrity check value for the decrypted data using the specified integrity algorithm and SK_ai shared key (see 5.13.4.4) stored in the MGMT_DATA SA parameter; or
   B) If the integrity algorithm is AUTH_COMBINED, then compute an integrity check value for the decrypted data in combination with decrypting the CIPHERTEXT field in step 1);

   and
3) Verify that the computed integrity check value matches the one contained in the INTEGRITY CHECK VALUE field in the Encrypted payload (see 7.7.3.5.10.1).

If the integrity checking of the Encrypted payload fails, then the error shall be processed as described in 7.7.3.8.2.

7.7.3.5.10.6 Decrypting an Authentication step SECURITY PROTOCOL IN command Encrypted payload

Before performing any checks of data contained in the Encrypted payload received in a in the Authentication step SECURITY PROTOCOL IN parameter list (see 5.13.4.9.3), the application client should decrypt and check the integrity of the Encrypted payload as follows:

1) Decrypt the CIPHERTEXT field of the Encrypted payload using the contents of the INITIALIZATION VECTOR field in the Encrypted payload (see 7.7.3.5.10.1), the AAD described in 7.7.3.5.10.2, and following information from the MGMT_DATA_SA parameter (see 5.13.4.11):
   A) Encryption algorithm;
   B) Key length; and
   C) SK_er shared key (see 5.13.4.4);
2) Integrity check the decrypted data using the integrity algorithm specified in the MGMT_DATA SA parameter as follows:
   A) If the integrity algorithm is not AUTH_COMBINED, then compute an integrity check value for the decrypted data using the specified integrity algorithm and SK_ar shared key (see 5.13.4.4) stored in the MGMT_DATA SA parameter; or
   B) If the integrity algorithm is AUTH_COMBINED, then compute an integrity check value for the decrypted data in combination with decrypting the CIPHERTEXT field in step 1);

   and
3) Verify that the computed integrity check value matches the one contained in the INTEGRITY CHECK VALUE field in the Encrypted payload (see 7.7.3.5.10.1).

If the integrity checking of the Encrypted payload fails, then the application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

7.7.3.5.11 IKEv2-SCSI SA Creation Capabilities payload

The IKEv2-SCSI SA Creation Capabilities payload (see table x28) lists all the security algorithms that the device server allows to be used in an IKEv2-SCSI CCS. Events that are outside the scope of this standard may change the contents of the IKEv2-SCSI SA Creation Capabilities payload at any time.

Table x28 — IKEv2-SCSI SA Creation Capabilities payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD (00h)							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	Reserved							
6								
7	NUMBER OF ALGORITHM DESCRIPTORS							
IKEv2-SCSI cryptographic algorithm descriptors								
8	First IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)							
	. . .							
	Last IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)							
n								

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The NEXT PAYLOAD field is set to zero (i.e., No Next Payload) in the IKEv2-SCSI SA Creation Capabilities payload.

The CRIT bit is set to one in the IKEv2-SCSI SA Creation Capabilities payload.

The NUMBER OF ALGORITHM DESCRIPTORS field contains the number of IKEv2-SCSI cryptographic algorithm descriptors that follow in the IKEv2-SCSI SA Creation Capabilities payload.

Each IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6) describes one combination of security algorithm and algorithm attributes that the device server allows to be used in an IKEv2-SCSI CCS. If more than one set of algorithm attributes (e.g., key length) is allowed for any allowed security algorithm, a different SCSI cryptographic algorithms descriptor shall be included for each set of algorithm attributes.

The SCSI cryptographic algorithms descriptors shall be ordered by:

1) Increasing algorithm type;
2) Increasing algorithm identifier within the same algorithm type; and
3) Increasing key length, if any, within the same algorithm identifier.

The algorithms allowed may be a subset of the algorithms supported by the device server.

The method for changing which of the device server supported algorithms are allowed is outside the scope of this standard, but changes in allowed algorithms do not take effect until the new list is returned to any application client in an IKEv2-SCSI SA Creation Capabilities payload.

7.7.3.5.12 IKEv2-SCSI SA Cryptographic Algorithms payload

The IKEv2-SCSI SA Cryptographic Algorithms payload (see table x29) lists the security algorithms that are being used in the creation and deletion of an SA using an IKEv2-SCSI CCS.

Table x29 — IKEv2-SCSI SA Cryptographic Algorithms payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4		Reserved						
5								
6	(MSB)	USAGE DATA LENGTH (0000h)						
7								(LSB)
8	(MSB)	SAI						
15								(LSB)
16		Reserved						
18								
19	NUMBER OF ALGORITHM DESCRIPTORS							
IKEv2-SCSI cryptographic algorithm descriptors								
20	First IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)							
	. . .							
n	Last IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)							

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the IKEv2-SCSI SA Cryptographic Algorithms payload.

The USAGE DATA LENGTH field is set to zero in the IKEv2-SCSI SA Cryptographic Algorithms payload.

The SAI field is reserved.

The NUMBER OF ALGORITHM DESCRIPTORS field contains the number of IKEv2-SCSI cryptographic algorithm descriptors that follow in the IKEv2-SCSI SA Cryptographic Algorithms payload. If a device server receives a NUMBER OF ALGORITHM DESCRIPTORS field that does not contain five, then the error shall be processed as described in 7.7.3.8.3.

Each IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6) describes one combination of security algorithm and algorithm attributes to be used during the IKEv2-SCSI CCS.

The IKEv2-SCSI cryptographic algorithm descriptors are ordered as follows:

1) One ENCR IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.2);
2) One PRF IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.3);
3) One INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4);
4) One D-H IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.5);
5) One SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.6).
6) One SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.6).

If a device server receives an IKEv2-SCSI SA Cryptographic Algorithms payload that does not contain the IKEv2-SCSI cryptographic algorithm descriptors described in this subclause in the order described in this subclause, then the error shall be processed as described in 7.7.3.8.3.

If the device server receives an IKEv2-SCSI SA Cryptographic Algorithms payload that contains an ENCR IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to ENCR_NULL, then the error shall be processed as described in 7.7.3.8.3.

In the Key Exchange step SECURITY PROTOCOL IN parameter data (see 5.13.4.8.3), the device server returns the IKEv2-SCSI SA Cryptographic Algorithms payload received during the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) to confirm acceptance of the algorithms.

7.7.3.5.13 IKEv2-SCSI UT Cryptographic Algorithms payload

The IKEv2-SCSI UT Cryptographic Algorithms payload (see table x30) lists the usage type of and security algorithms to be used by the SA that is created as a result of an IKEv2-SCSI CCS.

Table x30 — IKEv2-SCSI UT Cryptographic Algorithms payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (n+1)						
3								(LSB)
4	(MSB)	SAI						
11								(LSB)
12	(MSB)	SA TYPE						
13								(LSB)
14	(MSB)	USAGE DATA LENGTH (j)						
15								(LSB)
16		USAGE DATA						
16+j-1								
16+j		Reserved						
16+j+2								
16+j+3	NUMBER OF ALGORITHM DESCRIPTORS							
IKEv2-SCSI cryptographic algorithm descriptors								
16+j+4		First IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)						
		. . .						
n		Last IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6)						

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the IKEv2-SCSI UT Cryptographic Algorithms payload.

The SA TYPE field specifies the usage type for the SA and is selected from among those listed in table 45 (see 5.13.2.2). If a device server receives an SA TYPE field that contains an SA usage type whose use the device server does not allow, then the error shall be processed as described in 7.7.3.8.3.

The method for changing which of the device server supported SA usage types are allowed is outside the scope of this standard.

The USAGE DATA LENGTH field specifies number of bytes of usage data that follow.

NOTE x11 - The contents of the USAGE DATA LENGTH field differ from those found in most SCSI length fields, however, they are consistent with the IKEv2 usage (see RFC 4306).

The SAI field is reserved.

The USAGE DATA field contains information to be stored in the USAGE_DATA SA parameter (see 3.1.103) if the SA is generated (see 5.13.4.11).

The NUMBER OF ALGORITHM DESCRIPTORS field contains the number of IKEv2-SCSI cryptographic algorithm descriptors that follow in the IKEv2-SCSI UT Cryptographic Algorithms payload. If a device server receives a NUMBER OF ALGORITHM DESCRIPTORS field that contains a value other than two, then the error shall be processed as described in 7.7.3.8.3.

Each IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6) describes one combination of security algorithm and algorithm attributes to be used by the SA created as a result of the IKEv2-SCSI CCS. The IKEv2-SCSI cryptographic algorithm descriptors are ordered as follows:

1) One ENCR IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.2); and
2) One INTEG IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.4).

If a device server receives an IKEv2-SCSI UT Cryptographic Algorithms payload that does not contain the IKEv2-SCSI cryptographic algorithm descriptors described in this subclause in the order described in this subclause, then the error shall be processed as described in 7.7.3.8.3.

7.7.3.5.14 IKEv2-SCSI Timeout Values payload

The IKEv2-SCSI Timeout Values payload (see table x31) specifies the timeout intervals associated with an IKEv2-SCSI CCS

Table x31 — IKEv2-SCSI Timeout Values payload format

Bit / Byte	7	6	5	4	3	2	1	0
0	NEXT PAYLOAD							
1	CRIT (1b)	Reserved						
2	(MSB)	IKE PAYLOAD LENGTH (10h)						
3								(LSB)
4	Reserved							
6								
7	NUMBER OF TIMEOUT VALUES (02h)							
8	(MSB)	IKEV2-SCSI PROTOCOL TIMEOUT						
11								(LSB)
12	(MSB)	IKEV2-SCSI SA INACTIVITY TIMEOUT						
15								(LSB)

The NEXT PAYLOAD field, CRIT bit, and IKE PAYLOAD LENGTH field are described in 7.7.3.5.1.

The CRIT bit is set to one in the IKEv2-SCSI Timeout Values payload.

The NUMBER OF TIMEOUT VALUES field specifies the number of four-byte timeout values that follow. If the number of timeout values is less than two, then the IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3.

The IKEV2-SCSI PROTOCOL TIMEOUT field specifies the number of seconds that the device server shall wait for the next command in the IKEv2-SCSI CCS. If the timeout expires before the device server receives an IKEv2-SCSI CCS command, the device server shall abandon the IKEv2-SCSI CCS as described in 5.13.4.12.

The IKEV2-SCSI SA INACTIVITY TIMEOUT field specifies the number of seconds that the device server shall wait for the next command that uses an SA. This value is copied to the TIMEOUT SA parameter when the SA is generated (see 5.13.4.11).

The device server shall replace any timeout value that is set to zero with a value of ten (i.e., ten seconds).

The maximum value for the protocol timeout should be long enough to allow the application client to continue the IKEv2-SCSI CCS, but short enough that if an incomplete IKEv2-SCSI CCS is abandoned, the device server discards the state for that IKEv2-SCSI CCS and becomes available to for another IKEv2-SCSI CCS without excessive delay.

7.7.3.6 IKEv2-SCSI cryptographic algorithm descriptors

7.7.3.6.1 Overview

Each IKEv2-SCSI cryptographic algorithm descriptor (see table x32) specifies one algorithm used for encryption, integrity checking, key generation, or authentication.

Table x32 — IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE							
1	Reserved							
2	(MSB)			IKE DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			ALGORITHM IDENTIFIER				
7								(LSB)
8				ALGORITHM ATTRIBUTES				
11								

The ALGORITHM TYPE field (see table x33) specifies the type of cryptographic algorithm to which the IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x33 — ALGORITHM TYPE field

Code	Name	Description	Reference
01h	ENCR	Encryption algorithm	7.7.3.6.2
02h	PRF	Pseudo-random function	7.7.3.6.3
03h	INTEG	Integrity algorithm	7.7.3.6.4
04h	D-H	Diffie-Hellman group	7.7.3.6.5
05h - F0h	Restricted		RFC 4306
F9h	SA_AUTH_OUT	IKEv2-SCSI authentication algorithm for SECURITY PROTOCOL OUT data	7.7.3.6.6
FAh	SA_AUTH_IN	IKEv2-SCSI authentication algorithm for SECURITY PROTOCOL IN data	7.7.3.6.6
All others	Reserved		

The IKE DESCRIPTOR LENGTH field contains 12 (i.e., the total number of bytes in the IKEv2-SCSI SA Cryptographic Algorithms descriptor including the ALGORITHM TYPE field and reserved byte).

NOTE x12 - The contents of the IKE DESCRIPTOR LENGTH field differ from those found in most SCSI length fields, however, they are consistent with the IKEv2 usage (see RFC 4306).

The contents of the ALGORITHM IDENTIFIER field and ALGORITHM ATTRIBUTES field depend on the contents of the ALGORITHM TYPE field (see table x33). The ALGORITHM ATTRIBUTES field is reserved in some IKEv2-SCSI SA Cryptographic Algorithms descriptor formats.

7.7.3.6.2 Encryption algorithm (ENCR) IKEv2-SCSI cryptographic algorithm descriptors

When the ALGORITHM TYPE field is set to ENCR (i.e., 01h) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x34), the descriptor specifies an encryption algorithm to be applied during the IKEv2-SCSI Authentication step (see 5.13.4.9) and when the SA created by the IKEv2-SCSI is applied to user data.

Table x34 — ENCR IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE (01h)							
1	Reserved							
2	(MSB)							
3	IKE DESCRIPTOR LENGTH (000Ch)							(LSB)
4	(MSB)							
7	ALGORITHM IDENTIFIER							(LSB)
8								
9	Reserved							
10	(MSB)							
11	KEY LENGTH							(LSB)

The ALGORITHM TYPE field and IKE DESCRIPTOR LENGTH field are described in 7.7.3.6.1.

The ALGORITHM IDENTIFIER field (see table x35) specifies the encryption algorithm to which the ENCR IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x35 — ENCR ALGORITHM IDENTIFIER field (page 1 of 2)

Code	Description	Salt [a] length (bytes)	Key length (bytes)	Support	Reference
0001 000Bh	ENCR_NULL [b]	n/a	0	Mandatory	
0001 000Ch	AES-CBC [b]	n/a	16	Optional	RFC 3602
			24	Prohibited	
			32	Optional	

[a] See RFC 4106 and RFC 4309.

[b] If the INTEG cryptographic algorithm descriptor (see 7.7.3.6.4) in the same IKEv2-SCSI SA Cryptographic Algorithms payload or the same IKEv2-SCSI SA Cryptographic Algorithms payload as this ENCR cryptographic algorithm descriptor has the ALGORITHM IDENTIFIER field set to AUTH_COMBINED, then the error shall be processed as described in 7.7.3.8.3.

[c] If the INTEG cryptographic algorithm descriptor (see 7.7.3.6.4) in the same IKEv2-SCSI SA Cryptographic Algorithms payload or the same IKEv2-SCSI SA Cryptographic Algorithms payload as this ENCR cryptographic algorithm descriptor does not have the ALGORITHM IDENTIFIER field set to AUTH_COMBINED, then the error shall be processed as described in 7.7.3.8.3.

Table x35 — ENCR ALGORITHM IDENTIFIER field (page 2 of 2)

Code	Description	Salt [a] length (bytes)	Key length (bytes)	Support	Reference
0001 0010h	AES-CCM with a 16 byte MAC [c]	3	16	Optional	RFC 4309
			24	Prohibited	
			32	Optional	
0001 0014h	AES-GCM with a 16 byte MAC [c]	4	16	Optional	RFC 4106
			24	Prohibited	
			32	Optional	
0001 0400h – 0001 FFFFh	Vendor Specific				
0000 0000h – 0000 FFFFh	Restricted				IANA
All others	Reserved				

[a] See RFC 4106 and RFC 4309.

[b] If the INTEG cryptographic algorithm descriptor (see 7.7.3.6.4) in the same IKEv2-SCSI SA Cryptographic Algorithms payload or the same IKEv2-SCSI SA Cryptographic Algorithms payload as this ENCR cryptographic algorithm descriptor has the ALGORITHM IDENTIFIER field set to AUTH_COMBINED, then the error shall be processed as described in 7.7.3.8.3.

[c] If the INTEG cryptographic algorithm descriptor (see 7.7.3.6.4) in the same IKEv2-SCSI SA Cryptographic Algorithms payload or the same IKEv2-SCSI SA Cryptographic Algorithms payload as this ENCR cryptographic algorithm descriptor does not have the ALGORITHM IDENTIFIER field set to AUTH_COMBINED, then the error shall be processed as described in 7.7.3.8.3.

ENCR_NULL indicates that encryption is not to be applied when the SA created by the IKEv2-SCSI is applied to user data.

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains an IKEv2-SCSI SA Cryptographic Algorithms payload (see 7.7.3.5.12) that contains:

a) An ENCR IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to ENCR_NULL;
b) The following combination of IKEv2-SCSI cryptographic algorithm descriptors (see 7.7.3.6.4):
   A) An INTEG IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to a value other than AUTH_COMBINED; and
   B) An ENCR IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to a value that table x35 describes as requiring AUTH_COMBINED as the integrity check algorithm;

   or
c) The following combination of IKEv2-SCSI cryptographic algorithm descriptors:
   A) An INTEG IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to AUTH_COMBINED; and
   B) An ENCR IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to a value that table x35 does not describe as requiring AUTH_COMBINED as the integrity check algorithm.

The KEY LENGTH field specifies the number of bytes in the shared key (see 3.1.w) for the encryption algorithm to which the ENCR IKEv2-SCSI cryptographic algorithm descriptor applies.

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains:

a) No ENCR IKEv2-SCSI cryptographic algorithm descriptors;
b) More than one ENCR IKEv2-SCSI cryptographic algorithm descriptor;
c) An ENCR IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload (see 7.7.3.5.11) last returned by the device server to any application client; or
d) An ENCR IKEv2-SCSI cryptographic algorithm descriptor that contains:
   A) An algorithm identifier that is not shown in table x35; or
   B) A key length that:
      a) Does not match one of the values shown in table x35; or
      b) Is not supported by the device server.

7.7.3.6.3 Pseudo-random function (PRF) IKEv2-SCSI cryptographic algorithm descriptors

When the algorithm type field is set to PRF (i.e., 02h) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x36), the descriptor specifies the pseudo-random function and KDF to be used during the Key Exchange step completion (see 5.13.4.8.4).

Table x36 — PRF IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE (02h)							
1	Reserved							
2	(MSB)			IKE DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			ALGORITHM IDENTIFIER				
7								(LSB)
8	Reserved							
11								

The ALGORITHM TYPE field and IKE DESCRIPTOR LENGTH field are described in 7.7.3.6.1.

The ALGORITHM IDENTIFIER field (see table x37) specifies PRF and KDF to which the PRF IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x37 — PRF ALGORITHM IDENTIFIER field

Code	Description	Support	Output length (bytes)	Reference	
				PRF [a]	KDF [b]
0002 0002h	IKEv2-use based on SHA-1	Optional	20	RFC 2104	5.13.3.3
0002 0004h	IKEv2-use based on AES-128 in CBC mode	Optional	16	RFC 4434	5.13.3.4
0002 0005h	IKEv2-use based on SHA-256	Optional	32	RFC 4868	5.13.3.3
0002 0007h	IKEv2-use based on SHA-512	Optional	64	RFC 4868	5.13.3.3
0002 0400h – 0002 FFFFh	Vendor Specific				
0000 0000h – 0000 FFFFh	Restricted			IANA	
All others	Reserved				

[a] PRFs are equivalent to the prf() functions defined in RFC 4306.
[b] KDFs are equivalent to the prf+() functions defined in RFC 4306.

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains:

a) No PRF IKEv2-SCSI cryptographic algorithm descriptors;
b) More than one PRF IKEv2-SCSI cryptographic algorithm descriptor;
c) A PRF IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload (see 7.7.3.5.11) last returned by the device server to any application client; or
d) An PRF IKEv2-SCSI cryptographic algorithm descriptor that contains an algorithm identifier that is not shown in table x37.

7.7.3.6.4 Integrity algorithm (INTEG) IKEv2-SCSI cryptographic algorithm descriptors

When the algorithm type field is set to INTEG (i.e., 03h) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x38), the descriptor specifies an integrity checking (i.e., data authentication) algorithm to be applied during the IKEv2-SCSI Authentication step (see 5.13.4.9) and when the SA created by the IKEv2-SCSI is applied to user data.

Table x38 — INTEG IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE (03h)							
1	Reserved							
2	(MSB)			IKE DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			ALGORITHM IDENTIFIER				
7								(LSB)
8				Reserved				
11								

The ALGORITHM TYPE field and IKE DESCRIPTOR LENGTH field are described in 7.7.3.6.1.

The ALGORITHM IDENTIFIER field (see table x39) specifies integrity checking algorithm and shared key length to which the INTEG IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x39 — INTEG ALGORITHM IDENTIFIER field

Code	IKEv2 Name	Key length (bytes)	Support	Reference
0003 0002h	AUTH_HMAC_SHA1_96	20	Optional	RFC 2404
0003 000Ch	AUTH_HMAC_SHA2_256_128	32	Optional	RFC 4868
0003 000Eh	AUTH_HMAC_SHA2_512_256	64	Optional	RFC 4868
F003 0001h	AUTH_COMBINED	0	Optional	this subclause
0003 0400h – 0003 FFFFh	Vendor Specific			
0000 0000h – 0000 FFFFh	Restricted			IANA
All others	Reserved			

The AUTH_COMBINED integrity checking algorithm is used with encryption algorithms that include integrity checking as described in 7.7.3.6.2. The AUTH_COMBINED algorithm identifier specifies that no additional integrity check is performed, as indicated by the zero-length key.

The key length used with an integrity checking algorithm is determined by the algorithm identifier as shown in table x39.

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains:

a) No INTEG IKEv2-SCSI cryptographic algorithm descriptors;
b) More than one INTEG IKEv2-SCSI cryptographic algorithm descriptor;
c) An INTEG IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload (see 7.7.3.5.11) last returned by the device server to any application client; or
d) An INTEG IKEv2-SCSI cryptographic algorithm descriptor that contains an algorithm identifier that is not shown in table x39.

7.7.3.6.5 Diffie-Hellman group (D-H) IKEv2-SCSI cryptographic algorithm descriptors

When the algorithm type field is set to D-H (i.e., 04h) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x40), the descriptor specifies Diffie-Hellman group and Diffie-Hellman algorithm used during the IKEv2-SCSI Key Exchange step (see 5.13.4.8) to derive a shared key (see 3.1.w) that is known only to the application client and device server.

Table x40 — D-H IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE (04h)							
1	Reserved							
2	(MSB)			IKE DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			ALGORITHM IDENTIFIER				
7								(LSB)
8	Reserved							
11								

The ALGORITHM TYPE field and IKE DESCRIPTOR LENGTH field are described in 7.7.3.6.1.

The ALGORITHM IDENTIFIER field (see table x41) specifies Diffie-Hellman algorithm, group, and shared key length to which the D-H IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x41 — D-H ALGORITHM IDENTIFIER field

Code	Description	Key length (bytes)	Support	Reference
0004 000Eh	2 048-bit MODP group (finite field D-H)	256	Optional	RFC 3526
0004 000Fh	3 072-bit MODP group (finite field D-H)	384	Optional	RFC 3526
0004 0010h	4 096-bit MODP group (finite field D-H)	512	Optional	RFC 3526
0004 0013h	256-bit random ECP group	64	Optional	RFC 4753
0004 0015h	521-bit random ECP group	132	Optional	RFC 4753
0004 0400h – 0004 FFFFh	Vendor Specific			
0000 0000h – 0000 FFFFh	Restricted			IANA
All others	Reserved			

The key length of the public value transferred in the KEY EXCHANGE DATA field (see 7.7.3.5.3) is determined by the algorithm identifier as shown in table x41.

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains:

a) No D-H IKEv2-SCSI cryptographic algorithm descriptors;
b) More than one D-H IKEv2-SCSI cryptographic algorithm descriptor;
c) A D-H IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload (see 7.7.3.5.11) last returned by the device server to any application client; or
d) An D-H IKEv2-SCSI cryptographic algorithm descriptor that contains an algorithm identifier that is not shown in table x41.

7.7.3.6.6 IKEv2-SCSI authentication algorithm IKEv2-SCSI cryptographic algorithm descriptors

When the algorithm type field is set to SA_AUTH_OUT (i.e., F9h) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x42), the descriptor specifies Authentication payload authentication algorithm used by the IKEv2-SCSI Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2).

When the algorithm type field is set to SA_AUTH_IN (i.e., FAh) in an IKEv2-SCSI cryptographic algorithm descriptor (see table x42), the descriptor specifies Authentication payload authentication algorithm used by the IKEv2-SCSI Authentication step SECURITY PROTOCOL IN command (see 5.13.4.9.3).

Table x42 — SA_AUTH_OUT and SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	ALGORITHM TYPE (F9h or FAh)							
1	Reserved							
2	(MSB)			IKE DESCRIPTOR LENGTH (000Ch)				
3								(LSB)
4	(MSB)			ALGORITHM IDENTIFIER				
7								(LSB)
8	Reserved							
11								

The ALGORITHM TYPE field and IKE DESCRIPTOR LENGTH field are described in 7.7.3.6.1.

The ALGORITHM IDENTIFIER field (see table x43) specifies Authentication payload authentication algorithm to which the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor or SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor applies.

Table x43 — SA_AUTH_OUT and SA_AUTH_IN ALGORITHM IDENTIFIER field

Code	Description	Support	Reference
00F9 0000h	SA_AUTH_NONE	Optional	this subclause
00F9 0001h	RSA Digital Signature [a]	Optional	RFC 4306
00F9 0002h	Shared Key Message Integrity Code	Optional	RFC 4306 [b, c]
00F9 0009h	ECDSA with SHA-256 on the P-256 curve [a]	Optional	RFC 4754
00F9 000Bh	ECDSA with SHA-512 on the P-521 curve [a]	Optional	RFC 4754
00F9 0400h – 00F9 FFFFh	Vendor Specific	Optional	
0000 0000h – 0000 FFFFh	Restricted	Prohibited	IANA
All others	Reserved		

[a] Use of certificates with this digital signature authentication algorithm is optional.

[b] The 17 ASCII character non-terminated pre-shared key (see 3.1.k) pad string "Key Pad for IKEv2" specified by RFC 4306 is replaced by the 22 ASCII character non-terminated pre-shared key pad string "Key Pad for IKEv2-SCSI".

[c] The pre-shared key (see 3.1.k) requirements used by this standard (see 5.13.4.5) apply in addition to those found in RFC 4306.

SA_AUTH_NONE specifies the omission of the IKEv2-SCSI Authentication step (see 5.13.4.9) as follows:

a) The presence of an SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and an SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE is an SA Creation Capabilities payload (see 7.7.3.5.11) indicates that the device server is allowed to negotiate the omission of the IKEv2-SCSI Authentication step; and
b) The presence of an SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and an SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE is an IKEv2-SCSI Cryptographic Algorithms payload (see 7.7.3.5.12) indicates the following based upon the command whose parameter data carries the payload:
   A) In the parameter list for a Key Exchange SECURITY PROTOCOL OUT command (see 5.13.4.8.2), SA_AUTH_NONE specifies that the application client is requesting that the IKEv2-SCSI Authentication step be skipped; and
   B) In the parameter data for a Key Exchange SECURITY PROTOCOL IN command (see 5.13.4.8.3), SA_AUTH_NONE indicates that the device server has agreed to skip the IKEv2-SCSI Authentication step.

If it is reported by a device server in its capabilities and selected by an application client, then the IKEv2-SCSI Authentication step is skipped and the resulting SAs are not authenticated.

An SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE shall not appear in an IKEv2-SCSI Cryptographic Algorithms payload except as described in 5.13.4.6. An SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE shall not appear in an IKEv2-SCSI Cryptographic Algorithms payload except as described in 5.13.4.6.

The Shared Key Message Integrity Code is based on a pre-shared key (see 3.1.k and 5.13.4.5) that is associated with the identity in the Identification payload (see 7.7.3.5.4).

The IKEv2-SCSI CCS state maintained for the I_T_L nexus shall be abandoned as described in 7.7.3.8.3 if the parameter list contains:

a) No SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptors;
b) No SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptors;
c) More than one SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor;
d) More than one SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor;
e) An SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload (see 7.7.3.5.11) last returned by the device server to any application client;
f) An SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor that does not appear in the SA Creations Capabilities payload last returned by the device server to any application client
g) An SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor that contains an algorithm identifier that is not shown in table x43;
h) An SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor that contains an algorithm identifier that is not shown in table x43;
i) An SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE, and an SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to a value other than SA_AUTH_NONE; or
j) An SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE, and an SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor with the ALGORITHM IDENTIFIER field set to a value other than SA_AUTH_NONE.

7.7.3.7 Errors in IKEv2-SCSI security protocol commands

For a single I_T_L nexus, the device server shall ensure that the two or four IKEv2-SCSI CCS commands are processed in the order described in 5.13.4.1 based only on the contents of the CDB (i.e., the SECURITY PROTOCOL OUT parameter data shall not be processed unless the tests in table x44 specify the processing of the command) using the tests and responses shown in table x44.

Editors Note 4 - ROW: The 'Before Key Exchange step SECURITY PROTOCOL OUT command returns GOOD status' column in table x44 has been completely redone in r3. It is impossible to properly mark this with a change bar. The column needs to be reviewed carefully. The description of commands for which processing may be repeated also is updated, but these changes are marked with change bars.

Table x44 — IKEv2-SCSI command ordering processing requirements on a single I_T_L nexus (page 1 of 2)

IKEv2-SCSI SECURITY PROTOCOL OUT command or SECURITY PROTOCOL IN command received	Time → Before Key Exchange step SECURITY PROTOCOL OUT command returns GOOD status	After a Key Exchange step SECURITY PROTOCOL OUT command returns GOOD status	After a Key Exchange step SECURITY PROTOCOL IN command returns GOOD status	After an Authentication step SECURITY PROTOCOL OUT command returns GOOD status	After an Authentication step SECURITY PROTOCOL IN command returns GOOD status
Key Exchange step SECURITY PROTOCOL OUT command	Process the command as described in this standard	Do not process the command [a]	Do not process the command [a]	Do not process the command [a]	Same as before Key Exchange step SECURITY PROTOCOL OUT command
Key Exchange step SECURITY PROTOCOL IN command	No IKEv2-SCSI CCS exists [b]	Process the command as described in this standard	Repeat processing of the command [c]	Do not process the command [a]	Do not process the command [a]

[a] The command shall be terminated and the IKEv2-SCSI CCS shall be continued as follows:
a) The command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS; and
b) The device server shall continue the IKEv2-SCSI CCS by preparing to receive another IKEv2-SCSI CCS SECURITY PROTOCOL OUT command.

[b] The command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

[c] Processing of the SECURITY PROTOCOL IN commands in an IKEv2-SCSI CCS may be repeated. The device server should save the information necessary to repeat processing of these commands until the number of seconds specified in the IKEV2-SCSI PROTOCOL TIMEOUT field of the IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14) have elapsed since the processing of the Authentication step SECURITY PROTOCOL OUT command that returned GOOD status.

Table x44 — IKEv2-SCSI command ordering processing requirements on a single I_T_L nexus (page 2 of 2)

IKEv2-SCSI SECURITY PROTOCOL OUT command or SECURITY PROTOCOL IN command received	Time → Before Key Exchange step SECURITY PROTOCOL OUT command returns GOOD status	After a Key Exchange step SECURITY PROTOCOL OUT command returns GOOD status	After a Key Exchange step SECURITY PROTOCOL IN command returns GOOD status	After an Authentication step SECURITY PROTOCOL OUT command returns GOOD status	After an Authentication step SECURITY PROTOCOL IN command returns GOOD status
Authentication step SECURITY PROTOCOL OUT command	No IKEv2-SCSI CCS exists [b]	Do not process the command [a]	Process the command as described in this standard	Do not process the command [a]	Do not process the command [a]
Authentication step SECURITY PROTOCOL IN command		Do not process the command [a]	Do not process the command [a]	Process the command as described in this standard	Repeat processing of the command [c]
Command with an invalid field in the CDB	No IKEv2-SCSI CCS exists [b]	Do not process the command [a]	Do not process the command [a]	Do not process the command [a]	No IKEv2-SCSI CCS exists [b]

[a] The command shall be terminated and the IKEv2-SCSI CCS shall be continued as follows:
a) The command shall be terminated with CHECK CONDITION status, with the sense key set to NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS; and
b) The device server shall continue the IKEv2-SCSI CCS by preparing to receive another IKEv2-SCSI CCS SECURITY PROTOCOL OUT command.

[b] The command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to INVALID FIELD IN CDB.

[c] Processing of the SECURITY PROTOCOL IN commands in an IKEv2-SCSI CCS may be repeated. The device server should save the information necessary to repeat processing of these commands until the number of seconds specified in the IKEV2-SCSI PROTOCOL TIMEOUT field of the IKEv2-SCSI Timeout Values payload (see 7.7.3.5.14) have elapsed since the processing of the Authentication step SECURITY PROTOCOL OUT command that returned GOOD status.

The processing shown in table x44 shall be performed before any other error handling described in this subclause.

7.7.3.8 Errors in IKEv2-SCSI security protocol parameter data

7.7.3.8.1 Overview

Errors in the parameter data transferred to the device sever by an IKEv2-SCSI SECURITY PROTOCOL OUT command are classified (see table x45) based on the ease with which they may be used to mount denial of service attacks against IKEv2-SCSI SA creation operations by an attacker that has not participated as the application client or device server in the Key Exchange step SECURITY PROTOCOL OUT command (see 5.13.4.8.2) that started the IKEv2-SCSI CCS.

Table x45 — IKEv2-SCSI parameter error categories

Denial of service attack potential	Applicable SECURITY PROTOCOL OUT commands	Error handling description	Reference
High [a]	Authentication step, and Delete operation	If possible, the IKEv2-SCSI CCS state maintained for an I_T_L nexus is not changed	7.7.3.8.2
Minimal [b]	Key Exchange step, Authentication step, and Delete operation	The IKEv2-SCSI CCS state maintained for an I_T_L nexus is abandoned	7.7.3.8.3

[a] Attacks capable of causing significant harm by sending a malformed IKEv2-SCSI SECURITY PROTOCOL OUT command to the device server.

[b] Collusive attacks (i.e., attacks that require knowledge of the IKEv2-SCSI CCS shared keys and participation in the IKEv2-SCSI CCS) or attacks that produce no significant harm. Collusive attacks depend on collusion between the attacker and the application client (i.e., require the application client to act against its own best interests). Abandoning the IKEv2-SCSI CCS is a justified response to such attacks.

7.7.3.8.2 Errors with high denial of service attack potential

Errors detected before or during the decryption and integrity checking of an Encrypted payload in an Authentication step SECURITY PROTOCOL OUT command or a Delete operation SECURITY PROTOCOL OUT command have a high potential for being a denial of service attack against one or more application clients (see table x45 in 7.7.3.8.1).

The device server shall respond to these SECURITY PROTOCOL OUT command errors as follows:

a) The command shall be terminated with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to SA CREATION PARAMETER VALUE REJECTED; and
b) The device server shall continue the IKEv2-SCSI CCS, if any, by preparing to receive an Authentication step SECURITY PROTOCOL OUT command.

If a specific field is the cause for returning the SA CREATION PARAMETER VALUE REJECTED additional sense code, then the SKSV bit may be set to one and SENSE KEY SPECIFIC field shall be set as defined in 4.5.2.4.2.

{{SA CREATION PARAMETER VALUE REJECTED is a new additional sense code.}}

7.7.3.8.3 Errors with minimal denial of service attack potential

The errors that have minimal denial of service attack potential for IKEv2-SCSI SA creation (see table x45 in 7.7.3.8.1) are:

a) All errors detected for a Key Exchange step SECURITY PROTOCOL OUT command or its associated parameter data (e.g., errors detected in the IKEv2-SCSI header) that is attempting to establish a new IKEv2-SCSI CCS on an I_T_L nexus; and
b) All errors that are detected after the Encrypted payload has been successfully decrypted and integrity checked in an Authentication step SECURITY PROTOCOL OUT command (see 5.13.4.9.2) or a Delete operation SECURITY PROTOCOL OUT command (see 5.13.4.13);

The device server shall respond to these SECURITY PROTOCOL OUT command errors by terminating the command with a CHECK CONDITION status, with the sense key set to ILLEGAL REQUEST, and the additional sense code set to SA CREATION PARAMETER VALUE INVALID. The state being maintained for an IKEv2-SCSI CCS on the I_T_L nexus on which the command was received, if any, shall be abandoned (see 5.13.4.12).

If a specific field is the cause for returning the SA CREATION PARAMETER VALUE INVALID additional sense code, then the SKSV bit shall be set to one and SENSE KEY SPECIFIC field shall be set as defined in 4.5.2.4.2.

{{SA CREATION PARAMETER VALUE INVALID is a new additional sense code.}}

7.7.3.9 Translating IKEv2 errors

IKEv2 (see RFC 4306) defines an error reporting mechanism based on the Notify payload. This standard translates such error reports into the device server and application actions defined in this subclause.

If a device server is required by IKEv2 to report an error using a Notify payload, the device server shall translate error into a CHECK CONDITION status with the sense key and additional sense code shown in table x46. The device server shall terminate the SECURITY PROTOCOL OUT command (see 6.30) that transferred the parameter list in which the IKE requirements for one or more payloads (see 7.7.3.5) require use of the Notify payload to report an error. The SECURITY PROTOCOL OUT command shall be terminated as described in this subclause. The device server shall not report the IKE errors described in this subclause by terminating a SECURITY PROTOCOL IN command (see 6.29) with a CHECK CONDITION status.

Table x46 — IKEv2 Notify payload error translations for IKEv2-SCSI

<table>
<tr><th colspan="2">IKEv2 (see RFC 4306)</th><th colspan="2">IKEv2-SCSI</th></tr>
<tr><th>Error Type</th><th>Description</th><th>Additional sense code</th><th>Sense key</th></tr>
<tr><td>0000h</td><td colspan="3">Reserved</td></tr>
<tr><td>0001h</td><td>UNSUPPORTED_CRITICAL_PAYLOAD</td><td>SA CREATION PARAMETER NOT SUPPORTED</td><td rowspan="9">ILLEGAL REQUEST</td></tr>
<tr><td>0004h</td><td>INVALID_IKE_SPI</td><td rowspan="4">SA CREATION PARAMETER VALUE INVALID
or
SA CREATION PARAMETER VALUE REJECTED</td></tr>
<tr><td>0005h</td><td>INVALID_MAJOR_VERSION</td></tr>
<tr><td>0007h</td><td>INVALID_SYNTAX [a]</td></tr>
<tr><td>0009h</td><td>INVALID_MESSAGE_ID</td></tr>
<tr><td>000Bh</td><td>INVALID_SPI</td><td>SA CREATION PARAMETER VALUE INVALID [b]</td></tr>
<tr><td>000Eh</td><td>NO_PROPOSAL_CHOSEN [c]</td><td rowspan="2">SA CREATION PARAMETER VALUE INVALID</td></tr>
<tr><td>0011h</td><td>INVALID_KE_PAYLOAD [c]</td></tr>
<tr><td>0018h</td><td>AUTHENTICATION_FAILED</td><td>AUTHENTICATION FAILED</td><td>ABORTED COMMAND</td></tr>
<tr><td>0022h - 0027h</td><td>See RFC 4306 [d]</td><td>n/a</td><td>n/a</td></tr>
<tr><td>2000h - 3FFFh</td><td colspan="3">Vendor Specific</td></tr>
<tr><td>All others</td><td colspan="3">Restricted</td></tr>
</table>

[a] This sense key and one of the additional sense codes shown shall be returned for a syntax error within an Encrypted payload (see 7.7.3.5.10) regardless of conflicting IKEv2 requirements.

[b] SA CREATION PARAMETER VALUE INVALID shall be used for an invalid SAID in an IKEv2-SCSI SECURITY PROTOCOL IN or SECURITY PROTOCOL OUT. The additional sense code for an invalid SAID in all other commands is specified by the appropriate command standard (see 3.1.17).

[c] An application client recovers by restarting processing with the Device Capabilities step (see 5.13.4.7) to rediscover the device server's capabilities.

[d] These IKEv2 Error Types are used for features that are not supported by IKEv2-SCSI SA creation.

{{All additional sense codes in table x46 are new.}}

If an application client detects an IKEv2 error that RFC 4306 requires to be reported with a Notify payload, the application client then the application client should abandon the IKEv2-SCSI CCS and notify the device server that it is abandoning the IKEv2-SCSI CCS as described in 5.13.4.12.

{{New parameter data subclause text ends here. Additions/deletions markups resume.}}

…

8.3.3.1 ACCESS CONTROL OUT introduction

The service actions of the ACCESS CONTROL OUT command (see table 420) are used to request service actions by the access controls coordinator to limit or grant access to the logical units by initiator ports. If the ACCESS CONTROL OUT command is implemented, the ACCESS CONTROL IN command also shall be implemented. The ACCESS CONTROL OUT command shall not be affected by access controls.

Table 420 — ACCESS CONTROL OUT service actions
{{no changes in table 420 contents}}

The ACCESS CONTROL OUT command may be addressed to any logical unit whose standard INQUIRY data (see 6.4.2) has the ACC bit set to one (e.g., LUN 0), in which case it shall be processed in the same manner as if the command had been addressed to the ACCESS CONTROLS well known logical unit. If an ACCESS CONTROL OUT command is received by a device server whose standard INQUIRY data has the ACC bit set to zero, 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 COMMAND OPERATION CODE.

If an ACCESS CONTROL OUT command is received while an IKEv2-SCSI CCS is in progress (see 5.13.4), the command shall be terminated with a CHECK CONDITION status, with the sense key NOT READY, and the additional sense code set to LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS. The sense key specific additional sense data may be set as described in 5.13.5.

{{LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS is a new additional sense code.}}

… {{The remainder of 8.3.3.1 is unchanged by this proposal.}} …

Annex C
(Informative)
IKEv2 protocol details and variations for IKEv2-SCSI

{{All of Annex C is new. Additions/deletions markups are not applied in this annex.}}

The IKEv2 protocol details and variations specified in RFC 4306 apply to IKEv2-SCSI (i.e., this standard) as follows:

a) Any SECURITY PROTOCOL OUT command with a transfer length of up to 16 384 bytes is not terminated with an error due to the number of bytes transferred;
b) The timeout and retransmission mechanisms defined in RFC 4306 are not used by this standard, instead a new payload is defined to transfer appropriate timeout values to the device server;
c) Each SCSI command used by this standard completes by conveying a status from the device server to the application client;
d) The IKEv2 header EXCHANGE TYPE field is reserved in this standard as a result of equivalent information being transferred in the SECURITY PROTOCOL OUT command and SECURITY PROTOCOL IN command CDBs;
e) The IKEv2 header VERSION bit is reserved in this standard;
f) This standard uses the pseudo-random functions (PRF) functions defined by RFC 4306;
g) The key derivation functions defined and used by this standard (see 5.13.3) are equivalent to the PRF+ found in RFC 4306;
h) The SA creation transactions (see 3.1.o) defined by this standard are not overlapped. If an application client attempts to start a second SA creation transaction before the first is completed, the offending command is terminated as described in 5.13.4.1, but this does not affect the SA creation transaction that is already in progress;
i) The NO_PROPOSAL_CHOSEN and INVALID_KE_PAYLOAD notify error types are replaced by the SA CREATION PARAMETER VALUE INVALID additional sense code (see 7.7.3.9) because IKEv2-SCSI has a different negotiation structure. As defined in RFC 4306, an IKEv2 initiator offers one or more proposals to a responder without knowing what is acceptable to the responder, and chooses a DH group without knowing whether it is acceptable to the responder. These two notify error types allow the responder to inform the initiator that one or more of its choices are not acceptable. In contrast, an IKEv2-SCSI application client obtains the device server capabilities in the Device Capabilities step (see 5.13.4.2) and selects algorithms from them in the Key Exchange step (see 5.13.4.8). An error only occurs if the application client has made an invalid selection, hence the SA CREATION PARAMETER VALUE INVALID description;
j) IKEv2 version numbers (see RFC 4306) are used by this standard (see 7.7.3.4), but the ability to respond to an unsupported version number with the highest version number to be used is not supported, and this standard does not include checks for version downgrade attacks;
k) IKEv2 cookies (see RFC 4306) are not used by this standard;
l) IKEv2 cryptographic algorithm negotiation (see RFC 4306) is replaced by the Device Server Capabilities step (see 5.13.4.7) and the Key Exchange step (see 5.13.4.8) (i.e., the IKEv2 proposal construct is not used by this standard);
m) In this standard an SA is rekeyed by replacing it with a new SA:
    A) CHILD_SAs are not used by this standard;
    B) The RFC 4306 discussion of CHILD_SAs does not apply to this standard;
    C) Coexistence of the original SA and the new SA is achieved for rekeying purposes by restricting the device server's ability to delete SAs to the following cases:
        a) Expiration of a timeout (see 7.7.3.5.14);
        b) Processing of an IKEv2-SCSI Delete function (see 5.13.4.13); and
        c) Responding to an initial contact notification (see 7.7.3.5.8);
        and

D) IKEv2 does not support rekeying notification for IKE_SAs, therefore this standard does not support rekeying notification;

n) The AUTH METHOD field in the Authentication payload is reserved in this standard (see 7.7.3.5.6). The choice of authentication methods for both transfer directions is negotiated using the SA_AUTH_OUT IKEv2-SCSI cryptographic algorithm descriptor and SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor (see 7.7.3.6.6) during the Key Exchange step (see 5.13.4.8).

o) The usage of Certificate Encodings in the Certificate payload and Certificate Request payload (see 7.7.3.5.5) are constrained as follows:
   A) In accordance with the recommendations in RFC 4718, Certificate Encoding values 01h-03h and 05h-0Ah are prohibited;
   B) This standard forbids the use of URL-based Certificate Encodings (i.e., Certificate Encodings values 0Ch and 0Dh); and
   C) Certificate Encoding values that RFC 4306 defines as vendor specific are reserved in this standard;

p) Deleting an SA requires knowing the SAIs (i.e., SPIs) in both directions and including both SAIs in the Delete payload. The RFC 4306 description of the Delete payload is vague enough to allow this. The requirement is consistent with the SCSI model for SAs;

q) Traffic Selectors (see RFC 4306) are not used by this standard;

r) The requirements in RFC 4306 on nonces are be followed for the random nonces (see 3.1.95) defined by this standard;

s) The RFC 4306 requirements on address and port agility are specific to the user datagram protocol and the IP protocol and do not apply to this standard;

t) Keys for the Authentication step are generated as specified in RFC 4306;

u) This standard uses a slightly modified version of the authentication calculations in RFC 4306 (see 7.7.3.5.6);

v) The RFC 4306 sections that describe the following features are not used by this standard:
   A) Extensible authentication protocol methods;
   B) Generating keying Material for CHILD_SAs;
   C) Rekeying an IKE SA using CREATE_CHILD_SA;
   D) Requesting an internal address;
   E) Requesting the peer's version;
   F) IPComp;
   G) NAT traversal; and
   H) Explicit congestion notification;

w) IKEv2 Error Handling (see RFC 4306) is replaced by the use of CHECK CONDITION status and sense data by this standard. See 7.7.3.9 for details of how errors reported in the Notify payload are translated to sense data;

x) The description of how AUTH_COMBINED is used in the Encrypted payload in this standard predates the definition of equivalent functionality in IETF standards; and

y) Because of the command-response nature of SCSI, protecting against denial of service attacks against the device server is very difficult, and no such protection is defined by this standard. Protection against denial of service attacks against the application client is described in 7.7.3.8.

Where this standard uses IKE payload names (see 7.7.3.4) RFC 4306 uses the shorthand notation shown in table C.1.

Table C.1 — IKE payload names shorthand

IKE payload name in this standard [a]	RFC 4306 shorthand [b]
Security Association	SAi or SAr
Key Exchange	KEi or KEr
Identification – Application Client	IDi
Identification – Device Server	IDr
Certificate	CERTi or CERTr
Certificate Request	CERTREQi
Authentication	AUTHi or AUTHr
Nonce	NONCEi or NONCEr
Notify	N-ICi or N-ICr
Delete	Di
Vendor ID	Vi or Vr
Traffic Selector – Application Client	TSi
Traffic Selector – Device Server	TSr
Encrypted	Ei or Er
Configuration	CPi or CPr
Extensible Authentication	EAPi or EAPr

[a] To facilitate future enhancements, all IKE payloads are listed in this table, but not all entries in this table are used in this standard.

[b] In RFC 4306 the lowercase i indicates initiator and r indicates responder. In this standard, the initiator is the application client and all such IKE payloads (e.g., KEi) appear in a SECURITY PROTOCOL OUT parameter list. The responder is always the device server in this standard and all such IKE payloads (e.g., AUTHr) appear in SECURITY PROTOCOL IN parameter data.

{{The proposed SPC-4 changes end here.}}

Summary of new additional sense codes

recommend 00h/1Eh for — CONFLICTING SA CREATION REQUEST
recommend 04h/13h for — LOGICAL UNIT NOT READY, SA CREATION IN PROGRESS
recommend 2Ah/14h for — SA CREATION CAPABILITIES DATA HAS CHANGED
recommend 74h/0Ch for — UNABLE TO DECRYPT PARAMETER LIST
recommend 74h/10h for — SA CREATION PARAMETER VALUE INVALID
recommend 74h/11h for — SA CREATION PARAMETER VALUE REJECTED (see 5.13.4.12)
recommend 74h/30h for — SA CREATION PARAMETER NOT SUPPORTED {{referenced only in table x46}}
recommend 74h/40h for — AUTHENTICATION FAILED