



ISO/IEC 14776-481

Edition 1.0 2019-12

INTERNATIONAL STANDARD



**Information technology – Small Computer System Interface (SCSI) –
Part 481: Security features for SCSI commands (SFSC)**



Documents

Document Sharing Hub



THIS PUBLICATION IS COPYRIGHT PROTECTED
Copyright © 2019 ISO/IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about ISO/IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

IEC Central Office
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigendum or an amendment might have been published.

IEC publications search - webstore.iec.ch/advsearchform

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,...). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

Electropedia - www.electropedia.org

The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary

67 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.



ISO/IEC 14776-481

Edition 1.0 2019-12

INTERNATIONAL STANDARD



**Information technology – Small Computer System Interface (SCSI) –
Part 481: Security features for SCSI commands (SFSC)**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 35.200

ISBN 978-2-8322-7663-1

Warning! Make sure that you obtained this publication from an authorized distributor.



ISO/IEC 14776-481

Information technology - Small Computer System Interface (SCSI) - Part 481: Security features for SCSI commands (SFSC)

Reference

ISO/IEC 14776-481

Contents

Page

FOREWORD.....	9
INTRODUCTION.....	11
1 Scope.....	12
2 Normative references.....	12
3 Terms and definitions, symbols, abbreviations, and conventions	14
3.1 Terms and definitions	14
3.2 Abbreviations and symbols.....	24
3.2.1 Abbreviations.....	24
3.2.2 Symbols.....	25
3.2.3 Mathematical operators	25
3.3 Keywords.....	25
3.4 Editorial conventions	27
3.5 Numeric and character conventions	27
3.5.1 Numeric conventions	27
3.5.2 Units of measure	28
3.5.3 Byte encoded character strings conventions.....	29
3.6 Bit and byte ordering	29
4 Security features model common to all device types.....	31
4.1 Security features for SCSI devices.....	31
4.1.1 Security associations.....	31
4.1.1.1 Principles of SAs	31
4.1.1.2 SA parameters.....	32
4.1.1.3 Creating an SA	34
4.1.2 Key derivation functions	35
4.1.2.1 KDFs overview	35
4.1.2.2 IKEv2-based iterative KDF	36
4.1.2.3 HMAC-based KDFs.....	36
4.1.2.4 AES-XCBC-PRF-128 IKEv2-based iterative KDF	38
4.1.3 Using IKEv2-SCSI to create an SA	38
4.1.3.1 Overview.....	38
4.1.3.2 IKEv2-SCSI Protocol summary	42
4.1.3.3 IKEv2-SCSI Authentication.....	44
4.1.3.3.1 Overview.....	44
4.1.3.3.2 Pre-shared key authentication.....	45
4.1.3.3.3 Digital signature authentication	46
4.1.3.3.3.1 Overview.....	46
4.1.3.3.3.2 Certificates and digital signature authentication	46
4.1.3.3.3.3 Example of certificate use for digital signature authentication.....	47
4.1.3.3.3.4 Handling of the Certificate Request payload and the Certificate payload	47
4.1.3.4 Constraints on skipping the Authentication step.....	47
4.1.3.4 Summary of IKEv2-SCSI shared keys nomenclature and shared key sizes	49
4.1.3.5 Device Server Capabilities step.....	50
4.1.3.6 IKEv2-SCSI Key Exchange step	52
4.1.3.6.1 Overview.....	52
4.1.3.6.2 Key Exchange step SECURITY PROTOCOL OUT command	52
4.1.3.6.3 Key Exchange step SECURITY PROTOCOL IN command	53
4.1.3.6.4 Key Exchange step completion	54
4.1.3.6.5 After the Key Exchange step	54
4.1.3.7 IKEv2-SCSI Authentication step.....	54

4.1.3.7.1 Overview	54
4.1.3.7.2 Authentication step SECURITY PROTOCOL OUT command	55
4.1.3.7.3 Authentication step SECURITY PROTOCOL IN command	56
4.1.3.8 Generating shared keys	57
4.1.3.8.1 Overview	57
4.1.3.8.2 Generating shared keys when the Authentication step is skipped	58
4.1.3.8.3 Generating shared keys when the Authentication step is processed	58
4.1.3.8.4 Initializing shared key generation	58
4.1.3.8.4.1 Initializing for SA creation shared key generation	58
4.1.3.8.4.2 Initializing for generation of shared keys used by the created SA	59
4.1.3.8.5 Generating shared keys used for SA management	59
4.1.3.8.6 Generating shared keys for use by the created SA	60
4.1.3.9 IKEv2-SCSI SA generation	61
4.1.3.10 Abandoning an IKEv2-SCSI CCS	62
4.1.3.11 Deleting an IKEv2-SCSI SA	63
4.1.4 Security progress indication	63
4.1.5 ESP-SCSI encapsulations for parameter data	64
4.1.5.1 Overview	64
4.1.5.2 ESP-SCSI required inputs	64
4.1.5.3 ESP-SCSI data format before encryption and after decryption	65
4.1.5.4 ESP-SCSI outbound data descriptors	66
4.1.5.4.1 Overview	66
4.1.5.4.2 ESP-SCSI CDBs or Data-Out Buffer parameter lists including a descriptor length	67
4.1.5.4.2.1 Initialization vector absent	67
4.1.5.4.2.2 Initialization vector present	68
4.1.5.4.3 ESP-SCSI Data-Out Buffer parameter lists for externally specified descriptor length	70
4.1.5.4.3.1 Initialization vector absent	70
4.1.5.4.3.2 Initialization vector present	71
4.1.5.5 ESP-SCSI Data-In Buffer parameter data descriptors	71
4.1.5.5.1 Overview	71
4.1.5.5.2 ESP-SCSI Data-In Buffer parameter data including a descriptor length	72
4.1.5.5.2.1 Initialization vector absent	72
4.1.5.5.2.2 Initialization vector present	74
4.1.5.5.3 ESP-SCSI Data-In Buffer parameter data for externally specified descriptor length	75
4.1.5.5.3.1 Initialization vector absent	75
4.1.5.5.3.2 Initialization vector present	76
4.1.6 Security algorithm codes	77
4.2 Secure random numbers	79
5 Security protocol parameters for all device types	80
5.1 Security protocol information description	80
5.1.1 Overview	80
5.1.2 CDB description	80
5.1.3 Supported security protocols list description	81
5.1.4 Certificate data description	82
5.1.4.1 Certificate overview	82
5.1.4.2 Public Key certificate description	82
5.1.4.3 Attribute certificate description	82
5.1.5 Security compliance information description	83
5.1.5.1 Security compliance information overview	83
5.1.5.2 Compliance descriptor overview	84
5.1.5.3 FIPS 140 compliance descriptor	85
5.2 SA creation capabilities	86
5.2.1 Overview	86
5.2.2 SA creation capabilities CDB description	86
5.2.3 SA creation capabilities parameter data formats	87
5.2.3.1 Supported device server capabilities formats parameter data format	87

5.2.3.2 IKEv2-SCSI device server capabilities parameter data format.....	88
5.3 IKEv2-SCSI	88
5.3.1 Overview.....	88
5.3.2 IKEv2-SCSI SECURITY PROTOCOL IN CDB description	89
5.3.3 IKEv2-SCSI SECURITY PROTOCOL OUT CDB description	90
5.3.4 IKEv2-SCSI parameter data format.....	91
5.3.5 IKEv2-SCSI payloads.....	98
5.3.5.1 IKEv2-SCSI payload format.....	98
5.3.5.2 No Next payload	99
5.3.5.3 Key Exchange payload.....	100
5.3.5.4 Identification – Application Client payload and Identification – Device Server payload.....	101
5.3.5.5 Certificate payload.....	102
5.3.5.6 Certificate Request payload	103
5.3.5.7 Authentication payload	104
5.3.5.8 Nonce payload.....	106
5.3.5.9 Notify payload	107
5.3.5.10 Delete payload.....	108
5.3.5.11 Encrypted payload.....	109
5.3.5.11.1 Combined mode encryption.....	109
5.3.5.11.2 Encrypted payload introduction	110
5.3.5.11.3 IKEv2-SCSI AAD	112
5.3.5.11.4 Processing a received Encrypted payload	113
5.3.5.12 IKEv2-SCSI SA Creation Capabilities payload.....	115
5.3.5.13 IKEv2-SCSI SA Cryptographic Algorithms payload.....	116
5.3.5.14 IKEv2-SCSI SAUT Cryptographic Algorithms payload	118
5.3.5.15 IKEv2-SCSI Timeout Values payload.....	119
5.3.6 IKEv2-SCSI cryptographic algorithm descriptors	120
5.3.6.1 Overview.....	120
5.3.6.2 ENCR IKEv2-SCSI cryptographic algorithm descriptor	122
5.3.6.3 PRF IKEv2-SCSI cryptographic algorithm descriptor	124
5.3.6.4 INTEG IKEv2-SCSI cryptographic algorithm descriptor	126
5.3.6.5 D-H IKEv2-SCSI cryptographic algorithm descriptor.....	127
5.3.6.6 IKEv2-SCSI authentication algorithm IKEv2-SCSI cryptographic algorithm descriptor.....	129
5.3.7 Errors in IKEv2-SCSI security protocol commands	131
5.3.8 Errors in IKEv2-SCSI security protocol parameter data	133
5.3.8.1 Overview.....	133
5.3.8.2 Errors with high denial of service attack potential	133
5.3.8.3 Errors with low denial of service attack potential.....	134
5.3.9 Translating IKEv2 errors.....	134
Annex A (informative) Security goals and threat model	136
A.1 Overview	136
A.2 Security goals.....	136
A.3 Threat model.....	137
A.4 Types of attacks	137
A.5 SCSI security considerations	138
Annex B (informative) Variations between this document and equivalent security protocols	139
B.1 IKEv2 protocol details and variations for IKEv2-SCSI.....	139
B.2 ESP protocol details and variations for ESP-SCSI	142
BIBLIOGRAPHY	143

Figures

	Page
Figure 1 — SCSI document structure	11
Figure 2 — SA relationships	31
Figure 3 — IKEv2-SCSI Device Server Capabilities step	42
Figure 4 — IKEv2-SCSI Key Exchange step	42
Figure 5 — IKEv2-SCSI Authentication step.....	43
Figure 6 — IKEv2-SCSI Delete operation.....	44

Tables

	Page
Table 1 — Numbering conventions examples	28
Table 2 — Comparison of decimal prefixes and binary prefixes	29
Table 3 — Minimum SA parameters	32
Table 4 — USAGE_TYPE SA parameter	34
Table 5 — Security protocols that create SAs	35
Table 6 — KDFs summary	36
Table 7 — HMAC-based KDFs	37
Table 8 — Hash functions used by HMAC based on KDF_ID	37
Table 9 — RFC 3566 parameter translations for the KDF based on AES-XCBC-PRF-128	38
Table 10 — IKEv2-SCSI shared key names and SA shared key names	49
Table 11 — Shared key size determination	50
Table 12 — Device Server Capabilities step parameter data requirements	51
Table 13 — IKEv2-SCSI command terminations that do not abandon the CCS	62
Table 14 — ESP-SCSI data format before encryption and after decryption	65
Table 15 — ESP-SCSI outbound data descriptors	66
Table 16 — ESP-SCSI CDBs or Data-Out Buffer parameter list descriptor without initialization vector	67
Table 17 — ESP-SCSI CDBs or Data-Out Buffer full parameter list descriptor	69
Table 18 — ESP-SCSI Data-Out Buffer parameter list descriptor without length and initialization vector	70
Table 19 — ESP-SCSI Data-Out Buffer parameter list descriptor without length	71
Table 20 — ESP-SCSI Data-In Buffer parameter data descriptors	72
Table 21 — ESP-SCSI Data-In Buffer parameter data descriptor without initialization vector	72
Table 22 — ESP-SCSI Data-In Buffer full parameter data descriptor	74
Table 23 — ESP-SCSI Data-In Buffer parameter data descriptor without length and initialization vector	75
Table 24 — ESP-SCSI Data-In Buffer parameter data descriptor without length	76
Table 25 — Security algorithm codes	77
Table 26 — SECURITY PROTOCOL SPECIFIC field for SECURITY PROTOCOL IN protocol 00h	80
Table 27 — Supported security protocols SECURITY PROTOCOL IN parameter data	81
Table 28 — Certificate data SECURITY PROTOCOL IN parameter data	82
Table 29 — Security compliance information SECURITY PROTOCOL IN parameter data	83
Table 30 — Compliance descriptor format	84
Table 31 — COMPLIANCE DESCRIPTOR TYPE field	84
Table 32 — FIPS 140 compliance descriptor	85
Table 33 — RELATED STANDARD field	85
Table 34 — SECURITY PROTOCOL SPECIFIC field for the SA creation capabilities	87
Table 35 — Supported device server capabilities formats parameter data	87
Table 36 — IKEv2-SCSI device server capabilities parameter data	88
Table 37 — SECURITY PROTOCOL SPECIFIC field as defined by the IKEv2-SCSI SECURITY PROTOCOL IN command	89
Table 38 — SECURITY PROTOCOL SPECIFIC field as defined by the IKEv2-SCSI SECURITY PROTOCOL OUT command	90
Table 39 — IKEv2-SCSI SECURITY PROTOCOL OUT command and SECURITY PROTOCOL IN command parameter data	91
Table 40 — IKEv2-SCSI header checking of SAs	93
Table 41 — NEXT PAYLOAD field	94
Table 42 — MESSAGE ID field	95
Table 43 — Next payload values in SECURITY PROTOCOL OUT/IN parameter data	96
Table 44 — IKEv2-SCSI payload format	98
Table 45 — Key Exchange payload format	100
Table 46 — Identification payload format	101
Table 47 — ID TYPE field	101
Table 48 — Certificate payload format	102
Table 49 — CERTIFICATE ENCODING field	102
Table 50 — Certificate Request payload format	103
Table 51 — Authentication payload format	104
Table 52 — Nonce payload format	106

Table 53 — Notify payload format.....	107
Table 54 — Delete payload format	108
Table 55 — Encrypted payload format.....	110
Table 56 — Plaintext format for Encrypted payload CIPHERTEXT field.....	112
Table 57 — IKEv2-SCSI SA Creation Capabilities payload format.....	115
Table 58 — IKEv2-SCSI SA Cryptographic Algorithms payload format	116
Table 59 — IKEv2-SCSI SAUT Cryptographic Algorithms payload format.....	118
Table 60 — IKEv2-SCSI Timeout Values payload format.....	119
Table 61 — IKEv2-SCSI cryptographic algorithm descriptor format.....	120
Table 62 — ALGORITHM TYPE field	121
Table 63 — ENCR IKEv2-SCSI cryptographic algorithm descriptor format.....	122
Table 64 — ENCR ALGORITHM IDENTIFIER field.....	123
Table 65 — PRF IKEv2-SCSI cryptographic algorithm descriptor format.....	124
Table 66 — PRF ALGORITHM IDENTIFIER field.....	125
Table 67 — INTEG IKEv2-SCSI cryptographic algorithm descriptor format	126
Table 68 — INTEG ALGORITHM IDENTIFIER field	126
Table 69 — D-H IKEv2-SCSI cryptographic algorithm descriptor format.....	127
Table 70 — D-H ALGORITHM IDENTIFIER field	128
Table 71 — SA_AUTH_OUT and SA_AUTH_IN IKEv2-SCSI cryptographic algorithm descriptor format....	129
Table 72 — SA_AUTH_OUT and SA_AUTH_IN ALGORITHM IDENTIFIER field	130
Table 73 — IKEv2-SCSI command order processing requirements on a single I_T_L nexus.....	132
Table 74 — IKEv2-SCSI parameter error categories.....	133
Table 75 — IKEv2 Notify payload error translations for IKEv2-SCSI.....	135
Table B.1 — IKE payload names shorthand	141

INFORMATION TECHNOLOGY – SMALL COMPUTER SYSTEM INTERFACE (SCSI) –

Part 481: Security features for SCSI commands (SFSC)

FOREWORD

- 1) ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.
- 2) The formal decisions or agreements of IEC and ISO on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees and ISO member bodies.
- 3) IEC, ISO and ISO/IEC publications have the form of recommendations for international use and are accepted by IEC National Committees and ISO member bodies in that sense. While all reasonable efforts are made to ensure that the technical content of IEC, ISO and ISO/IEC publications is accurate, IEC or ISO cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees and ISO member bodies undertake to apply IEC, ISO and ISO/IEC publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any ISO, IEC or ISO/IEC publication and the corresponding national or regional publication should be clearly indicated in the latter.
- 5) ISO and IEC do not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. ISO or IEC are not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or ISO or its directors, employees, servants or agents including individual experts and members of their technical committees and IEC National Committees or ISO member bodies for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication of, use of, or reliance upon, this ISO/IEC publication or any other IEC, ISO or ISO/IEC publications.
- 8) Attention is drawn to the normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this ISO/IEC publication may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 14776-481 was prepared by subcommittee 25: Interconnection of information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

The list of all currently available parts of the ISO/IEC 14776 series, under the general title *Information technology – Small computer system interface (SCSI)*, can be found on the IEC and ISO web sites.

The text of this standard is based on the following documents:

CDV	Report on voting
JTC1-SC25/2845/CDV	JTC1-SC25/2871/RVC

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2, except as described in 3.4 and 3.5.

A bilingual version of this publication may be issued at a later date.

IMPORTANT - The 'colour inside' logo on the cover page of this publication indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

ISO/IEC 14776 (all parts) provides for many different commands that define device models and commands for different SCSI devices. This document defines security features for use by all SCSI devices. This document defines the security model that is basic to every device model and the parameter data that may apply to any device model.

Figure 1 shows the relationship of this document to the other documents and related projects in the SCSI family of standards.

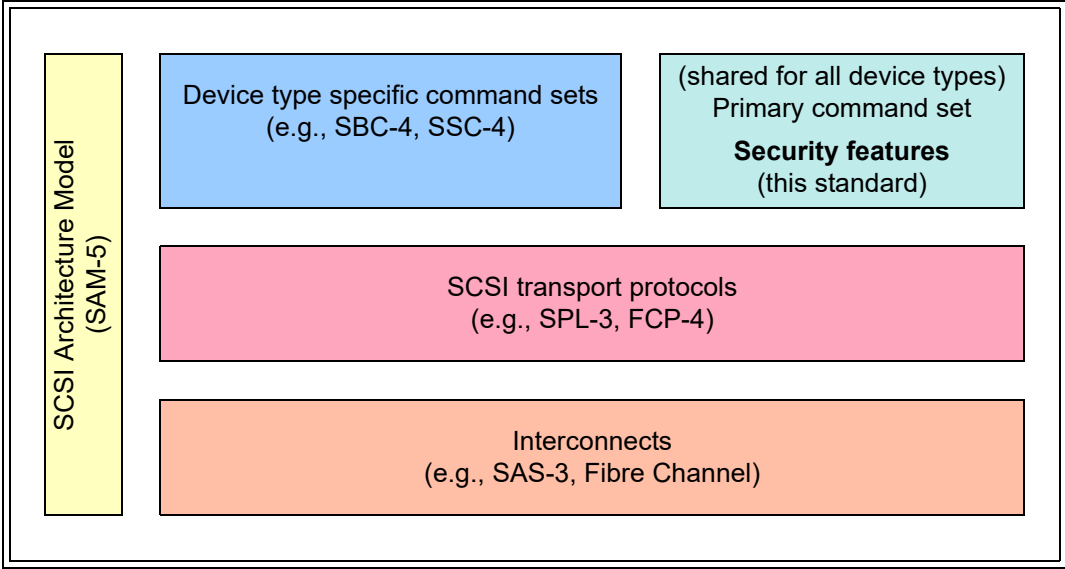


Figure 1 — SCSI document structure

The SCSI document structure in figure 1 shows the general applicability of the documents to one another. Figure 1 is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture.

The term SCSI is used to refer to ISO/IEC 14776 (all parts).

These documents specify the interfaces, functions and operations necessary to ensure interoperability between conforming implementations. This document contains a functional description. Conforming implementations employ any design technique that does not violate interoperability.

INFORMATION TECHNOLOGY – SMALL COMPUTER SYSTEM INTERFACE (SCSI) –

Part 481: Security features for SCSI commands (SFSC)

1 Scope

ISO/IEC 14776 (all parts) provides for many different types of SCSI devices (e.g., disks, tapes, media changers). This part of ISO/IEC 14776 defines a device model that is applicable to all SCSI devices. Other command standards expand on the general SCSI device model in ways appropriate to specific types of SCSI devices.

ISO/IEC 14776 (all parts) specifies the interfaces, functions, and operations necessary to ensure interoperability between conforming SCSI implementations. This document is a functional description. Conforming implementations employ any design technique that does not violate interoperability.

This document defines security features for use by all SCSI devices. This document defines the security model that is basic to every device model and the parameter data that applies to any device model. For additional information on the security goals and threat model discussed in this document see Annex A.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14776-415, *Information Technology – Small computer system interface (SCSI) - Part 415: SCSI Architecture Model - 5 (SAM-5)*

ISO/IEC 10646:2017, *Information technology - Universal Coded Character Set (UCS)*

INCITS 496-2012, *Information Technology - Fibre Channel - Security Protocols - 2 (FC-SP-2)*

INCITS 496-2012/AM1-2015, *Information Technology - Fibre Channel - Security Protocols - 2/Amendment 1 -(FC-SP-2/AM1)*

INCITS 502, *Information technology – SCSI Primary Commands - 5 (SPC-5)*

INCITS 516, *Information technology – SCSI Stream Commands - 4 (SSC-4)*

ANSI/IEEE 1619.1-2007, *Standard for Authenticated Encryption with Length Expansion for Storage Devices*

RFC 2410, *The NULL Encryption Algorithm and Its Use With IPsec*¹

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

RFC 3526, *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*¹

RFC 3566, *The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec*¹

RFC 3602, *The AES-CBC Cipher Algorithm and Its Use with IPsec*¹

¹ Copies of the IETF RFCs may be obtained at <http://www.ietf.org/>.

RFC 4106, *The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload* ¹

RFC 4309, *Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload* ¹

RFC 4434, *The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE)* ¹

RFC 5996, *Internet Key Exchange Protocol Version 2 (IKEv2)* ¹

RFC 6151, *Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms* ¹

RFC 7296, *Internet Key Exchange Protocol Version 2 (IKEv2)* ¹

NIST SP (Special Publication) 800-38D, *Recommendation for Block Cipher Modes of Operation: Galois/Counter (GCM) Mode for Confidentiality and Authentication and GMAC* ²

FIPS 180-4, *Secure Hash Standard* ²

FIPS 198-1, *The Keyed-Hash Message Authentication Code (HMAC)* ²

² Copies of the NIST and FIPS standards may be obtained at <http://csrc.nist.gov/publications/>.

3 Terms and definitions, symbols, abbreviations, and conventions

3.1 Terms and definitions

For the purposes of this document the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- IEC Electropedia: available at <http://www.electropedia.org/>
- ISO Online browsing platform: available at <http://www.iso.org/obp>

3.1.1

additional authenticated data

AAD

bytes of data input to a combined mode encryption algorithm (e.g., AES-GCM) as part of integrity checking computations

Note 1 to entry: The IKEv2-SCSI Encrypted payload (see 5.3.5.11) uses AAD.

3.1.2

additional sense code

combination of the ADDITIONAL SENSE CODE field and the ADDITIONAL SENSE CODE QUALIFIER field in sense data

Note 1 to entry: See SPC-5.

3.1.3

application client

object that is the source of SCSI commands

Note 1 to entry: See SAM-5.

3.1.4

byte

sequence of eight contiguous bits considered as a unit

3.1.5

command

request describing a unit of work to be performed by a device server

Note 1 to entry: See SAM-5.

3.1.6

command descriptor block

CDB

structure used to communicate commands from an application client to a device server (see SPC-5)

Note 1 to entry: A CDB may have a fixed length of up to 16 bytes or a variable length of between 12 bytes and 260 bytes.

3.1.7

command standard

SCSI standard that defines the model, commands, and parameter data for a device type

Note 1 to entry: SBC-3, SSC-4, SMC-3, MMC-6, and SES-2 are examples of command standards.

3.1.8

cryptographic command sequence

CCS

defined sequence of SECURITY PROTOCOL IN commands (see SPC-5) and SECURITY PROTOCOL OUT commands (see SPC-5) that realize the cryptographic protocol of a specified security operation

Note 1 to entry: 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 3.1.25) SA creation transaction (see 3.1.53) are a CCS.

3.1.9**Data-In Buffer**

buffer specified by the application client to receive data from the device server during the processing of a command

Note 1 to entry: See SAM-5.

3.1.10**Data-Out Buffer**

buffer specified by the application client to supply data that is sent from the application client to the device server during the processing of a command

Note 1 to entry: See SAM-5.

3.1.11**deferred error**

CHECK CONDITION status and sense data that is returned as the result of an error or exception condition that occurred during processing of a previous command for which GOOD status or CONDITION MET status has already been returned

Note 1 to entry: See SPC-5.

3.1.12**designation**

distinguishing name, identifier, or title

3.1.13**device server**

object within a logical unit that processes SCSI commands according to the rules of command management

Note 1 to entry: See SAM-5.

3.1.14**device type****peripheral device type**

device model implemented by the logical unit and indicated to the application client by the contents of the PERIPHERAL DEVICE TYPE field in the standard INQUIRY data (see SPC-5)

3.1.15**Encapsulating Security Payload for SCSI****ESP-SCSI**

method for transferring encrypted and/or integrity checked parameter data in Data-In Buffers and/or Data-Out Buffers based on Encapsulating Security Payload (see RFC 4303)

Note 1 to entry: See 4.1.5.

3.1.16**Extended CDB****XCDB**

CDB that contains another CDB and additional information to support extra processing

Note 1 to entry: Some security features use XCDBs.

Note 2 to entry: See SPC-5.

3.1.17**Extended Unique Identifier 48-bit format****EUI-48**

48-bit identifier that is globally unique

Note 1 to entry: The IEEE maintains a tutorial describing EUI-48 at

<https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/tutorials/eui.pdf>.

3.1.18**Extended Unique Identifier 64-bit format
EUI-64**

64-bit identifier that is globally unique

Note 1 to entry: The IEEE maintains a tutorial describing EUI-64 at <https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/tutorials/eui.pdf>.

3.1.19**field**

group of one or more contiguous bits, a part of a larger structure such as a CDB (see 3.1.6) or sense data (see 3.1.72)

3.1.20**hard reset**

condition resulting from the events defined by SAM-5 in which the SCSI device performs the hard reset operations described in SAM-5, this document, and the applicable command standards

3.1.21**hashed message authentication code
HMAC**

type of message authentication code that is calculated using the cryptographic hash function as defined in FIPS 198-1 (see clause 2) in combination with a secret key

3.1.22**I_T nexus**

nexus between a SCSI initiator port and a SCSI target port

Note 1 to entry: See SAM-5.

3.1.23**I_T nexus loss**

condition resulting from the events defined by SAM-5

Note 1 to entry: In response to this condition, the SCSI device performs the I_T nexus loss operations described in SAM-5, this document, and other applicable standards.

3.1.24**I_T_L nexus**

nexus between a SCSI initiator port, a SCSI target port, and a logical unit

Note 1 to entry: See SAM-5.

3.1.25**IKEv2-SCSI**

Internet Key Exchange protocol version 2 for SCSI

Note 1 to entry: See 4.1.3.

3.1.26**IKEv2-SCSI key**

shared key (see 3.1.75) used to provide security for IKEv2-SCSI operations

Note 1 to entry: IKEv2-SCSI keys used after SA creation is complete are maintained in the MGMT_DATA SA parameter (see 3.1.56).

Note 2 to entry: The shared key names used by IKEv2-SCSI are listed in 4.1.3.4.

Note 3 to entry: The creation and deletion of an SA are examples of IKEv2-SCSI operations that use IKEv2-SCSI keys.

3.1.27**IKEv2-SCSI CCS**

CCS (see 3.1.8) that is the Key Exchange step and Authentication step, if any, of an IKEv2-SCSI (see 3.1.25) SA creation transaction (see 3.1.53)

3.1.28**information unit****IU**

delimited and sequenced set of information in a format appropriate for transport by the service delivery subsystem

Note 1 to entry: A CDB for a specific SCSI command is an example of an IU.

3.1.29**initiator port name**

name (see 3.1.39) of a SCSI initiator port

3.1.30**integrity check value**

value used to cryptographically validate the integrity of a specified set of bytes that contain specified data based on a shared key (see 3.1.75)

3.1.31**internet assigned numbers authority****IANA**

service that assigns and manages registries of code values (i.e., code points) for the IETF and other users of the public internet

Note 1 to entry: See <http://www.ietf.org>.

Note 2 to entry: See <http://www.iana.org>.

Note 3 to entry: IANA maintains code values used by Internet Key Exchange version 2 (IKEv2) at <http://www.iana.org/assignments/ikev2-parameters>

3.1.32**key derivation function****KDF**

algorithm that is used to derive cryptographic keying material from a shared secret and other information

3.1.33**least significant bit****LSB**

<in a binary code> bit or bit position with the smallest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value

EXAMPLE – In the number 0001b, the bit that is set to one.

3.1.34**left-aligned**

type of field containing ASCII data in which unused bytes are placed at the end of the field (i.e., highest offset) and are filled with ASCII space (20h) characters

Note 1 to entry: See SPC-5.

3.1.35**logical unit**

externally addressable entity within a SCSI target device that implements a SCSI device model and contains a device server

Note 1 to entry: See SAM-5.

3.1.36**logical unit number****LUN**

encoded 64-bit identifier for a logical unit

Note 1 to entry: See SAM-5.

3.1.37**logical unit reset**

condition resulting from the events defined by SAM-5

Note 1 to entry: In response to this condition, the logical unit performs the logical unit reset operations described in SAM-5, this document, and other applicable standards.

3.1.38**most significant bit****MSB**

<in a binary code> bit or bit position with the largest numerical weighting in a group of bits that, when taken as a whole, represent a numerical value

EXAMPLE – In the number 1000b, the bit that is set to one.

3.1.39**name**

label of an object that is unique within a specified context and should never change

Note 1 to entry: In some situations, the term name and world wide identifier (WWID) are interchangeable.

3.1.40**nexus**

relationship between two SCSI devices, and the SCSI initiator port and SCSI target port objects within those SCSI devices

Note 1 to entry: See SAM-5.

3.1.41**nonce**

value that is used one and only one time to provide uniqueness to a value (e.g., a secure cryptographic key) in whose derivation the nonce participates or to uniquely identify a single instance of something (e.g., a timestamp) exchanged between an application client and a device server

3.1.42**null-padded**

type of field in which unused bytes are placed at the end of the field (i.e., highest offset) and are filled with ASCII null (00h) characters

Note 1 to entry: See SPC-5.

3.1.43**null-terminated**

type of field in which the last used byte (i.e., highest offset) is required to contain an ASCII null (00h) character

Note 1 to entry: See SPC-5.

3.1.44**organizationally unique identifier****OUI****company_id****IEEE company_id**

numeric identifier that is assigned by the IEEE such that no assigned identifiers are identical

Note 1 to entry: The IEEE prefers OUI for EUI-48 identifiers (see 3.1.17) and company_id for EUI-64 identifiers (see 3.1.18). However, the numeric identifier is called an OUI when that identifier is assigned by the IEEE.

Note 2 to entry: The IEEE maintains a tutorial describing the OUI at

<https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/tutorials/eui.pdf>.

3.1.45**page**

regular parameter structure or format used by several commands and identified with a value known as a page code

3.1.46**peripheral device**

object that connects to a logical unit

3.1.47**plaintext**

unencrypted bytes

3.1.48**power cycle**

power being removed from and later applied to a SCSI device

3.1.49**pre-shared key**

shared key (see 3.1.75) that is established by a method outside the scope of this document prior to the initiation of the function that requires its use

Note 1 to entry: The requirements on pre-shared keys that are particular to this document are described in 4.1.3.3.2.

Note 2 to entry: Some of the RFCs referenced by this document use the name pre-shared secret for the same information that this document calls a pre-shared key.

Note 3 to entry: See SPC-5.

3.1.50**random nonce**

secure random number (see 4.2) that has a negligible chance of repeating and whose uses include providing significant uniqueness and randomness in cryptographic calculations

3.1.51**request for comment****RFC**

standard developed by the Internet Engineering Task Force

Note 1 to entry: See <http://www.ietf.org/>.

3.1.52**right-aligned**

type of field containing ASCII data in which unused bytes are placed at the start of the field (i.e., lowest offset) and are filled with ASCII space (20h) characters

Note 1 to entry: See SPC-5.

3.1.53**SA creation transaction**

sequence of SECURITY PROTOCOL IN commands (see SPC-5) and SECURITY PROTOCOL OUT commands (see SPC-5) that is used to create an SA between an application client and device server

Note 1 to entry: A CCS (see 3.1.8) is a kind of SA creation transaction.

3.1.54**SA generation**

computation and initialization of the SA parameter values (see 3.1.56) required to create an SA (see 3.1.70)

Note 1 to entry: SA generation is performed separately by the application client and device server after all SCSI commands required to create an SA have been performed without error

Note 2 to entry: See 4.1.3.9.

3.1.55**SA keys**

shared keys (see 3.1.75) that are maintained in the USAGE_DATA SA parameter (see 3.1.56)

Note 1 to entry: SA keys are used to provide security for the operations that use the SA (e.g., encryption of command parameter data).

Note 2 to entry: The shared key names used by IKEv2-SCSI are listed in 4.1.3.4.

3.1.56**SA parameter**

parameter stored by both an application client and a device server

Note 1 to entry: SA parameters are associated with one SA (see 3.1.70) and identified by a pair of SAs (see 3.1.71).

Note 2 to entry: See 4.1.1.2.

3.1.57**SA participant**

application client or device server that participates in the creation or use of an SA (see 3.1.70)

3.1.58**salt bytes**

sequence of bytes whose contents are unpredictable in advance of their use

Note 1 to entry: Salt bytes are kept secret prior to their use and disclosed during their use (e.g., bytes added to a cryptographic operation to increase the entropy in the result (see RFC 4106)).

3.1.59**SCSI device**

device that contains one or more SCSI ports that are each connected to a service delivery subsystem and supports a SCSI application protocol

Note 1 to entry: See SAM-5.

3.1.60**SCSI device name**

name (see 3.1.39) of a SCSI device that is world wide unique within the protocol of a SCSI domain (see 3.1.61) in which the SCSI device has SCSI ports (see 3.1.64)

Note 1 to entry: The SCSI device name may be made available to other SCSI devices or SCSI ports in ways that are defined by a SCSI transport protocol standard (see 3.1.67).

Note 2 to entry: See SAM-5.

3.1.61**SCSI domain**

interconnection of two or more SCSI devices and a service delivery subsystem

3.1.62**SCSI initiator device**

SCSI device (see 3.1.59) containing application clients (see 3.1.3) and SCSI initiator ports (see 3.1.63)

Note 1 to entry: SCSI initiator devices originate device service and task management requests (see SAM-5) to be processed by a SCSI target device (see 3.1.65) and receive device service and task management responses from SCSI target devices.

Note 2 to entry: See SAM-5.

3.1.63**SCSI initiator port****initiator port**

SCSI initiator device (see SAM-5) object that acts as the connection between application clients and a service delivery subsystem through which requests and responses are routed

Note 1 to entry: See SAM-5.

3.1.64**SCSI port**

SCSI initiator port (see 3.1.63) or SCSI target port (see 3.1.66)

3.1.65**SCSI target device**

SCSI device (see 3.1.59) containing logical units (see 3.1.35) and SCSI target ports (see 3.1.66)

Note 1 to entry: SCSI target devices receive device service and task management requests (see SAM-5) for processing and send device service and task management responses to SCSI initiator devices.

Note 2 to entry: See SAM-5.

3.1.66**SCSI target port
target port**

SCSI target device (see SAM-5) object that acts as the connection between device servers and task managers and a service delivery subsystem through which requests and responses are routed

Note 1 to entry: See SAM-5.

3.1.67**SCSI transport protocol standard
protocol standard**

SCSI standard that defines a SCSI transport protocol

Note 1 to entry: FCP-4, SPL-3, SRP, and SBP-3 are examples of SCSI transport protocol standards.

3.1.68**secure hash algorithm****SHA**

secure hash algorithm (e.g., SHA-1) defined in FIPS 180-4, Secure Hash Standard (SHS)

Note 1 to entry: See clause 2.

3.1.69**secure random number**

random number that is generated in ways that protect the random number from security attacks

Note 1 to entry: See 4.2.

3.1.70**security association
SA**

relationship and associated security processing between an application client and a device server that is used to apply security functions (e.g., integrity checking, encryption) to data that is transferred in either direction

Note 1 to entry: See 4.1.1.

3.1.71**security association index
SAI**

number representing the parameters for an SA as stored internally by the application client or device server

Note 1 to entry: In other security models, this value is called the security parameters index (SPI).

Note 2 to entry: See 4.1.1.

3.1.72**sense data**

data describing an error, exceptional condition, or completion information

Note 1 to entry: See SPC-5.

3.1.73**sense key**

contents of the SENSE KEY field in sense data (see SPC-5)

3.1.74**service delivery subsystem**

part of a SCSI domain that transmits service requests to a logical unit or SCSI target device and returns logical unit or SCSI target device responses to a SCSI initiator device

Note 1 to entry: See SAM-5.

3.1.75**shared key****SK**

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)

Note 1 to entry: The shared key names used by IKEv2-SCSI are listed in 4.1.3.4.

3.1.76**SK_ai**

shared key (see 3.1.75) used for integrity checking data in a Data-Out Buffer

EXAMPLE – The integrity checking the Encrypted payload in an IKEv2-SCSI Authentication step SECURITY PROTOCOL OUT command (see 4.1.3.7.2) uses an SK_ai.

3.1.77**SK_ar**

shared key (see 3.1.75) used for integrity checking data in a Data-In Buffer

EXAMPLE – The integrity checking the Encrypted payload in an IKEv2-SCSI Authentication step SECURITY PROTOCOL IN command (see 4.1.3.7.3) uses an SK_ar.

3.1.78**SK_d**

shared key (see 3.1.75) KDF input material used to generate the shared keys stored in the KEYMAT SA parameter (see 3.1.56)

3.1.79**SK_ei**

shared key (see 3.1.75) used for encrypting data in a Data-Out Buffer

EXAMPLE – Encrypting the Encrypted payload in an IKEv2-SCSI Authentication step SECURITY PROTOCOL OUT command (see 4.1.3.7.2) uses an SK_ei.

3.1.80**SK_er**

shared key (see 3.1.75) used for encrypting data in a Data-In Buffer

EXAMPLE – Encrypting the Encrypted payload in an IKEv2-SCSI Authentication step SECURITY PROTOCOL IN command (see 4.1.3.7.3) uses an SK_er.

3.1.81**SK_pi**

shared key (see 3.1.75) used to construct the IKEv2-SCSI Data-Out Buffer Authentication payload (see 4.1.3.7.2)

3.1.82**SK_pr**

shared key (see 3.1.75) used to construct the IKEv2-SCSI Data-In Buffer Authentication payload (see 4.1.3.7.3)

3.1.83**status**

one byte of response information that contains a coded value defined in SAM-5, transferred from a device server to an application client upon completion of each command

Note 1 to entry: See SAM-5.

3.1.84**system**

one or more SCSI domains operating as a single configuration

3.1.85**target port name**

name (see 3.1.39) of a SCSI target port

3.1.86**task set**

group of commands within a logical unit, whose interaction is dependent on the task management (i.e., queuing) and ACA rules

Note 1 to entry: The Control mode page (see SPC-5) defines management capabilities for task sets.

Note 2 to entry: See SAM-5.

3.1.87**trust anchor**

authoritative entity for which trust is assumed and not derived

3.1.88**unit attention condition**

asynchronous status information that a logical unit establishes to report to the initiator ports associated with one or more I_T nexuses

Note 1 to entry: See SAM-5.

3.1.89**user data**

data contained in logical blocks (e.g., see SBC-3) that is not protection information (see SPC-5)

Note 1 to entry: Each command standard (see 3.1.7) may provide its own definition of user data.

3.1.90**UTF-8**

character set that is a transformation format of the character set defined by ISO/IEC 10646

Note 1 to entry: See RFC 3629.

3.1.91**word**

sequence of 16 contiguous bits considered as a unit

3.1.92**zero-padded**

type of field in which unused bytes are placed at the end of the field (i.e., highest offset) and are filled with zeros

Note 1 to entry: See SPC-5.

3.2 Abbreviations and symbols

3.2.1 Abbreviations

Abbreviation	Meaning
AAD	Additional Authenticated Data (see 3.1.1)
ACA	Auto Contingent Allegiance (see SAM-5)
AES	Advanced Encryption Standards
AES-GCM	Advanced Encryption Standard - Galois Counter Mode (see RFC 4106)
ASCII	American Standard Code for Information Interchange INCITS 4-1986 (R2002)(see INCITS 4-1986 (R2002))
ASN.1	Abstract Syntax Notation One (see clause 2)
CCS	Cryptographic Command Sequence (see 3.1.8)
CDB	Command Descriptor Block (see 3.1.6)
CSEC	Communications Security Establishment Canada
ECP	Elliptic Curve group modulo Prime (see RFC 5903)
ECDSA	Elliptic Curve Digital Signature Algorithm (see RFC 4754)
ESP-SCSI	Encapsulating Security Payload for SCSI (see 3.1.15)
EUI-48	Extended Unique Identifier 48-bit format (see 3.1.17)
EUI-64	Extended Unique Identifier 64-bit format (see 3.1.18)
FC-SP-2	Fibre Channel Security Protocols-2
FIPS	Federal Information Processing Standard
HMAC	Hashed Message Authentication Code (see 3.1.21)
HTTP	Hypertext Transfer Protocol (see RFC 2616)
IANA	Internet Assigned Numbers Authority (see 3.1.31)
ID	Identifier or Identification
IEEE	Institute of Electrical and Electronics Engineers
IKEv2	Internet Key Exchange version 2 (see clause 2)
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IU	Information Unit
KDF	Key Derivation Function (see 3.1.32)
LSB	Least Significant Bit (see 3.1.33)
LUN	Logical Unit Number (see 3.1.36)
MAC	Message Authentication Code
MODP	Modular exponential (see RFC 3526)
MSB	Most Significant Bit (see 3.1.38)
OUI	Organizationally Unique Identifier (see 3.1.44)
PKI	Public Key Infrastructure (see RFC 5280 and RFC 6818)
PRF	Pseudo-Random Function (see RFC 7296)
RFC	Request For Comments (see 3.1.51)
SA	Security Association (see 3.1.70)
SAI	Security Association Index (see 3.1.71)
SAM-5	SCSI Architecture Model - 5 (see clause 2)
SAUT	SA Usage Type (see 5.3.5.14)
SBC-4	SCSI Block Commands - 4
SCSI	Small Computer System Interface
SFSC	Security Features for SCSI Commands (this document)
SHA-1	Secure Hash Algorithm, 160 bits (see 3.1.68)
SHA-256	Secure Hash Algorithm, 256 bits (see 3.1.68)
SHA-384	Secure Hash Algorithm, 384 bits (see 3.1.68)
SHA-512	Secure Hash Algorithm, 512 bits (see 3.1.68)

Abbreviation	Meaning
SK	Shared Key (see 3.1.75)
SPC-5	SCSI Primary Commands - 5 (see clause 2)
SSC-4	SCSI Stream Commands - 4 (see clause 2)
URI	Uniform Resource Identifier (see RFC 3986)
URL	Uniform Resource Locator (see RFC 3986)
VPD	Vital Product Data (see SPC-5)
VS	Vendor Specific (see 3.3.12)
XCDB	Extended CDB (see 3.1.16)

3.2.2 Symbols

Symbol	Meaning
	concatenation
n/a	not applicable
x, xx	any valid value for a bit or field

3.2.3 Mathematical operators

Mathematical operator	Meaning
+	plus
−	minus
/	divided by
×	multiplied by
<	less than
≤	less than or equal to
>	greater than
≥	greater than or equal to
=	equal to
≠	not equal to

3.3 Keywords

3.3.1

invalid

keyword used to describe an illegal or unsupported bit, byte, word, field or code value

Note 1 to entry: Receipt by the device server of an invalid bit, byte, word, field or code value shall be reported as an error.

3.3.2

mandatory

keyword indicating an item that is required to be implemented as defined in this document

3.3.3

may

keyword indicating flexibility of choice with no implied preference

3.3.4

may not

keyword indicating flexibility of choice with no implied preference

3.3.5**obsolete**

keyword indicating that an item was defined in prior SCSI standards but has been removed from this document

3.3.6**option, optional**

keywords that describe features that are not required to be implemented by this document

Note 1 to entry: If any optional feature defined by this document is implemented, then it shall be implemented as defined in this document.

3.3.7**prohibited**

keyword used to describe a feature, function, or coded value that is defined in a non-SCSI standard (i.e., a standard that is not a member of the SCSI family of standards) to which this document makes a normative reference where the use of said feature, function, or coded value is not allowed for implementations of this document

3.3.8**reserved**

keyword referring to bits, bytes, words, fields and code values that are set aside for future standardization

Note 1 to entry: A reserved bit, byte, word or field shall be set to zero, or in accordance with a future extension to this document.

Note 2 to entry: Recipients are not required to check reserved bits, bytes, words or fields for zero values.

Note 3 to entry: Receipt of reserved code values in defined fields shall be reported as an error.

3.3.9**restricted**

keyword referring to bits, bytes, words, and fields that are set aside for other identified standardization purposes

Note 1 to entry: A restricted bit, byte, word, or field shall be treated as a reserved bit, byte, word or field in the context where the restricted designation appears.

3.3.10**shall**

keyword indicating a mandatory requirement

Note 1 to entry: Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to this document.

3.3.11**should**

keyword indicating flexibility of choice with a strongly preferred alternative

3.3.12**vendor specific****VS**

keyword indicating that specification of the referenced item (e.g., a bit, field, code value) is determined by the SCSI device vendor

Note 1 to entry: Specification of the referenced item is determined by the SCSI device vendor and may be used differently in various implementations.

3.4 Editorial conventions

Certain words and terms used in this document have a specific meaning beyond the normal English meaning. These words and terms are defined either in 3.1 or in the text where they first appear. Names of commands, statuses, sense keys, and additional sense codes are in all uppercase (e.g., REQUEST SENSE). Lowercase is used for words having the normal English meaning.

If there is more than one CDB length for a particular command (e.g., MODE SENSE(6) and MODE SENSE(10)) and the name of the command is used in a sentence without any CDB length descriptor (e.g., MODE SENSE), then the condition described in the sentence applies to all CDB lengths for that command.

The names of fields are in small uppercase (e.g., ALLOCATION LENGTH). When a field name is a concatenation of acronyms, uppercase letters may be used for readability (e.g., NORMACA). Normal case is used when the contents of a field are being discussed. Fields containing only one bit are usually referred to as the name bit instead of the name field.

When the value of the bit or field is not relevant, x or xx appears in place of a specific value.

Lists sequenced by lowercase or uppercase letters show no ordering relationship between the listed items.

EXAMPLE 1 – The following list shows no relationship between the colors named:

- a) red, specificity one of the following colors:
 - A) crimson; or
 - B) amber;
- b) blue; or
- c) green.

Lists sequenced by numbers show an ordering relationship between the listed items.

EXAMPLE 2 – The following list shows the order in which a page is meant to be read:

- 1) top;
- 2) middle; and
- 3) bottom.

Lists are associated with an introductory paragraph or phrase, and are numbered relative to that paragraph or phrase (i.e., all lists begin with an a) or 1) entry).

If a conflict arises between text, tables, or figures, the order of precedence to resolve the conflicts is text; then tables; and finally figures. Not all tables or figures are fully described in the text. Tables show data format and values. Notes do not constitute any requirements for implementors.

3.5 Numeric and character conventions

3.5.1 Numeric conventions

A binary number is represented in this document by any sequence of digits comprised of only the Arabic numerals 0 and 1 immediately followed by a lower-case b (e.g., 0101b). Underscores or spaces may be included in binary number representations to increase readability or delineate field boundaries (e.g., 0 0101 1010b or 0_0101_1010b).

A hexadecimal number is represented in this document by any sequence of digits comprised of only the Arabic numerals 0 to 9 and/or the upper-case English letters A to F immediately followed by a lower-case h

(e.g., FA23h). Underscores or spaces may be included in hexadecimal number representations to increase readability or delineate field boundaries (e.g., B FD8C FA23h or B_FD8C_FA23h).

A decimal number is represented in this document by any sequence of digits comprised of only the Arabic numerals 0 to 9 not immediately followed by a lower-case b or lower-case h (e.g., 25).

A range of numeric values is represented in this document in the form “a to z”, where a is the first value included in the range, all values between a and z are included in the range, and z is the last value included in the range (e.g., the representation “0h to 3h” includes the values 0h, 1h, 2h, and 3h).

This document uses the following conventions for representing decimal numbers:

- a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
- b) the thousands separator (i.e., separating groups of three digits in a portion of the number) is a space;
- c) the thousands separator is used in both the integer portion and the fraction portion of a number; and
- d) the decimal representation for a year is 1999 not 1 999.

Table 1 shows some examples of decimal numbers represented using various conventions.

Table 1 — Numbering conventions examples

ISO/IEC	United States	This document
0,6	0.6	0.6
3,141 592 65	3.14159265	3.141 592 65
1 000	1,000	1 000
1 323 462,95	1,323,462.95	1 323 462.95

A decimal number represented in this document with an overline over one or more digits following the decimal point is a number where the overlined digits are infinitely repeating (e.g., $666.\overline{6}$ means $666.666\ 666\ldots$ or $666\ 2/3$ and $12.\overline{142\ 857}$ means $12.142\ 857\ 142\ 857\ldots$ or $12\ 1/7$).

3.5.2 Units of measure

This document represents values using both decimal units of measure and binary units of measure. Values are represented by the following formats:

- a) for values based on decimal units of measure:
 - 1) numerical value (e.g., 100);
 - 2) space;
 - 3) prefix symbol and unit:
 - 1) decimal prefix symbol (e.g., M) (see table 2); and
 - 2) unit abbreviation (e.g., B);
 and
- b) for values based on binary units of measure:
 - 1) numerical value (e.g., 1 024);
 - 2) space;
 - 3) prefix symbol and unit:
 - 1) binary prefix symbol (e.g., Gi) (see table 2); and
 - 2) unit abbreviation (e.g., b).

Table 2 compares the prefix, symbols, and power of the binary and decimal units.

Table 2 — Comparison of decimal prefixes and binary prefixes

Decimal			Binary		
Prefix name	Prefix symbol	Power (base-10)	Prefix name	Prefix symbol	Power (base-2)
kilo	k	10^3	kibi	Ki	2^{10}
mega	M	10^6	mebi	Mi	2^{20}
giga	G	10^9	gibi	Gi	2^{30}
tera	T	10^{12}	tebi	Ti	2^{40}
peta	P	10^{15}	pebi	Pi	2^{50}
exa	E	10^{18}	exbi	Ei	2^{60}
zetta	Z	10^{21}	zebi	Zi	2^{70}
yotta	Y	10^{24}	yobi	Yi	2^{80}

3.5.3 Byte encoded character strings conventions

When this document requires one or more bytes to contain specific encoded characters, the specific characters are enclosed in single quotation marks. The single quotation marks identify the start and end of the characters that are required to be encoded but are not themselves to be encoded. The characters that are to be encoded are shown in the case that is to be encoded.

An ASCII space character (i.e., 20h) may be represented in a string by the character '␣' (e.g., 'SCSI␣device').

The encoded characters and the single quotation marks that enclose them are preceded by text that specifies the character encoding methodology and the number of characters required to be encoded.

EXAMPLE - Using the notation described in this subclause, stating that the eleven ASCII characters 'SCSI device' represent encoded characters is the same as writing out the following sequence of byte values: 53h 43h 53h 49h 20h 64h 65h 76h 69h 63h 65h.

3.6 Bit and byte ordering

This subclause describes the representation of fields in a table that defines the format of a SCSI structure (e.g., the format of a CDB).

If a field consists of more than one bit and contains a single value (e.g., a number), the LSB is shown on the right and the MSB is shown on the left (e.g., in a byte, bit 7 is the MSB and is shown on the left, and bit 0 is the LSB and is shown on the right). The MSB and LSB are not labeled if the field consists of 8 or fewer bits.

If a field consists of more than one byte and contains a single value, the byte containing the MSB is stored at the lowest address and the byte containing the LSB is stored at the highest address (i.e., big-endian byte ordering). The MSB and LSB are labeled.

If a field consists of more than one byte and always contains multiple fields each with their own values (e.g., a descriptor), then there is no MSB and LSB of the field itself and thus there are no MSB and LSB labels. Each individual field has an MSB and LSB that are labeled as appropriate in the table, if any, that describes the format of the sub-structure having multiple fields.

If a field that consists of more than one byte contains either multiple fields each with their own values or a single value, then:

- a) the MSB and LSB are labeled and they apply as described in this subclause whenever the field contains a single value; and
- b) the labels on constituent fields supersede the single value labels whenever the field contains multiple fields.

If a field contains a text string (e.g., ASCII or UTF-8), the MSB label is the MSB of the first character and the LSB label is the LSB of the last character.

When required for clarity, multiple byte fields may be represented with only two rows in a table. This condition is represented by values in the byte number column not increasing by one in each subsequent table row, thus indicating the presence of additional bytes.

4 Security features model common to all device types

4.1 Security features for SCSI devices

4.1.1 Security associations

4.1.1.1 Principles of SAs

Before an application client and device server begin applying security functions (e.g., data integrity checking, data encryption) to messages (i.e., data that is transferred in either direction between them), they perform a security protocol to create at least one SA (see 4.1.1.3). The result of the SA creation protocol is two sets of SA parameters (see 4.1.1.2), one that is maintained by the application client and one that is maintained by the device server.

In this model, SAs decouple the process of creating a security relationship from its usage in processing security functions. This decoupling allows either the creation or the usage of an SA to be upgraded in response to changing security threats without requiring both processes to be upgraded simultaneously.

Figure 2 shows the relationship between application clients and device servers with respect to SAs.

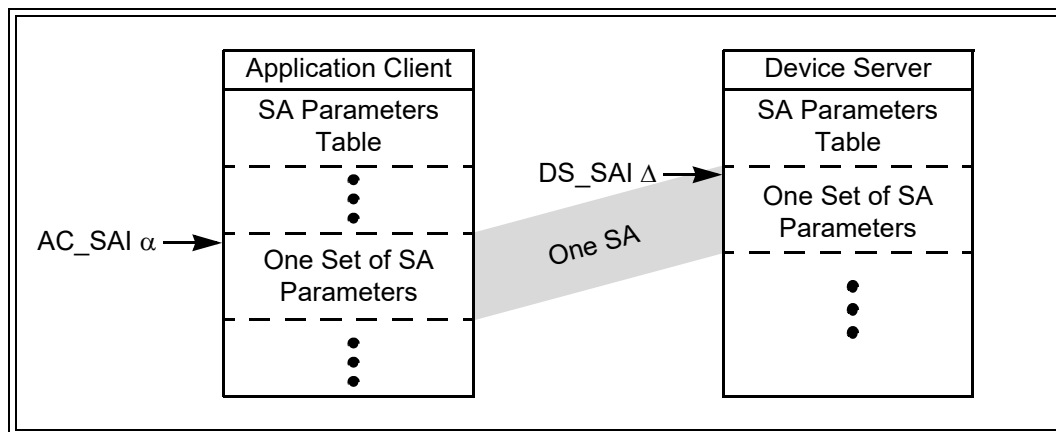


Figure 2 — SA relationships

In both the application client and the device server, the SA parameters are modeled as being stored in an indexed array and the SAI (i.e., the AC_SAI or the DS_SAI in figure 2) identifies one set of SA parameters within that array. The application client and device server are not required to store the parameters for any given SA in the same array locations. In order to support this implementation flexibility, a single SA is modeled as having two different SAI values (i.e., one for the application client and one for the device server).

The device server shall maintain a single SA parameters table for all I_T nexuses.

SAs shall not be preserved across a power cycle, hard reset, or logical unit reset. SAs shall not be affected by an I_T nexus loss.

4.1.1.2 SA parameters

Each SAI shall identify one set of SA parameters that includes at least the SA parameters defined in table 3. Individual security protocols define how the SA parameters are generated and/or used by that security protocol.

Table 3 — Minimum SA parameters (part 1 of 3)

Name	Description	Size (bytes) ^a		Scope ^b
		Min.	Max.	
SA parameters that identify and manage the SA.				
AC_SAI	The SAI used by the application client to identify the SA. ^c	4	4	Public
DS_SAI	The SAI used by the device server to identify the SA. ^c	4	4	Public
TIMEOUT	The number of seconds that may elapse after the completion of an SA access operation (i.e., SA creation or SA usage by a command) before the device server should discard the state associated with this SA (e.g., the SA parameters). If SA state is discarded because no SA access operations are received during the specified interval, then the device server shall respond to further attempts to access the SA as if the SA had never been created. This parameter shall not be set to zero.	4	4	Public
SA parameters that are incorporated in messages to prevent message replay attacks.				
AC_SQN	A sequence number that is incremented for each response message received by an application client on which a security function is performed and used to detect replay attacks (see A.4).	8	8	Public
DS_SQN	A sequence number that is incremented for each request message received by a device server on which a security function is performed and used to detect replay attacks.	8	8	Public
^a These size values are guidelines. Specific security protocols may place more exacting size requirements on SA parameters. ^b Public SA parameters may be transferred outside a SCSI device unencrypted. Secret SA parameters shall not be transferred outside a SCSI device. ^c SAI values between 0 and 255, inclusive, are reserved. ^d Nonce SA parameters shall be at least half the size of the KEY_SEED SA parameter. ^e The number of bits of entropy in the KEY_SEED SA parameter should be as close to the number of bits in the KEY_SEED SA parameter as possible (see RFC 3766). ^f Fields within a Field specific SA parameter are Public or Secret as defined by the applicable SA creation protocol.				

Table 3 — Minimum SA parameters (part 2 of 3)

Name	Description	Size (bytes) ^a		Scope ^b
		Min.	Max.	
SA parameters that are used by security functions to derive the secret keys that are applied to messages (e.g., for encryption).				
AC_NONCE ^d	A random nonce value that is generated by the application client and used as an input to the key derivation security algorithm specified by the KDF_ID SA parameter during the derivation of an encryption key.	16	64	Public
DS_NONCE ^d	A random nonce value that is generated by the device server and used as an input to the key derivation security algorithm specified by the KDF_ID SA parameter during the derivation of an encryption key.	16	64	Public
KEY_SEED ^e	A value that is known only to the application client and device server that are creating this SA that in combination with the applicable nonce is used to derive the KEYMAT SA parameter. The KEY_SEED SA parameter shall be set to zero as part of completing the SA creation function.	16	64	Secret
KDF_ID	A security algorithm (see 4.1.6) coded value that identifies the KDF used by the application client and device server.	4	4	Public
SA parameters that are used by security functions to secure messages between the application client and device server.				
KEYMAT	A value that is known only to the application client and device server that are participating in this SA that may be subdivided into one or more key values that are used in security functions that secure messages. The contents of KEYMAT depend on the USAGE_TYPE SA parameter value.	14	1 024	Secret
^a These size values are guidelines. Specific security protocols may place more exacting size requirements on SA parameters. ^b Public SA parameters may be transferred outside a SCSI device unencrypted. Secret SA parameters shall not be transferred outside a SCSI device. ^c SAI values between 0 and 255, inclusive, are reserved. ^d Nonce SA parameters shall be at least half the size of the KEY_SEED SA parameter. ^e The number of bits of entropy in the KEY_SEED SA parameter should be as close to the number of bits in the KEY_SEED SA parameter as possible (see RFC 3766). ^f Fields within a Field specific SA parameter are Public or Secret as defined by the applicable SA creation protocol.				

Table 3 — Minimum SA parameters (part 3 of 3)

Name	Description	Size (bytes) ^a		Scope ^b
		Min.	Max.	
SA parameters that are used by SA management functions.				
USAGE_TYPE	A coded value (see table 4) that indicates how the SA is used.	2	2	Public
USAGE_DATA	Information associated with how the SA is used (e.g., cryptographic algorithms and key sizes). The contents of USAGE_DATA depend on the USAGE_TYPE SA parameter value.	0	1 024	Public
MGMT_DATA	SA data that is used in ways defined by the SA creation protocol to perform SA management functions (e.g., deletion of the SA).	0	1 024	Field specific ^f
^a These size values are guidelines. Specific security protocols may place more exacting size requirements on SA parameters. ^b Public SA parameters may be transferred outside a SCSI device unencrypted. Secret SA parameters shall not be transferred outside a SCSI device. ^c SAI values between 0 and 255, inclusive, are reserved. ^d Nonce SA parameters shall be at least half the size of the KEY_SEED SA parameter. ^e The number of bits of entropy in the KEY_SEED SA parameter should be as close to the number of bits in the KEY_SEED SA parameter as possible (see RFC 3766). ^f Fields within a Field specific SA parameter are Public or Secret as defined by the applicable SA creation protocol.				

The USAGE_TYPE SA parameter (see table 4) provides an indication of how the SA is to be used.

Table 4 — USAGE_TYPE SA parameter

Code ^a	Description	USAGE_TYPE SA parameter		Reference
		Usage model	Description	
0000h to 0080h	Reserved			
0081h	Tape data encryption	ESP-SCSI ^b	None ^c	SSC-4
0082h to 8000h	Reserved			
8001h	Obsolete			
8002h to FFFFh	Reserved			
^a USAGE_TYPE values between 8000h and CFFFh inclusive place additional constraints on how an SA is to be created as described in 5.3.5.14. ^b ESP-SCSI usage is defined in 4.1.5. ^c The USAGE DATA LENGTH field in the IKEv2-SCSI SAUT Cryptographic Algorithms payload (see 5.3.5.14) shall contain zero.				

4.1.1.3 Creating an SA

The SECURITY PROTOCOL IN command (see SPC-5) and SECURITY PROTOCOL OUT command (see SPC-5) security protocols shown in table 5 are used to create SAs. The process of creating an SA establishes the SA parameter (see 4.1.1.2) values as follows:

- a) initial values for:
 - A) AC_SQN set as described in 4.1.3.9; and
 - B) DS_SQN set as described in 4.1.3.9;
- b) unchanging values for the lifetime of the SA:
 - A) AC_SAI;
 - B) DS_SAI;
 - C) TIMEOUT;
 - D) KDF_ID;
 - E) KEYMAT;
 - F) USAGE_TYPE;
 - G) USAGE_DATA; and
 - H) MGMT_DATA;
 and
- c) values that are zero (see 4.1.3.9) upon completion of SA creation:
 - A) KEY_SEED;
 - B) AC_NONCE; and
 - C) DS_NONCE.

Table 5 — Security protocols that create SAs

Security Protocol Code	Description	References
40h	SA creation capabilities	5.2
41h	IKEv2-SCSI	4.1.3 and 5.3

4.1.2 Key derivation functions

4.1.2.1 KDFs overview

A KDF produces KEYMAT from in input KEY_SEED and STRING as follows:

$\text{KEYMAT} = \text{KDF}(\text{KEY_SEED}, \text{STRING})$

Where:

KEY_SEED is an SA parameter (see 4.1.1.2); and
 STRING is a specified sequence of bytes.

Table 6 summarizes the KDFs defined by this document.

Table 6 — KDFs summary

Security Algorithm Code (see table 25)	Description	Reference
8002 0002h	IKEv2-based iterative HMAC KDF based on SHA-1	4.1.2.3
8002 0005h	IKEv2-based iterative HMAC KDF based on SHA-256	4.1.2.3
8002 0006h	IKEv2-based iterative HMAC KDF based on SHA-384	4.1.2.3
8002 0007h	IKEv2-based iterative HMAC KDF based on SHA-512	4.1.2.3
8002 0004h	IKEv2-based iterative KDF based on AES-128 in XCBC mode	4.1.2.4

4.1.2.2 IKEv2-based iterative KDF

To produce a sufficient number of bits in KEYMAT, the IKEv2-based (see RFC 7296) iterative KDF uses the following equation:

$$\text{KEYMAT} = T_1 \parallel T_2 \parallel T_3 \parallel T_4 \parallel \dots \parallel T_N$$

Where:

$T_1 = \text{KDF}(\text{KEY_SEED}, \text{STRING} \parallel 01\text{h});$
 $T_2 = \text{KDF}(\text{KEY_SEED}, T_1 \parallel \text{STRING} \parallel 02\text{h});$
 $T_3 = \text{KDF}(\text{KEY_SEED}, T_2 \parallel \text{STRING} \parallel 03\text{h});$
 $T_4 = \text{KDF}(\text{KEY_SEED}, T_3 \parallel \text{STRING} \parallel 04\text{h});$
 $\dots =$
 $T_N = \text{KDF}(\text{KEY_SEED}, T_{N-1} \parallel \text{STRING} \parallel N\text{h});$
 KDF is the function defined in 4.1.2.1;
 KEY_SEED is an SA parameter (see 4.1.1.2); and
 STRING is a specified sequence of bytes.

Protocols that use the IKEv2-based iterative KDF to generate KEYMAT should ensure that the number of KEYMAT bits requested does not cause N to exceed 255. If N reaches 256, then:

- 1) the requested number of KEYMAT bits is not returned; and
- 2) the request to produce KEYMAT shall be terminated with an error.

4.1.2.3 HMAC-based KDFs

If the KDF_ID is one of those shown in table 7, the KDF is a combination of the:

- a) HMAC function defined in FIPS 198-1 (see clause 2);
- b) secure hash function shown in table 7 for the specified KDF_ID value; and
- c) IKEv2-based iterative KDF technique (see 4.1.2.2).

The technique requires the following inputs from or related to the SA parameters:

- a) AC_SAI;
- b) DS_SAI;
- c) AC_NONCE;
- d) DS_NONCE;
- e) KEY_SEED; and

- f) the number of KEYMAT bits that are to be produced.

The USAGE_TYPE SA parameter and USAGE_DATA SA parameter (see 4.1.1.2) specify the KEYMAT size as part of the security protocol that performs SA creation (see 4.1.1.3).

The IKEv2-based iterative KDF technique (see 4.1.2.2) is applied with the following inputs:

- a) IFUNC (see 4.1.2.2) is the HMAC function defined in FIPS 198-1 with the translation of inputs names shown in table 7;
- b) KEY_SEED is the KEY_SEED SA parameter; and
- c) STRING contains the concatenated contents of the following SA parameters:
 - 1) AC_NONCE;
 - 2) DS_NONCE;
 - 3) AC_SAI; and
 - 4) DS_SAI.

Table 7 — HMAC-based KDFs

FIPS 198-1 inputs selected by KDF_ID	KDF_ID (see table 6)			
	8002 0002h	8002 0005h	8002 0006h	8002 0007h
<i>H</i> (i.e., hash function)	SHA-1 (see table 8)	SHA-256 (see table 8)	SHA-384 (see table 8)	SHA-512 (see table 8)
<i>B</i> (i.e., hash input block size) ^a	64	64	128	128
<i>L</i> (i.e., hash output block size) ^{a, b}	20	32	48	64
<i>K</i> (i.e., key)	KEY_SEED SA parameter			
<i>text</i>	STRING as defined in this subclause and used in 4.1.2.2			

^a In accordance with FIPS 198-1, all sizes are shown in bytes.

^b The HMAC-based KDFs defined by this document do not truncate (i.e., the truncation described in FIPS 198-1 does not apply).

Details of the hash functions that act as inputs to the FIPS 198-1 HMAC function are shown in table 8.

Table 8 — Hash functions used by HMAC based on KDF_ID

KDF_ID (see table 6)	Function	Description
8002 0002h	SHA-1	HMAC input <i>H</i> is the SHA-1 secure hash function defined in FIPS 180-4 (see clause 2).
8002 0005h	SHA-256	HMAC input <i>H</i> is the SHA-256 secure hash function defined in FIPS 180-4.
8002 0006h	SHA-384	HMAC input <i>H</i> is the SHA-384 secure hash function defined in FIPS 180-4.
8002 0007h	SHA-512	HMAC input <i>H</i> is the SHA-512 secure hash function defined in FIPS 180-4.

4.1.2.4 AES-XCBC-PRF-128 IKEv2-based iterative KDF

the KDF_ID is 8002 0004h, the KDF is a combination of:

- a) the AES-XCBC-PRF-128 secure hash function defined in RFC 4434 (see clause 2) and RFC 3566; and
- b) the IKEv2-based iterative KDF technique (see 4.1.2.2).

The technique requires the following inputs from or related to the SA parameters:

- a) AC_SAI;
- b) DS_SAI;
- c) AC_NONCE;
- d) DS_NONCE;
- e) KEY_SEED; and
- f) the number of KEYMAT bits that are to be produced.

The IKEv2-based iterative KDF (see 4.1.2.2) is applied with the following inputs:

- a) IFUNC (see 4.1.2.2) is the AES-XCBC-PRF-128 secure hash function with the translation of inputs names shown in table 9;
- b) KEY_SEED is the KEY_SEED SA parameter; and
- c) STRING contains the concatenated contents of the following SA parameters:
 - 1) AC_NONCE;
 - 2) DS_NONCE;
 - 3) AC_SAI; and
 - 4) DS_SAI.

Table 9 — RFC 3566 parameter translations for the KDF based on AES-XCBC-PRF-128

RFC 3566 Parameter	Translation
K (i.e., key)	KEY_SEED SA parameter
M (i.e., message)	STRING as defined in this subclause and used in 4.1.2.2

4.1.3 Using IKEv2-SCSI to create an SA

4.1.3.1 Overview

The IKEv2-SCSI protocol is a subset of the IKEv2 protocol (see RFC 7296) that this document defines for use in the creation and maintenance of an SA.

An IKEv2-SCSI SA creation transaction shall only be initiated by the application client.

The IKEv2-SCSI protocol creates the following pair of IKE SAs (see RFC 7296):

- a) an SA that protects data transferred from the application client to the device server; and
- b) an SA that protects data transferred 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 4.1.3.5): the application client determines the device server's cryptographic capabilities;
- 2) **Key Exchange step** (see 4.1.3.6): the application client and device server:
 - A) perform a key exchange;

- B) determine SAs;
 - C) generate the shared keys used for SA management (e.g., SA creation and deletion) (see 4.1.3.8); and
 - D) may complete the generation of the SA (see 4.1.3.9); and
- 3) **Authentication step** (see 4.1.3.7): unless omitted by application client and device server negotiations in the previous steps, the application client and device server:
- A) authenticate:
 - a) each other;
 - b) the key exchange; and
 - c) the capability selection; and
 - B) complete the generation of the SA (see 4.1.3.9).

The values in the SECURITY PROTOCOL field and the SECURITY PROTOCOL SPECIFIC field in the SECURITY PROTOCOL IN command (see SPC-5) and SECURITY PROTOCOL OUT command (see SPC-5) identify the step of the IKEv2-SCSI protocol (see 5.3.2).

The Key Exchange step and the Authentication step depend on the results from the Device Server Capabilities step in order to create an SA. During the 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 Authentication step;
- b) shared keys that are used by the Authentication step and to delete the SA; and
- c) shared keys that are used by SCSI usage type specific operations that obtain security from the generated SA.

More details about these shared keys are provided in 4.1.3.4.

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 since the Device Server Capabilities step, 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 Server 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:

- a) if the Authentication step is skipped (see 4.1.3.3.4):
 - 1) a Key Exchange step SECURITY PROTOCOL OUT command (see 4.1.3.6.2); and
 - 2) a Key Exchange step SECURITY PROTOCOL IN command (see 4.1.3.6.3);or
- b) if the Authentication step is performed:
 - 1) a Key Exchange step SECURITY PROTOCOL OUT command (see 4.1.3.6.2);
 - 2) a Key Exchange step SECURITY PROTOCOL IN command (see 4.1.3.6.3);
 - 3) an Authentication step SECURITY PROTOCOL OUT command (see 4.1.3.7.2); and
 - 4) an Authentication step SECURITY PROTOCOL IN command (see 4.1.3.7.3).

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 4.1.4) to ascertain the device server's progress for the command.

If an error is encountered, the device server or application client may abandon the IKEv2-SCSI CCS before the SA is created (see 4.1.3.10).

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 4.1.3.10;
- c) the SA being created by the IKEv2-SCSI CCS is deleted as described in 4.1.3.11;
- d) the number of seconds specified in the IKEV2-SCSI PROTOCOL TIMEOUT field of the IKEv2-SCSI Timeout Values payload (see 5.3.5.15) 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-5) 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 command or SECURITY PROTOCOL IN command with the SECURITY PROTOCOL field set to IKEv2-SCSI (i.e., 41h) 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 device server may terminate the command with CHECK CONDITION status, with the sense key set to ABORTED COMMAND, and the additional sense code set to CONFLICTING SA CREATION REQUEST.

Except for the PERSISTENT RESERVE OUT command (see SPC-5) 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 5.3.5.12) in the parameter data returned by the Device Server Capabilities step SECURITY PROTOCOL IN command (see 4.1.3.5):
 - A) an SA_AUTH_OUT algorithm descriptor (see 5.3.6.6) with the ALGORITHM IDENTIFIER field set to SA_AUTH_NONE; and
 - B) an SA_AUTH_IN algorithm descriptor (see 5.3.6.6) 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 5.3.5.13) in the Key Exchange step SECURITY PROTOCOL OUT command (see 4.1.3.6.2):
 - 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 SAUT Cryptographic Algorithms payload (see 5.3.5.14) 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 SAUT 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.

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 SECURITY PROTOCOL IN command, or SA_AUTH_NONE not being used by either command. Processing requirements placed on the SECURITY PROTOCOL OUT command during the Key Exchange step (see 5.3.6.6) ensure that this dependency is maintained.

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 SAUT 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 SAUT 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 SAUT 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 SAUT 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 A.4) are not of concern or are prevented by a means outside the scope of this document (e.g., physical security of the transport).

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.

EXAMPLE – Environments where it is appropriate to skip the Authentication step include those that protect data transfers using:

- a direct physical connection between the systems that contain the application client and the device server; or
- end-to-end security in the SCSI transport protocol (e.g., FC-SP-2).

4.1.3.2 IKEv2-SCSI Protocol summary

This subclause summarizes the IKE-v2-SCSI payloads (see 5.3.5) that are exchanged between an application client and a device server during all steps of an IKEv2-SCSI SA creation transaction using message diagrams. Each IKEv2-SCSI step (see 4.1.3.1) 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 3 shows the Device Server Capabilities step (see 4.1.3.5). The Device Server Capabilities step consists of a SECURITY PROTOCOL IN command carrying an IKEv2-SCSI SA Creation Capabilities payload (see 5.3.5.12). The IKEv2-SCSI header is not used.

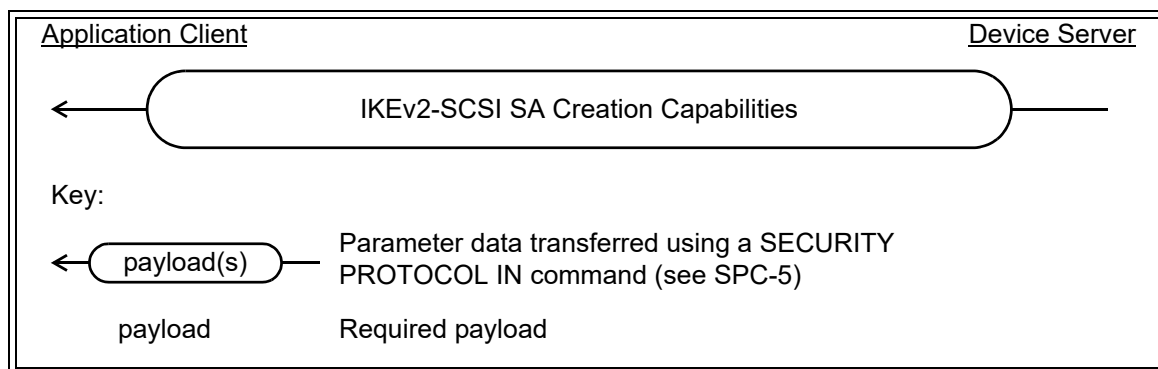


Figure 3 — IKEv2-SCSI Device Server Capabilities step

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

Figure 4 shows the Key Exchange step (see 4.1.3.6). The Key Exchange step consists of a SECURITY PROTOCOL OUT command followed by a SECURITY PROTOCOL IN command.

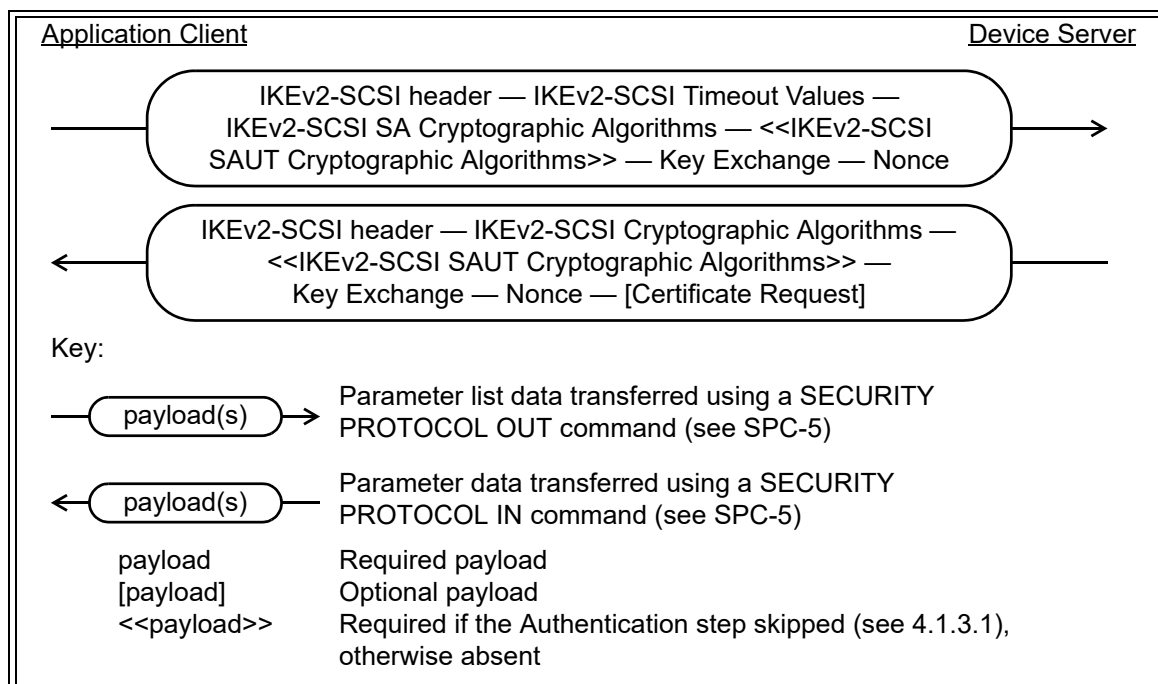


Figure 4 — IKEv2-SCSI Key Exchange step

The IKEv2-SCSI Timeout Values payload (see 5.3.5.15) contains timeouts for SA creation and usage.

The IKEv2-SCSI SA Cryptographic Algorithms payloads (see 5.3.5.13) are used to select and agree on the cryptographic algorithms used for creating the SA.

If the Authentication step is skipped (see 4.1.3.1), the IKEv2-SCSI SAUT Cryptographic Algorithms payloads (see 5.3.5.14) 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 during the Authentication step.

The Key Exchange payload (see 5.3.5.3) and Nonce payload (see 5.3.5.8) are part of the key and nonce exchanges that are used to generate the IKEv2-SCSI keys and SA keys.

The Certificate Request payload or payloads (see 5.3.5.6) enables the device server to request a certificate from the application client. If the Authentication step is being skipped (see 4.1.3.1), the device server shall not include any Certificate Request payloads in the parameter data. Use of the Certificate Request payload is described in 4.1.3.3.4.

Figure 5 shows the Authentication step (see 4.1.3.7). The Authentication step consists of a SECURITY PROTOCOL OUT command followed by a SECURITY PROTOCOL IN command.

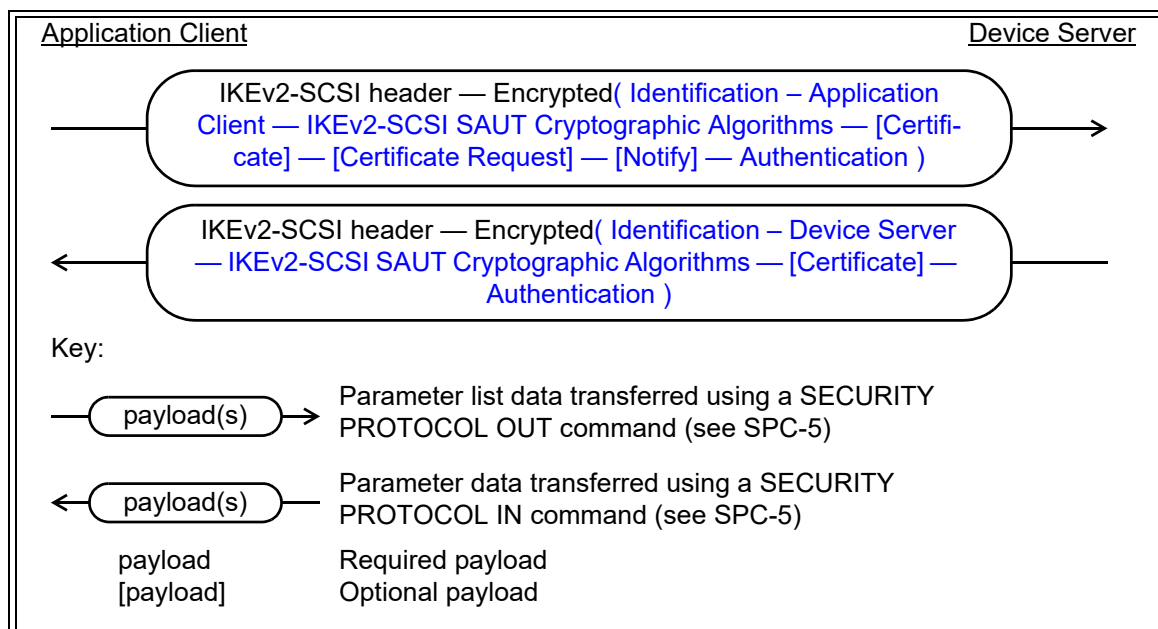


Figure 5 — IKEv2-SCSI Authentication step

An Encrypted payload (see 5.3.5.11) contains all other Authentication step payloads that are protected using the cryptographic algorithms determined by the IKEv2-SCSI SA Cryptographic Algorithms payloads (see 5.3.5.13) in the Key Exchange step (see figure 4).

The Identification payloads (see 5.3.5.4) contain the identities to be authenticated. These identities are not required to be SCSI names or identifiers.

The IKEv2-SCSI SAUT Cryptographic Algorithms payloads (see 5.3.5.14) are used to select and agree on usage of the SA and the cryptographic algorithms used by the created SA.

The Certificate payload or payloads (see 5.3.5.5) respond to the Certificate Request payload(s) sent by the device server in the Key Exchange step SECURITY PROTOCOL IN command.