INTERNATIONAL STANDARD

ISO/IEC 29167-10

Second edition
2017-09

Information technology — Automatic identification and data capture techniques —

Part 10:
Crypto suite AES-128 security services for air interface communications

Technologies de l'information — Techniques automatiques d'identification et de capture de données —

Partie 10: Services de sécurité par suite cryptographique AES-128 pour communications par interface radio

Reference number
ISO/IEC 29167-10:2017(E)

© ISO/IEC 2017

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

© ISO/IEC 2017 – All rights reserved

 © ISO/IEC 2017 – All rights reserved

Foreword

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 31, *Automatic identification and data capture techniques*.

This second edition cancels and replaces the first edition (ISO/IEC 29167-10:2015), which has been technically revised.

A list of all parts in the ISO/IEC 29167 series can be found on the ISO website.

© ISO/IEC 2017 – All rights reserved

Introduction

This document specifies the security services of an AES-128 crypto suite. AES has a fixed block size of 128 bits and a key size of 128 bits, 192 bits or 256 bits. This document uses AES with a fixed key size of 128 bits and is referred to as AES-128.

This document specifies procedures for the authentication of a Tag and or an Interrogator using AES-128 and provides the following features:

— Tag Authentication;

— Tag Authentication allows authenticated and encrypted reading of a part of the Tag's memory;

— Interrogator Authentication;

— Interrogator Authentication allows authenticated and encrypted writing of a part of the Tag's memory;

— Mutual Authentication.

Crypto suite only supports encryption on the Tag and uses encryption for "encrypting" messages sent from the Tag to the Interrogator and "decrypting" messages received from the Interrogator.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document might involve the use of patents concerning radio-frequency identification technology given in the clauses identified below.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licenses under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information on the declared patents may be obtained from:

Impinj, Inc.
Chris Dorio **Chief Strategy and Technology Officer**

The latest information on IP that might be applicable to this document can be found at www.iso.org/patents.

© ISO/IEC 2017 – All rights reserved

Information technology — Automatic identification and data capture techniques —

Part 10: Crypto suite AES-128 security services for air interface communications

1 Scope

This document specifies the crypto suite for AES-128 for the ISO/IEC 18000 air interfaces standards for radio frequency identification (RFID) devices. Its purpose is to provide a common crypto suite for security for RFID devices that might be referred by ISO committees for air interface standards and application standards.

This document specifies a crypto suite for AES-128 for an air interface for RFID systems. The crypto suite is defined in alignment with existing air interfaces.

This document specifies various authentication methods and methods of use for the encryption algorithm. A Tag and an Interrogator can support one, a subset, or all of the specified options, clearly stating what is supported.

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 9797-1, *Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher*

ISO/IEC 18000-63, *Information technology — Radio frequency identification for item management — Part 63: Parameters for air interface communications at 860 MHz to 960 MHz Type C*

ISO/IEC 19762, *Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary*

ISO/IEC 29167-1, *Information technology — Automatic identification and data capture techniques — Part 1: Security services for RFID*

3 Terms, definitions, symbols and abbreviated terms

For the purposes of this document, the terms and definitions given in ISO/IEC 19762 and the following apply.

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

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1 Terms and definitions

© ISO/IEC 2017 – All rights reserved

3.1.1
AES-CMAC-96 (key, data)
CMAC generation with input data "data", using initialization vector "IV" and 128-bit key "key", truncating the result by using only the 96 most significant bits from the 128-bit CMAC code

3.1.2
AES-DEC (key, data)
AES in ECB decryption mode of input data "data" and 128-bit key "key"

3.1.3
AES-ENC (key, data)
AES in ECB encryption mode of input data "data" and 128-bit key "key"

3.1.4
AuthenticationBlock
variable that contains information to verify the authenticity of the Tag or the Interrogator

3.1.5
bit string
ordered sequence of 0s and 1s

3.1.6
block cipher
family of functions and their inverse functions that is parameterized by keys

Note 1 to entry: The functions map bit strings of a fixed length to bit strings of the same length.

3.1.7
blocksize
number of bits in an input (or output) block of the block cipher

3.1.8
CBC_{ENC}_AES (IV, key, data)
AES in CBC encryption mode of input data "data", using initialization vector "IV" and 128-bit key "key", according to NIST/SP 800-38A

Note 1 to entry: Output blocks (O_i) are obtained from input blocks (I_i) as follows:

— O_1 = AES-ENC(key, I_1 XOR IV), and

— O_n = AES-ENC(key, I_n XOR $O_{(n-1)}$).

Note 2 to entry: C.2 describes the cipher block chaining.

3.1.9
CBC_{DEC}_AES_{INV} (IV, key, data)
AES in CBC decryption mode of input data "data", using initialization vector "IV" and 128-bit key "key", according to NIST/SP 800-38A

Note 1 to entry: Output blocks (O_i) are obtained from input blocks (I_i) as follows:

— O_1 = AES-DEC(key, I_1) XOR IV, and

— O_n = AES-DEC(key, I_n) XOR $I_{(n-1)}$).

3.1.10
CBC_{ENC}_AES_{INV} (IV, key, data)
CBC in encryption mode using initialization vector "IV" and 128-bit key "key"

Note 1 to entry: Output blocks (O_i) are obtained from input blocks (I_i) as follows:

— O_1 = AES-DEC(key, I_1 XOR IV), and

 © ISO/IEC 2017 – All rights reserved

— O_n = AES-DEC(key, I_n XOR $O_{(n-1)}$).

3.1.11
CBC_{DEC}_AES (IV, key, data)
CBC in decryption mode using initialization vector "IV" and 128-bit key "key"

Note 1 to entry: Output blocks (O_i) are obtained from input blocks (I_i) as follows:

— O_1 = AES-ENC(key, I_1) XOR IV, and

— O_n = AES-ENC(key, I_n) XOR $I_{(n-1)}$)

3.1.12
ciphertext
encrypted plaintext

3.1.13
cipher-based message authentication code
CMAC
algorithm based on a symmetric key block cipher

Note 1 to entry: In this document, data is systematically padded with zero bits before computing the MAC, resulting in the last block of MAC inputs is always complete. Therefore, K1-MAC is always used. It makes the computation of K2-MAC useless.

Note 2 to entry: The computation of the MAC shall comply with the requirements of MAC method 5 in ISO/IEC 9797-1.

3.1.14
Command (Message)
data that the Interrogator sends to a Tag with "Message" as parameter

3.1.15
D
number of 128-bit blocks that can be added to the authentication response as custom data and header

3.1.16
data block
block
sequence of bits whose length is the block size of the block cipher

3.1.17
ENC_key
variable that contains the key that will be used for cryptographic confidentiality protection

Note 1 to entry: This variable shall be used for cryptographic confidentiality protection.

3.1.18
H
number of bits of the header

3.1.19
Header
H bits composed of BlockSize, Offset, Profile and BlockCount

3.1.20
Initialization Vector
IV
input block that some modes of operation require as an additional initial input

© ISO/IEC 2017 – All rights reserved

3.1.21
input block
data that is an input to either the forward cipher function or the inverse cipher function of the block cipher algorithm

3.1.22
Key
string of bits used by a cryptographic algorithm to transform plaintext into ciphertext or vice versa or to produce a message authentication code

3.1.23
KeyID
numerical designator for a single key

3.1.24
Key[KeyID].ENC_key
variable that contains the key that will be used for encryption

Note 1 to entry: This variable shall be used for encryption.

3.1.25
Key[KeyID].MAC_key
key that can be used for cryptographic integrity protection

3.1.26
MAC_key
variable that contains the key that will be used for cryptographic integrity protection

Note 1 to entry: This variable shall be used for cryptographic integrity protection.

3.1.27
Memory Profile
start pointer within the Tag's memory for addressing custom data block

3.1.28
Message
part of the command that is defined by the crypto suite

3.1.29
Mode of Operation
Mode
algorithm for the cryptographic transformation of data that features a symmetric key block cipher algorithm

3.1.30
output block
data that is an output of either the forward cipher function or the inverse cipher function of the block cipher algorithm

3.1.31
Plaintext
ordinary readable text before being encrypted into ciphertext or after being decrypted from ciphertext

3.1.32
Reply (Response)
data that the Tag returns to the Interrogator with "Response" as parameter

3.1.33
Response
part of the reply (stored or sent) that is defined by the crypto suite

© ISO/IEC 2017 – All rights reserved

3.1.34
word
bit string comprised of 16 bits

3.2 Symbols and abbreviated terms

AES Advanced Encryption Standard

CBC Cipher Block Chaining

CMAC Cipher-based Message Authentication Code

DIV integral part of a division

Field[a:b] selection from a string of bits in Field

For a > b, selection of a string of bits from the bit string Field. Selection ranges from bit number a until and including bit number b from the bits of the string in Field, whereby Field[0] represents the least significant bit. For selecting one single bit from Field a=b.

For example, Field[2:0] represents the selection of the three least significant bits of Field.

FIPS Federal Information Processing Standard

IV Initialization Vector

LSB Least Significant Byte

MAC Message Authentication Code

MPI Memory Profile Indicator

MSB Most Significant Byte

NIST National Institute of Standards and Technology (United States)

RFU Reserved for Future Use

TID Tag-IDentification or Tag IDentifier (depending on context)

UII Unique Identification ID

$xxxx_b$ binary notation of term "xxxx", where "x" represents a binary digit

$xxxx_h$ hexadecimal notation of term "xxxx", where "x" represents a hexadecimal digit

In this crypto suite, the bytes in the hexadecimal numbers are presented with the most significant byte at the left and the least significant byte at the right. The bit order per byte is also presented with the most significant bit at the left and the least significant bit at the right.

For example, in "ABCDEF" the byte "AB" is the most significant byte and the byte "EF" is the least significant byte.

|| concatenation of syntax elements, transmitted in the order written (from left to right)

For example, "123456" || "ABCDEF" results in "123456ABCDEF", where the byte "12" is the most significant byte and the byte "EF" is the least significant byte.

Note 1 to entry This protocol uses the following notational conventions:

— States and flags are denoted in bold. Some command parameters are also flags; a command parameter used as a flag will be bold. Example: **ready**.

© ISO/IEC 2017 – All rights reserved

— Command parameters are underlined. Some flags are also command parameters; a flag used as a command parameter will be underlined. Example: Pointer.

— Commands are denoted in italics. Variables are also denoted in italics. Where there might be confusion between commands and variables, this protocol will make an explicit statement. Example: *Query*.

4 Conformance

4.1 Air interface protocol specific information

To claim conformance with this document, an Interrogator or Tag shall comply with all relevant clauses of this document, except those marked as “optional”.

4.2 Interrogator conformance and obligations

To conform to this document, an Interrogator shall implement the mandatory commands defined in this document and conform to the relevant part of ISO/IEC 18000.

To conform to this document, an Interrogator can implement any subset of the optional commands defined in this document.

To conform to this document, the Interrogator shall not

— implement any command that conflicts with this document, or

— require the use of an optional, proprietary or custom command to meet the requirements of this document.

4.3 Tag conformance and obligations

To conform to this document, a Tag shall implement the mandatory commands defined in this document for the supported types and conform to the relevant part of ISO/IEC 18000.

To conform to this document, a Tag can implement any subset of the optional commands defined in this document.

To conform to this document, a Tag shall not

— implement any command that conflicts with this document, or

— require the use of an optional, proprietary or custom command to meet the requirements of this document.

5 Introduction of the AES-128 crypto suite

The Advanced Encryption Standard (AES) is an open, royalty-free, symmetric block cipher based on so-called substitution-permutation networks. AES is highly suitable for efficient implementation in both software and hardware, including extremely constrained environments such as RFID Tags. The AES cipher is standardized as ISO/IEC 18033-3.

AES is approved by the National Institute of Standards and Technology (NIST). It was approved as a standard in 2001 following a 5-year standardization process that involved a number of competing encryption algorithms and published as FIPS PUB 197 in November 2001.

© ISO/IEC 2017 – All rights reserved

AES was published, along with design criteria and test vectors, in Reference [2].

NOTE AES normally uses encryption to encrypt plaintext and decryption to decrypt ciphertext. This crypto suite uses encryption both to encrypt plaintext as well as to decrypt ciphertext. This allows the use of an encryption-only implementation on the Tag.

References for AES test vectors are provided in Annex D.

Annex F provides examples for the implementation of the functionality that is specified in this document.

6 Parameter definitions

Table 1 describes all the parameters that are used in this document.

Table 1 — Definition of AES-128 crypto suite parameters

Parameter	Description
AuthenticationBlock	Parameter used in IResponse of IAM3 Message with the parameters: AES-DEC(Key[KeyID].*ENC_key*, *C_IAM3*[11:0] || *Purpose_IAM3*[3:0] || *IRnd_IAM3*[31:0] || *TChallenge_IAM1*[79:0]) This parameter is only introduced to make the content of the IResponse of IAM3 Message easier to read.
C_MAM1[15:0]	16-bit predefined constant for MAM1 with the value "$DA83_h$" (for Tag to Interrogator response)
C_MAM2[11:0]	12-bit predefined constant for MAM2 with the value "$DA8_h$" (for Tag to Interrogator response)
C_TAM1[15:0]	16-bit predefined constant for TAM1 with the value "$96C5_h$" (for Tag to Interrogator response)
C_TAM2[15:0]	16-bit predefined constant for TAM2 with the value "$96C5_h$" (for Tag to Interrogator response)
C_TAM2_0[15:0]	16-bit predefined constant for TAM2 with the value "$96C0_h$" (for Tag to Interrogator response)
C_TAM2_1[15:0]	16-bit predefined constant for TAM2 with the value "$96C1_h$" (for Tag to Interrogator response)
C_TAM2_2[15:0]	16-bit predefined constant for TAM2 with the value "$96C2_h$" (for Tag to Interrogator response)
C_TAM2_3[15:0]	16-bit predefined constant for TAM2 with the value "$96C3_h$" (for Tag to Interrogator response)
C_IAM2[11:0]	12-bit predefined constant for IAM2 with the value "$DA8_h$" (for Interrogator to Tag response)
C_IAM3_0[11:0]	12-bit predefined constant for IAM3 with the value "$DA8_h$" (for Interrogator to Tag response)
C_IAM3_1[11:0]	12-bit predefined constant for IAM3 with the value "$DA9_h$" (for Interrogator to Tag response)
C_IAM3_2[11:0]	12-bit predefined constant for IAM3 with the value "DAA_h" (for Interrogator to Tag response)
C_IAM3_3[11:0]	12-bit predefined constant for IAM3 with the value "DAB_h" (for Interrogator to Tag response)
CUSTOMDATA(D*128-*H*)	Part of the Tag's memory that may be included in the authentication process
HEADER(*H*)	Header of *H* bits preceding the custom data
IChallenge_MAM1[79:0]	80-bit challenge generated by the Interrogator for use in MAM1
IChallenge_TAM1[79:0]	80-bit challenge generated by the Interrogator for use in TAM1
IChallenge_TAM2[79:0]	80-bit challenge generated by the Interrogator for use in TAM2

© ISO/IEC 2017 – All rights reserved

Table 1 *(continued)*

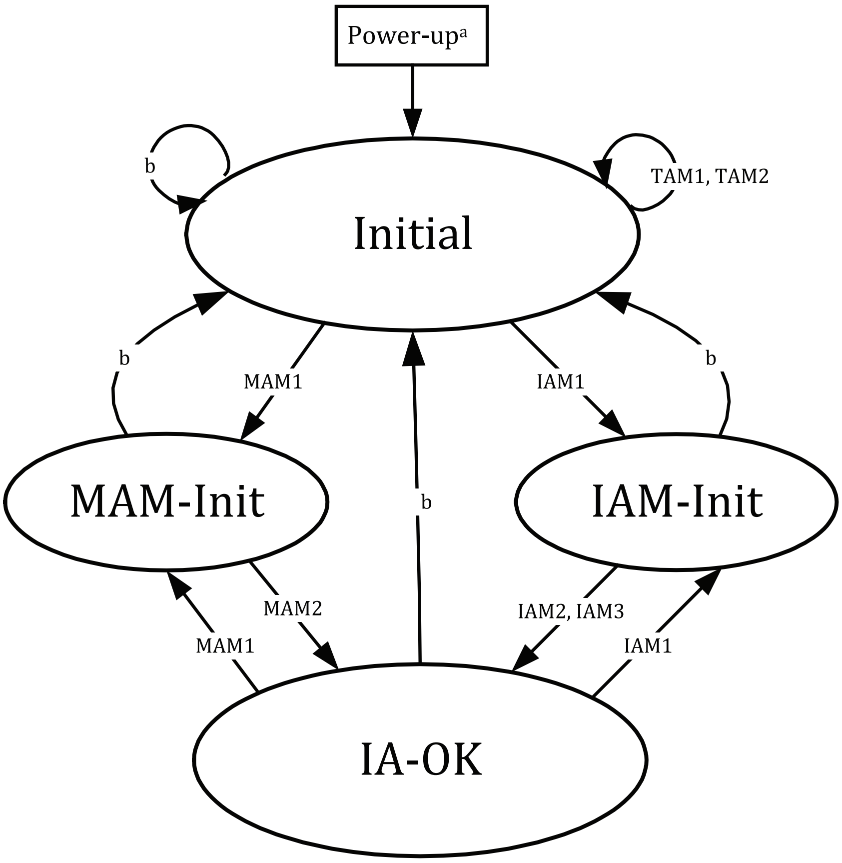
Parameter	**Description**
IRnd_IAM2[31:0]	32-bit random data generated by the Interrogator for use in IAM2
IRnd_IAM3[31:0]	32-bit random data generated by the Interrogator for use in IAM3
Key[KeyID]	Keyset identified by KeyID, consisting of *ENC_key* for encryption and (optional) *MAC_key* for integrity protection
MAC_key[127:0]	Variable that shall contain the key that will be used for cryptographic integrity protection
Purpose_IAM2[3:0]	Authentication purpose bits for IAM2 If *Purpose_IAM2*[3:3] = 0_b the bits [2:0] are RFU with value 000_b If *Purpose_IAM2*[3:3] = 1_b the bits [2:0] are manufacturer defined
Purpose_IAM3[3:0]	Authentication purpose bits for IAM3 If *Purpose_IAM3*[3:3] = 0_b the bits [2:0] are RFU with value 000_b If *Purpose_IAM3*[3:3] = 1_b the bits [2:0] are manufacturer defined
Purpose_MAM2[3:0]	Authentication purpose bits for MAM2 If *Purpose_MAM2*[3:3] = 0_b the bits [2:0] are RFU with value 000_b If *Purpose_MAM2*[3:3] = 1_b the bits [2:0] are manufacturer defined
TChallenge_IAM1[79:0]	80-bit challenge that the Tag generates for use in IAM1
TChallenge_MAM1[79:0]	80-bit challenge that the Tag generates for use in MAM1
TRnd_TAM1[31:0]	32-bit random data provided by the Tag for TAM1
TRnd_TAM2[31:0]	32-bit random data provided by the Tag for TAM2

7 Crypto suite state diagram

The transitions between the crypto suite states are specified in Figure 1.

The Tag shall transition from the Start State to the Next State conforming to the requirements specified in Annex A.

© ISO/IEC 2017 – All rights reserved



Key

a All variable fields will be reset at power-up.

b All errors result in a transition to **Initial** state.

Figure 1 — Crypto suite Tag state diagram

The Interrogator is considered authenticated only in the **IA-OK state**.

8 Initialization and resetting

After power-up and after a reset, the crypto suite shall transition into the **Initial** state.

After the Tag encounters an error condition, it shall transition into the **Initial** state.

After the Tag encounters an error condition, it may send an error reply to the Interrogator, but in that case the Tag shall select one Error Condition from the list that is specified in Annex B.

A transition to **Initial** state shall also cause a reset of all variables used by the crypto suite.

Implementations of this crypto suite shall assure that all memory used for intermediate results is cleared after each operation (message-response pair) and after reset.

9 Authentication

9.1 General

This document supports Tag Authentication, Interrogator Authentication and Mutual Authentication. All functions are implemented using a message-response exchange. This clause describes the details of the messages and responses that are exchanged between the Interrogator and Tag.

© ISO/IEC 2017 – All rights reserved

All message and response exchanges are listed in Table 2.

Table 2 — Message and response functions

Command	Function
TAM1 message	Send Interrogator challenge and request Tag authentication response
TAM1 response	Return Tag authentication response
TAM2 message	Send Interrogator challenge and request Tag authentication response with custom data
TAM2 response	Return Tag authentication response and custom data
IAM1 message	Send Interrogator authentication request
IAM1 response	Return Tag challenge
IAM2 message	Send Interrogator authentication response
IAM2 response	Return Interrogator Authentication result
IAM3 message	Send Interrogator authentication response plus custom data
IAM3 response	Return Interrogator Authentication result
MAM1 message	Send Interrogator challenge and request Tag authentication response and challenge
MAM1 response	Return Tag authentication response and challenge
MAM2 message	Send Interrogator authentication response
MAM2 response	Return Interrogator Authentication result

9.2 Adding custom data to authentication process

This document supports including part of the Tag's memory as custom data to the authentication process. The custom data may be protected (protection of integrity and authenticity) and/or encrypted (confidentiality protection) with the authentication. The authentication message shall include the reference KeyID to select an encryption key in Table 27 (see Clause 11). If protection of integrity and authenticity of the data is requested, the selected reference KeyID shall also contain a MAC key.

A Tag that supports including custom data in the authentication process shall define at least one and at most 16 memory profiles. All supported addresses or pointers for the memory profiles are specified in Annex E.

The memory profiles may also be linked to a key in Table 27 that shall be used for the encryption process to protect the data.

Custom data is specified as a number (1 to 16) of consecutive 16-bit or 64-bit blocks in the Tag's memory. The custom data block shall be defined by the parameters BlockSize, Profile, Offset and BlockCount. The mode of operation that shall be used for the encryption and/or protection of the custom data is specified by ProtMode.

BlockSize shall select the size of the custom data block; "0_b" specifies custom data in 64-bit blocks, "1_b" specifies custom data as 16-bit blocks.

Profile shall select one of the memory profiles that are supported by the Tag. The memory profiles are specified in Annex E.

Offset specifies a 12-bit offset (in multiples of 16-bit or 64-bit blocks) that needs to be added to the address that is specified by Profile. Minimum value "000000000000_b" corresponds to a zero offset. Maximum value is 111111111111_b (decimal 4095). For 16-bit blocks, the maximum bit pointer offset is 16 × 4095 = 65520. For 64-bit blocks, the maximum bit pointer offset is 64 × 4095 = 262080.

BlockCount specifies the 4-bit number of custom data blocks that need to be selected from the offset position onwards. Minimum value is "0000_b", corresponding to one single block. Maximum binary value is "1111_b", or decimal 15, corresponds to a maximum number of 16 blocks of custom data that shall be included. If the number of included bits of the custom data including header is not a multiple of 128 then padding with zeroes shall be applied to the least significant bits of the last block that has

 © ISO/IEC 2017 – All rights reserved

a non-zero block size of less than 128 bits. The Interrogator shall maintain the value of BlockCount for use as part of the MAC verification process. The Tag manufacturer shall specify the number of custom data blocks that can be included.

NOTE When combined with the values for BlockSize and BlockCount, as well as requiring that all component blocks are complete, the padding scheme proposed in this document is unambiguous.

HEADER is composed of the concatenation of the BlockSize, Profile, Offset, BlockCount and extra zeroes padding.

If BlockSize = "0_b", HEADER is a 64-bit value defined as:

HEADER = BlockSize[0:0] || Profile[3:0] || Offset[11:0] || BlockCount[3:0] || 00000000000_b || 00000000_h

If BlockSize = "1_b", Header is a 32-bit value defined as:

HEADER = BlockSize[0:0] || Profile[3:0] || Offset[11:0] || BlockCount[3:0] || 00000000000_b

H represents the number of bits of HEADER. If BlockSize = "0_b", then $H = 64$, else $H = 32$.

D represents the number of 128-bit custom data blocks including the header, when present, and is defined by BlockCount and BlockSize. The minimum value of *D* shall be 1. The maximum value of *D* supported by the Tag is specified by the Tag manufacturer.

If *n* represents the decimal value of BlockCount (0 to 15), then *D* is as follows:

— For TAM2

  — If BlockSize = 0_b and TAM2_Rev=0, then *D*: = (n+1) DIV 2 + (n+1) MOD2

  — If BlockSize = 1_b and TAM2_Rev=0, then *D*: = (n+8) DIV 8

  — If BlockSize = 0_b and TAM2_Rev=1, then *D*: = (n+2) DIV 2 + (n+2) MOD2

  — If BlockSize = 1_b and TAM2_Rev=1, then *D*: = (n+10) DIV 8

— For IAM3

  — If BlockSize = 0_b and TAM2_Rev=1, then *D*: = (n+2) DIV 2 + (n+2) MOD2

  — If BlockSize = 1_b and TAM2_Rev=1, then *D*: = (n+10) DIV 8

CUSTOMDATA(*D**128) contains the custom data as a bit string with a length of *D**128-*H* bits (including padding) for IAM3 and TAM2 command and TAM2_Rev=1.

CUSTOMDATA(*D**128) contains the custom data as a bit string with a length of *D**128 bits (including padding) for TAM2 command and TAM2_Rev=0.

NOTE Access rights to custom data can be restricted by the specification of the interface. Annex E describes protocol-specific implementations for various modes of the ISO/IEC 18000 series.

ProtMode specifies the mode of operation that shall be used for the encryption and/or protection of the custom data. Table 3 specifies the mode of operation for encryption algorithms and/or message authentication algorithms for the (optional) protection (authentication and/or encryption) of custom data.

© ISO/IEC 2017 – All rights reserved

Table 3 — Supported modes of operation for ProtMode

ProtMode[3:0][a]	Description
0000_b	Plaintext (no integrity and/or confidentiality protection requested)
0001_b	CBC (encryption only)
0010_b	CMAC (message authentication only)
0011_b	CBC + CMAC
0100_b	Reserved for future use
....	....
0111_b	Reserved for future use
1000_b	Manufacturer defined
....	....
1111_b	Manufacturer defined

[a] When a ProtMode is selected that specifies data encryption (ProtMode "0001_b" and "0011_b"), the Tag may assert a privilege that makes all memory accessible for the duration of the execution of the command. See Annex E for details of the air interface specific implementations.

NOTE This document specifies the use of an encryption-only implementation on the Tag.

9.3 Message and response formatting

The functionality of this document is implemented by means of a *Command*(Message)/(Response) exchange (see Figure 2) as described in the air interface specification.

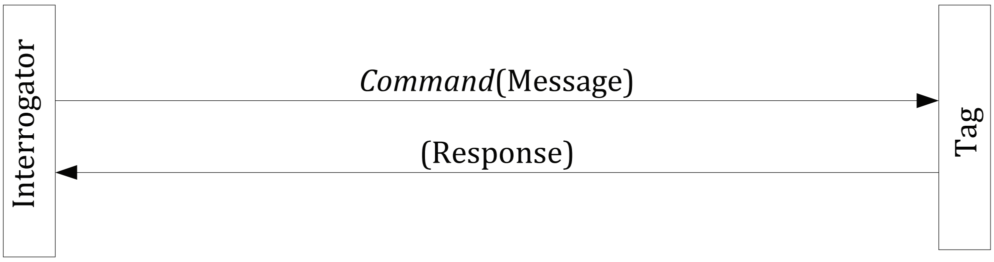


Figure 2 — Command(Message)/Response exchange

The "air interface part" of the Tag passes the Message on to the "crypto suite part" of the Tag and returns the Response from the "crypto suite part" as a Reply to the Interrogator. The crypto suite shall parse the Messages and process the data based on the value of AuthMethod, which is the first parameter (first two bits) of all Messages.

The following subclauses describe the formatting of Message and Response for Tag Authentication, Interrogator Authentication and Mutual Authentication. The Messages for Tag Authentication, Interrogator Authentication and Mutual Authentication shall be distinguished by AuthMethod.

If AuthMethod = "00_b", the Tag shall parse the Message for Tag Authentication as described in 9.4.

If AuthMethod = "01_b", the Tag shall parse Message for Interrogator Authentication as described in 9.5.

If AuthMethod = "10_b", the Tag shall parse Message for Mutual Authentication as described in 9.6.

If AuthMethod = "11_b", then the Tag shall return a "Not Supported" error condition.

© ISO/IEC 2017 – All rights reserved

9.4 Tag authentication (Method "00" = TAM)

9.4.1 General

Tag Authentication allows an Interrogator to authenticate a Tag by verifying the Tag's secret key with the TAM1 response. Optionally, the Tag may return part of its memory as custom data, which may be protected (protection of integrity, authenticity of origin and timeliness) and/or encrypted (confidentiality protection), with the TAM2 response.

Tag authentication only is implemented in TAM1 and Tag authentication with the addition of custom data is implemented as TAM2.

The following subclauses describe the formatting of Message and Response for Tag Authentication.

The TAM Messages are distinguished by the value of CustomData, which is the second parameter (third bit) of both TAM Messages.

If CustomData = "0_b", the Tag shall parse the TAM1 Message for Tag Authentication without custom data as described in 9.4.2.

If CustomData = "1_b", the Tag shall parse the TAM2 Message for Tag Authentication with custom data as described in 9.4.5.

9.4.2 TAM1 Message

For Tag authentication, the Interrogator shall generate an 80-bit random TAM1 Interrogator challenge and include that in the TAM1 message. The TAM1 message shall also include the reference KeyID to select an encryption key in Table 27 (see Clause 11).

The TAM1 Message format is specified in Table 4 and has the following parameters:

— AuthMethod: value "00_b" specifies the use for TAM;

— CustomData: flag with value "0_b" to indicate that no custom data is requested (TAM1);

— TAM1_RFU: value "00000_b" makes the total length of the TAM1 Message a multiple of 8-bits and might be used for future extensions of this document;

— KeyID: 8-bit value that specifies the key that shall be used for TAM1;

— IChallenge_TAM1: 80-bit random challenge that the Interrogator has generated for use in TAM1.

Table 4 — TAM1 Message format

	AuthMethod	CustomData	TAM1_RFU	KeyID	IChallenge_TAM1
# of bits	2	1	5	8	80
Description	00_b	0_b	00000_b	[7:0]	random Interrogator challenge

The Tag shall accept this message in any state. If the value of the parameters of the message are invalid, then the Tag shall transition to the **Initial** state, thereby aborting any cryptographic protocol that has not yet been completed.

If the length of the TAM1 message is <> 96 bits, then the Tag shall return an "Other Error" error condition.

If TAM1_RFU[4:0] is <> "00000_b", then the Tag shall return a "Not Supported" error condition.

If the Tag does not support key[KeyID].*ENC_key*, then the Tag shall return a "Not Supported" error condition.

© ISO/IEC 2017 – All rights reserved

9.4.3 TAM1 Response

If all parameters have been successful verified, then the Tag shall generate a response as specified in Table 5. The Tag shall generate the random data *TRnd_TAM1*[31:0] and encrypt the concatenation of the constant C_TAM1[15:0], the random data *TRnd_TAM1*[31:0] and the challenge IChallenge_TAM1[79:0] using Key[KeyID].*ENC_key.*

Table 5 — TAM1 Response

	TResponse
# of bits	128
Description	AES-ENC(Key[KeyID].*ENC_key*, C_TAM1[15:0] || *TRnd_TAM1*[31:0] || IChallenge_TAM1[79:0])

After returning the TAM1 Response (TResponse), the Tag shall remain in the **Initial** state.

9.4.4 Final Interrogator processing TAM1

The Interrogator (or the external application controlling the Interrogator) decrypts the TAM1 Response (TResponse) and shall verify whether C_TAM1 and IChallenge_TAM1 have the correct value. If the values are correct, then the Tag can be considered as authentic.

9.4.5 TAM2 Message

TAM2 is used for Tag Authentication if the Tag needs to return part of its memory as custom data. The custom data is specified in 9.2 and may be protected (protection of integrity and authenticity of origin) and/or decrypted (confidentiality protection).

The Interrogator shall generate an 80-bit random number for use as TAM2 Interrogator challenge.

The TAM2 Message format is specified in Table 6 and has the following parameters:

— AuthMethod[1:0]: value "00_b" specifies the use for TAM;

— CustomData[0:0]: value "1_b" specifies that custom data is included in the Message (TAM2);

— BlockSize[0:0]: indicator that defines the size of the custom data block, "0_b" specifies custom data as 64-bit block, "1_b" specifies custom data as 16-bit block. The support of 64-bit blocks is mandatory, the support of 16-bit blocks is optional;

— TAM2_Rev[0:0]: indicator that defines the TAM2 Message format to be used, "0_b" specifies to use the legacy TAM2 message format as published in ISO/IEC 29167-10:2015, "1_b" specifies to use the revised TAM2 message format in this document;

  The protocol instantiated in the case TAM2_Rev=1 is expected to be more cryptographically robust than the legacy version with TAM2_Rev=0. A certificational weakness in the TAM2 processing has been identified and while the practical impact may be low under typical usage conditions for TAM2_Rev=0, it can be further reduced or eliminated using TAM2_Rev=1 which is recommended henceforth.

— TAM2_RFU[2:0]: value "000_b" makes the total length of the TAM2 Message a multiple of 8-bits and might be used for future extensions of this document;

— KeyID[7:0]: value that specifies the key that will be used for TAM2;

— IChallenge_TAM2[79:0]: random challenge that the Interrogator has generated for use in TAM2;

— Profile[3:0]: pointer that selects a memory profile for the addition of custom data as specified in E.4.5;

— Offset[11:0]: value that specifies the number of multiples of 16-bit or 64-bit blocks that needs to be added to the address that is specified by Profile to define the first address of the custom data block;

© ISO/IEC 2017 – All rights reserved

— BlockCount[3:0]: number that defines the size of the custom data as a number of 16-bit or 64-bit blocks. If the number of included bits of the custom data including header is not a multiple of 128 then padding with zeroes shall be applied to the least significant bits of the last block that has a non-zero block size of less than 128 bits. The Interrogator shall maintain the value of BlockCount for use as part of the MAC verification process. The Tag manufacturer shall specify the number of custom data blocks that can be included;

— ProtMode[3:0]: value to select the mode of operation that shall be used to process the custom data as specified in Table 3.

Table 6 — TAM2 Message format

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00_b	1_b	[0:0]	[0:0]	000_b	[7:0]	random Interrogator challenge	[3:0]	[11:0]	[3:0]	[3:0]

The Tag shall accept this message in any state. If the parameters of the message are invalid, then the Tag shall transition to the **Initial** state, thereby aborting any cryptographic protocol that has not yet been completed.

If the length of the TAM2 message is <> 120 bits, then the Tag shall return an "Other Error" error condition.

If BlockSize = "1_b" and the Tag does not support value "1_b", then the Tag shall return a "Not Supported" error condition.

If TAM2_Rev specifies a TAM2 message format that is not supported by the Tag, then the Tag shall return a "Not Supported" error condition.

If TAM2_RFU[2:0] is <> "000_b", then the Tag shall return a "Not Supported" error condition.

If the Tag does not support key[KeyID].*ENC_key*, then the Tag shall return a "Not Supported" error condition.

If the memory profile specified in Profile is not supported by the Tag, then the Tag shall return a "Not Supported" error condition.

The Tag shall check if the specified memory profile has the right to use KeyID for further processing:

if Profile = "0000_b" and key[KeyID].MPI[0:0] = "1_b" or

if Profile = "0001_b" and key[KeyID].MPI[1:1] = "1_b" or

if Profile = "0010_b" and key[KeyID].MPI[2:2] = "1_b" or

if Profile = "0011_b" and key[KeyID].MPI[3:3] = "1_b" or

if Profile = "0100_b" and key[KeyID].MPI[4:4] = "1_b"or

if Profile = "0101_b" and key[KeyID].MPI[5:5] = "1_b" or

if Profile = "0110_b" and key[KeyID].MPI[6:6] = "1_b" or

if Profile = "0111_b" and key[KeyID].MPI[7:7] = "1_b" or

if Profile = "1000_b" and key[KeyID].MPI[8:8] = "1_b" or

if Profile = "1001_b" and key[KeyID].MPI[9:9] = "1_b" or

if Profile = "1010_b" and key[KeyID].MPI[10:10] = "1_b" or

© ISO/IEC 2017 – All rights reserved

if Profile = "1011_b" and key[KeyID].MPI[11:11] = "1_b" or

if Profile = "1100_b" and key[KeyID].MPI[12:12] = "1_b" or

if Profile = "1101_b" and key[KeyID].MPI[13:13] = "1_b" or

if Profile = "1110_b" and key[KeyID].MPI[14:14] = "1_b" or

if Profile = "1111_b" and key[KeyID].MPI[15:15] = "1_b",

then the key[KeyID] is authorized for this memory profile,

else key[KeyID] is not authorized for this memory profile and the Tag shall return a "Not Supported" error condition.

If the block of custom data specified by BlockSize, Profile, Offset and BlockCount is not supported by the Tag, then the Tag shall return a "Memory Overrun" error condition.

If the ProtMode value is not supported by the Tag, then the Tag shall return a “Not Supported” error condition.

9.4.6 TAM2 Response

9.4.6.1 General

If all parameters have been successful verified, then the Tag shall proceed with parsing the TAM2 message.

If TAM2_Rev = "0_b", then the Tag encrypts the concatenation of *C_TAM2*[15:0], *TRnd_TAM2*[31:0] and IChallenge_TAM2[79:0] and the custom data represented by *CUSTOMDATA*(*D**128). If TAM2_Rev = "1_b", then the constant C_TAM2 is replaced by one of the four constants, C_TAM2_0[15:0], C_TAM2_1[15:0], C_TAM2_2[15:0] or C_TAM2_3[15:0], as a function of the ProtMode value and the Tag encrypts the custom data represented by *CUSTOMDATA*(*D**128-*H*).

After returning the TAM2 Response (TResponse), the Tag shall remain in the **Initial** state.

9.4.6.2 Response if TAM2_Rev = "0_b" and ProtMode = "0000_b": Plaintext

The Tag shall add custom data in plaintext to the authentication block and generate a response as specified in Table 7.

The KeyID identifies the encryption key for AES encryption.

Table 7 — Response if TAM2_Rev = "0_b" and ProtMode = "0000_b": Plaintext

	TResponse
# of bits	128 + *D**128
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || *CUSTOMDATA*(*D**128)

9.4.6.3 Response if TAM2_Rev = "0_b" and ProtMode = "0001_b": CBC encryption only

The Tag shall add custom data with confidentiality protection to the authentication block and generate a response as specified in Table 8.

The Tag shall use AES encryption in CBC mode to encrypt all *D* custom data blocks.

The KeyID identifies the encryption key for AES encryption in CBC mode.

 © ISO/IEC 2017 – All rights reserved

Table 8 — Response if TAM2_Rev = "0_b" and ProtMode = "0001_b": CBC encryption only

	TResponse
# of bits	128 + *D**128
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV = AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, *CUSTOMDATA*(*D**128))

9.4.6.4 Response if TAM2_Rev = "0_b" and ProtMode = "0010_b": CMAC message authentication only

The Tag shall add custom data with integrity protection to the authentication block and generate a response as specified in Table 9.

Parameter KeyID identifies key[KeyID].*ENC_key* for AES encryption and key[KeyID].*MAC_key* for CMAC computation.

The Tag shall use AES-CMAC-96 to calculate the truncated 96-bit CMAC over the authentication block and the *D* following plaintext custom data blocks.

Table 9 — Response if TAM2_Rev = "0b" and ProtMode = "0010b": CMAC message authentication only

	TResponse
# of bits	128 + *D**128 + 96
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || *CUSTOMDATA*(*D**128) || AES-CMAC-96(Key[KeyID].*MAC_key*, AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || *CUSTOMDATA*(*D**128))

9.4.6.5 Response if TAM2_Rev = "0_b" and ProtMode = "0011_b": CBC encryption with CMAC message authentication

The Tag shall add custom data with confidentiality and integrity protection to the authentication block and generate a response as specified in Table 10.

Parameter KeyID identifies key[KeyID].*ENC_key* for AES encryption and key[KeyID].*MAC_key* for CMAC computation.

The Tag shall use AES encryption in CBC mode to encrypt the initial authentication block and all following *D* custom data blocks.

The Tag shall use AES-CMAC-96 to calculate the truncated 96-bit CMAC over the authentication block and the *D* following encrypted custom data blocks.

© ISO/IEC 2017 – All rights reserved

Table 10 — Response if TAM2_Rev = "0_b" and ProtMode = "0011_b": CBC encryption with CMAC message authentication

	TResponse
# of bits	128 + D*128 + 96
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV= AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, *CUSTOMDATA*(D*128)) || AES-CMAC-96(Key[KeyID].*MAC_key*, AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV= AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, *CUSTOMDATA*(D*128))

9.4.6.6 Response if TAM2_Rev = "1_b" and ProtMode = "0000_b": Plaintext

The tag shall compute the authentication block as the encryption of *C_TAM2_0*[15:0], *TRnd_TAM2*[31:0] and IChallenge_TAM2[79:0].

The Tag shall add the header and the custom data in plaintext to the authentication block and generate a response as specified in Table 11.

The KeyID identifies the encryption key for AES encryption.

Table 11 — Response if TAM2_Rev = "1_b" and ProtMode = "0000_b": Plaintext

	TResponse
# of bits	128 + D*128
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_0*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || HEADER(H) || *CUSTOMDATA*(D*128-H)

9.4.6.7 Response if TAM2_Rev = "1_b" and ProtMode = "0001_b": CBC encryption only

The tag shall compute the authentication block as the encryption of *C_TAM2_1*[15:0], *TRnd_TAM2*[31:0] and IChallenge_TAM2[79:0].

The Tag shall add the header and the custom data with confidentiality protection to the authentication block and generate a response as specified below and in Table 12.

The Tag shall use AES encryption in CBC mode to encrypt all D data blocks composed of the header and the custom data.

The KeyID identifies the encryption key for AES encryption in CBC mode.

 © ISO/IEC 2017 – All rights reserved

Table 12 — TAM2_Rev = "1_b" and Response if ProtMode = "0001_b": CBC encryption only

	TResponse
# of bits	128 + *D**128
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_1*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV = AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_1*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*))

9.4.6.8 Response if TAM2_Rev = "1_b" and ProtMode = "0010_b": CMAC message authentication only

The Tag shall compute the authentication block as the encryption of *C_TAM2_2*[15:0], *TRnd_TAM2*[31:0] and IChallenge_TAM2[79:0]

The Tag shall add the header and the custom data with integrity protection to the authentication block and generate a response as specified below and in Table 13.

Parameter KeyID identifies key[KeyID].*ENC_key* for AES encryption and key[KeyID].*MAC_key* for CMAC computation.

The Tag shall use AES-CMAC-96 to calculate the truncated 96-bit CMAC over the authentication block and the *D* following plaintext data blocks composed of the header and the custom data.

Table 13 — Response if TAM2_Rev = "1_b" and ProtMode = "0010_b": CMAC message authentication only

	TResponse
# of bits	128 + *D**128 + 96
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || HEADER(*H*)*CUSTOMDATA*(*D**128-*H*) || AES-CMAC-96(Key[KeyID].*MAC_key*, AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_2*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*))

9.4.6.9 Response if TAM2_Rev = "1_b" and ProtMode = "0011_b": CBC encryption with CMAC message authentication

The Tag computes the authentication block as the encryption of *C_TAM2_3*[15:0], *TRnd_TAM2*[31:0] and IChallenge_TAM2[79:0].

The Tag shall add the header and the custom data with confidentiality and integrity protection to the authentication block and generate a response as specified below and in Table 14.

Parameter KeyID identifies key[KeyID].*ENC_key* for AES encryption and key[KeyID].*MAC_key* for CMAC computation.

The Tag shall use AES encryption in CBC mode to encrypt the initial authentication block and all following *D* data blocks composed of the header and the custom data.

The Tag shall use AES-CMAC-96 to calculate the truncated 96-bit CMAC over the authentication block and the *D* following encrypted custom data blocks.

© ISO/IEC 2017 – All rights reserved

Table 14 — Response if TAM2_Rev = "1_b" and ProtMode = "0011_b": CBC encryption with CMAC message authentication

	TResponse
# of bits	128 + *D**128 + 96
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_3*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV= AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_3*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*)) || AES-CMAC-96(Key[KeyID].*MAC_key*, AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_3*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]) || CBC_{ENC}_AES (IV= AES-ENC(Key[KeyID].*ENC_key*, *C_TAM2_3*[15:0] || *TRnd_TAM2*[31:0] || IChallenge_TAM2[79:0]), Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*))

9.4.7 Final Interrogator processing TAM2

9.4.7.1 General

If ProtMode = 0010_b or ProtMode = 0011_b, the Interrogator (or the external application controlling the Interrogator) first checks the supplied MAC for correctness and aborts if MAC verification fails. The Interrogator (or the external application controlling the Interrogator) then decrypts the TAM2 Response (TResponse) according to the value of TAM2_Rev.

9.4.7.2 Final Interrogator processing for TAM2_Rev = "0_b"

The Interrogator (or the external application controlling the Interrogator) decrypts the TAM2 Response (TResponse) and shall verify whether *C_TAM2* and IChallenge_TAM2 have the correct value. If the values are correct, then the Tag can be considered as authentic and the custom data can be accepted. Otherwise, the interrogator aborts and does consider the Tag and the custom data as invalid.

Note that the confidentiality of the custom data is guaranteed only if ProtMode 0001_b or 0011_b and the data of the tag is only readable in encrypted mode. The integrity of the custom data is guaranteed if ProtMode = 0010_b or 0011_b. The integrity of the custom data is not guaranteed if ProtMode = 0001_b.

9.4.7.3 Final Interrogator processing for TAM2_Rev = "1_b"

The Interrogator (or the external application controlling the Interrogator) decrypts the first block of TAM2 Response (TResponse) and shall verify whether *C_TAM2 constant* and IChallenge_TAM2 have the correct value. For the C_TAM2 constant, the interrogator verifies that it corresponds to the ProtMode value:

If ProtMode = 0000_b, C_TAM2 shall be C_TAM2_0.

If ProtMode = 0001_b, C_TAM2 shall be C_TAM2_1.

If ProtMode = 0010_b, C_TAM2 shall be C_TAM2_2.

If ProtMode = 0011_b, C_TAM2 shall be C_TAM2_3

If the values are not correct, the interrogator aborts. If the values are correct and ProtMode = 0001_b or 0011_b, the interrogator decrypts the remaining blocks *D* of TAM2 response.

Then it verifies that HEADER is correct.

 © ISO/IEC 2017 – All rights reserved

If the values are correct, then the Tag can be considered as authentic and the custom data can be accepted. Otherwise, the interrogator aborts and does consider the Tag and the custom data as invalid.

NOTE The confidentiality of the custom data is guaranteed only if ProtMode 0001_b or 0011_b and the data of the tag is only readable in encrypted mode. The integrity of the custom data is guaranteed if ProtMode = 0010_b or 0011_b.

9.5 Interrogator authentication (Method “01” = IAM)

9.5.1 General

The IAM Messages are distinguished by the value of Step, the next 2-bit parameter in the Message after AuthMethod.

If Step = "00_b", the Tag shall parse the IAM1 Message for Interrogator Authentication as described in 9.5.2.

If Step = "01_b", the Tag shall parse the IAM2 and IAM3 Messages and process the data based on the value of CustomData, which is the third parameter in the IAM2 and IAM3 Messages.

If Step = "01_b" and CustomData = "0_b", the Tag shall parse the IAM2 Message for Interrogator Authentication without custom data as described in 9.5.5

If Step = "01_b" and CustomData = "1_b", the Tag shall parse the IAM3 Message for Interrogator Authentication with custom data as described in 9.5.8

If Step = "10_b", the Tag shall return a "Not Supported" error condition.

If Step = "11_b", the Tag shall return a "Not Supported" error condition.

9.5.2 IAM1 Message

To initiate the interrogator authentication, the Interrogator sends a request to get a challenge from the Tag.

The IAM1 Message format is specified in Table 15 and has the following parameters:

— AuthMethod[1:0]: value "01_b" specifies the use for IAM;

— Step[1:0]: value "00_b" specifies the use for the IAM1;

— IAM1_RFU[3:0]: value "0000_b" makes the total length of the IAM1 Message a multiple of 8-bits and might be used for future extensions of this document;

— KeyID[7:0]: identifier of the key in Table 27 (see Clause 11).

Table 15 — IAM1 Message format

	AuthMethod	Step	IAM1_RFU	KeyID
# of bits	2	2	4	8
Description	01_b	00_b	0000_b	[7:0]

The Tag shall accept this message only in the **Initial** or the **IA-OK** state (unless occupied by internal processing and not capable of receiving messages). If the parameters of the message are invalid, then the Tag shall transition to the **Initial** state, thereby aborting any cryptographic protocol that has not yet been completed.

If the length of the IAM1 message is <> 16 bits, then the Tag shall return an "Other Error" error condition.

If the value of IAM1_RFU[3:0] is <> "0000_b", then the Tag shall return a "Not Supported" error condition.

If the Tag does not support key[KeyID].*ENC_key*, then it shall return a "Not Supported" error condition.

© ISO/IEC 2017 – All rights reserved

9.5.3 IAM1 Response

The Tag shall generate a random challenge *TChallenge_IAM1*[79:0] and store a copy of *TChallenge_IAM1* for subsequent verification (see 9.5.5 or 9.5.8).

The Tag shall store a copy of KeyID for use in 9.5.5 or 9.5.8.

The Tag shall send the challenge *TChallenge_IAM1* in the IAM1 Response as specified in Table 16.

Table 16 — IAM1 Response format

	TResponse
# of bits	80
Description	*TChallenge_IAM1*[79:0]

After returning the IAM1 Response (TResponse), the Tag shall transition to the **IAM-Init** state.

9.5.4 Final Interrogator processing IAM1

The Interrogator (or the external application controlling the Interrogator) shall decrypt a concatenation of *C_IAM2* ($DA8_h$), *Purpose_IAM2*[3:0], *IRnd_IAM2*[31:0] and *TChallenge_IAM1* as input for the IAM2 Message or IAM3 Message.

NOTE Decryption is used here by the Interrogator because this allows the use of an encryption-only implementation on the Tag.

9.5.5 IAM2 Message

The IAM2 Message format is specified in Table 17 and has the following parameters:

— AuthMethod[1:0]: value "01_b" specifies the use for IAM;

— Step[1:0]: value "01_b" specifies the use of IAM2 or IAM3;

— CustomData[0:0]: value "0_b" specifies that no custom data included (IAM2);

— IAM2_RFU[2:0]: value "000_b" makes the total length of the IAM2 Message a multiple of 8-bits and might be used for future extensions of this document;

— IResponse[127:0]: bit string with decrypted concatenation of *C_IAM2*, *Purpose_IAM2*, *IRnd_IAM2* and *TChallenge_IAM1*.

C_IAM2 ($DA8_h$), *Purpose_IAM2*[3:0], *IRnd_IAM2*[31:0] are described in Table 1.

TChallenge_IAM1[79:0] is received in the IAM1 Response (see 9.5.4).

Table 17 — IAM2 Message format

	AuthMethod	Step	CustomData	IAM2_RFU	IResponse
# of bits	2	2	1	3	128
Description	01_b	01_b	0_b	000_b	AES-DEC(Key[KeyID].*ENC_key*, *C_IAM2*[11:0] || *Purpose_IAM2*[3:0] || *IRnd_IAM2*[31:0] || *TChallenge_IAM1*[79:0])

The Tag shall accept this message only in the **IAM-Init** state (unless occupied by internal processing and not capable of receiving messages). If the Tag is not in the **IAM-Init** state, it shall abort any cryptographic protocol that has not yet been completed and shall transition to the **Initial** state.

If the length of the IAM2 message is <> 136 bits, then the Tag shall return an "Other Error" error condition.

 © ISO/IEC 2017 – All rights reserved

If the value of IAM2_RFU[2:0] is <> "000_b", then the Tag shall return a "Not Supported" error condition.

If the parameter verifications have been completed successfully, the Tag shall perform an AES encryption of IResponse and retrieve *C_IAM2*[11:0], *Purpose_IAM2*[3:0], *IRnd_IAM2*[31:0] and *TChallenge_IAM1*[79:0]) for further verification.

NOTE The value for KeyID has been stored in IAM1 (see 9.5.3).

Cryptographic errors shall only be returned after all checks have been completed.

If the value of *C_IAM2*[11:0] is <> "$DA8_h$", then the Tag shall return a "Not Supported" error condition.

If the value of *Purpose_IAM2*[3:0] is <> "0000_b" and not supported by the Tag, then the Tag shall return a "Not Supported" error condition.

If the value for *TChallenge_IAM1*[79:0] is not equal to the copy of *TChallenge_IAM1*[79:0] that has been stored in IAM1 (see 9.5.3), then the Tag shall return a "Cryptographic Error" error condition.

9.5.6 IAM2 Response

If the Interrogator Authentication has been completed successfully, the Tag shall respond with an IAM2 Response that shall be empty (zero bits).

After returning IAM2 Response (TResponse), the Tag shall transition to the **IA-OK** state.

9.5.7 Final Interrogator processing IAM2

The reception of the IAM2 Response (with zero bits) is the acknowledgement for the Interrogator (or the external application controlling the Interrogator) that the Tag has transitioned to the **IA-OK** state.

9.5.8 IAM3 Message

9.5.8.1 General

The Interrogator shall use IAM3 if it wants to write custom data in the Tag's memory using Interrogator Authentication. The custom data is specified in 9.2 and may be protected (protection of integrity and authenticity of origin) and/or decrypted (confidentiality protection).

The IResponse in the IAM3 Message consists of three parts.

a) 128-bit *AuthenticationBlock*

   AES-DEC(Key[KeyID].*ENC_key*, *C_IAM3*[11:0] || *Purpose_IAM3*[3:0] || *IRnd_IAM3*[31:0] || *TChallenge_IAM1*[79:0])

   Purpose_IAM3[3:0] and *IRnd_IAM3*[31:0] are described in Table 1.

   According to ProtMode, *C_IAM3*[11:0] is the value C_IAM3_0[11:0], C_IAM3_1[11:0], C_IAM3_2[11:0] or C_IAM3_3[11:0] as defined in Table 1.

   If ProtMode = 0000_b, C_IAM3=C_IAM3_0.

   If ProtMode = 0001_b, C_IAM3=C_IAM3_1.

   If ProtMode = 0010_b, C_IAM3=C_IAM3_2.

   If ProtMode = 0011_b, C_IAM3=C_IAM3_3.

   TChallenge_IAM1[79:0] is received in the IAM1 Response (see 9.5.3)

   The Interrogator needs to use decryption because this allows the use of an encryption-only implementation on the Tag.

© ISO/IEC 2017 – All rights reserved

b) HEADER and the custom data block *CUSTOMDATA*(*D**128-*H*) consisting together of *D* data blocks

Depending on the value of ProtMode, the data block custom data can be in plaintext or in ciphertext.

If ciphertext is required, the Interrogator shall use AES in CBC decryption mode on the custom data, using the *AuthenticationBlock* as the Initialization Vector.

CIPHERDATA(*D**128) := CBC_{ENC}_AES_{INV} (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*))

c) Optionally, the 96-bit message authentication code block *MAC_Block*

The presence of this part depends on the value of ProtMode. The custom data block can be in plaintext or encrypted. If required, the interrogator shall use AES-CMAC-96 to protect the integrity of the message by calculating a message authentication code over the authentication block and the following *D* custom data blocks.

MAC_Block := AES-CMAC-96(Key[KeyID].MAC_key, *AuthenticationBlock* || HEADER || *CUSTOMDATA*/*CIPHERDATA*)

The length IResponse can either be (128 + *D**128) bits or (128 + *D**128 + 96) bits, depending on the presence of the message authentication code block.

The IAM3 Message format is specified in Table 18 and has the following parameters:

— AuthMethod[1:0]: value "01_b" specifies the use for IAM;

— Step[1:0]: value "01_b" specifies the use of IAM2 or IAM3;

— CustomData[0:0]: value "1_b" specifies that custom data is included in the Message (IAM3);

— BlockSize[0:0]: indicator that defines the size of the custom data block, "0_b" specifies custom data as 64-bit block, "1_b" specifies custom data as 16-bit block;

— IAM3_RFU[1:0]: value "00_b" makes the total length of the IAM3 Message a multiple of 8-bits and will be used for future extensions of this document;

— Profile[3:0]: pointer that selects a memory profile for the addition of custom data as specified in E.4.5;

— Offset[11:0]: value that specifies the number of multiples of 16-bit or 64-bit blocks that needs to be added to the address that is specified by Profile to define the first address of the custom data block;

— BlockCount[3:0]: number that defines the size of the custom data as a number of 16-bit or 64-bit blocks;

If the number of included bits of the header and custom data is not a multiple of 128, then padding with zeroes shall be applied to the least significant bits of the last block that has a non-zero block size of less than 128 bits. The Interrogator shall maintain the value of BlockCount for use as part of the MAC verification process. The Tag manufacturer shall specify the number of custom data blocks that can be included.

NOTE When combined with the values for BlockSize and BlockCount, as well as requiring that all component blocks are complete, the padding scheme proposed in this document is unambiguous.

— ProtMode[3:0]: value to select the mode of operation that shall be used to process the custom data as specified in Table 3;

— IResponse: bit string which content depends on the value of ProtMode and is defined as follows:

For ProtMode with value "0000_b", IResponse consists of (128 + *D**128) bits and contains:

AuthenticationBlock ||

HEADER(*H*)*CUSTOMDATA*(*D**128-*H*)

 © ISO/IEC 2017 – All rights reserved

For ProtMode with value "0001_b", IResponse consists of (128 + *D**128) bits and contains:

AuthenticationBlock ||

$CBC_{ENC}\_AES_{INV}$ (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, *HEADER*(*H*)||*CUSTOMDATA*(*D**128-*H*))

For ProtMode with value "0010_b", IResponse consists of (128 + *D**128 + 96) bits and contains:

AuthenticationBlock ||

HEADER(*H*) || *CUSTOMDATA*(*D**128) ||

AES-CMAC-96(Key[KeyID].*MAC_key*, *AuthenticationBlock* || *HEADER*(*H*) || *CUSTOMDATA*(*D**128-*H*))

For ProtMode with value "0011_b", IResponse consists of (128 + *D**128 + 96) bits and contains:

AuthenticationBlock ||

$CBC_{ENC}\_AES_{INV}$ (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, *HEADER*(*H*) || *CUSTOMDATA*(*D**128-*H*)) ||

AES-CMAC-96(Key[KeyID].*MAC_key*, *AuthenticationBlock* || $CBC_{ENC}\_AES_{INV}$ (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, *HEADER*(*H*) || *CUSTOMDATA*(*D**128-*H*)))

Table 18 — IAM3 Message format

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	128 + *D**128 or 128 + *D**128+96
Description	01_b	01_b	1_b	[0:0]	00_b	[3:0]	[11:0]	[3:0]	[3:0]	

The Tag shall accept this message only in the **IAM-Init** state (unless occupied by internal processing and not capable of receiving messages). If the Tag is not in the **IAM-Init** state, it shall abort any cryptographic protocol that has not yet been completed and shall transition to the **Initial** state.

The Tag shall verify the length of the IAM3 message.

If ProtMode is "0000_b" or "0001_b" and the length of the IAM3 message is <> (32 + 128 + *D**128) bits, then the Tag shall return an "Other Error" error condition.

If ProtMode is "0010_b" or "0011_b" and the length of the IAM3 message is <> (32 + 128 + *D**128 + 96) bits, then the Tag shall return an "Other Error" error condition.

If the ProtMode value is not supported by the Tag, then the Tag shall return a “Not Supported” error condition.

If the value of IAM3_RFU[1:0] is <> "00_b", then the Tag shall return a "Not Supported" error condition.

If the memory profile specified in Profile is not supported by the Tag, then the Tag shall return a "Not Supported" error condition.

The Tag shall check if the specified memory profile has the right to use KeyID for further processing:

If Profile = "0000_b" and key[KeyID].MPI[0:0] = "1_b" or

if Profile = "0001_b" and key[KeyID].MPI[1:1] = "1_b" or

if Profile = "0010_b" and key[KeyID].MPI[2:2] = "1_b" or

if Profile = "0011_b" and key[KeyID].MPI[3:3] = "1_b" or

if Profile = "0100_b" and key[KeyID].MPI[4:4] = "1_b"or

© ISO/IEC 2017 – All rights reserved

if Profile = "0101_b" and key[KeyID].MPI[5:5] = "1_b" or

if Profile = "0110_b" and key[KeyID].MPI[6:6] = "1_b" or

if Profile = "0111_b" and key[KeyID].MPI[7:7] = "1_b" or

if Profile = "1000_b" and key[KeyID].MPI[8:8] = "1_b" or

if Profile = "1001_b" and key[KeyID].MPI[9:9] = "1_b" or

if Profile = "1010_b" and key[KeyID].MPI[10:10] = "1_b" or

if Profile = "1011_b" and key[KeyID].MPI[11:11] = "1_b" or

if Profile = "1100_b" and key[KeyID].MPI[12:12] = "1_b" or

if Profile = "1101_b" and key[KeyID].MPI[13:13] = "1_b" or

if Profile = "1110_b" and key[KeyID].MPI[14:14] = "1_b" or

if Profile = "1111_b" and key[KeyID].MPI[15:15] = "1_b",

then key[KeyID] is authorized for this memory profile,

else key[KeyID] is not authorized for this memory profile and the Tag shall return a "Not Supported" error condition.

If the block of custom data specified by Profile, BlockSize, Offset and BlockCount is not supported by the Tag, then the Tag shall return a "Memory Overrun" error condition.

If the block of custom data specified by Profile, BlockSize, Offset and BlockCount is write locked, then the Tag shall return a "Memory Write Error" error condition.

In the following clauses, the length of IResponse will be referred to by the numerical designator *LoI*.

If the verifications have been completed successfully, the Tag shall perform an AES encryption of the authentication block in IResponse[*LoI-1*:*LoI-128*] and retrieve *C_IAM3*[11:0], *Purpose_IAM3*[3:0], *IRnd_IAM3*[31:0] and *TChallenge_IAM1*[79:0]) for further verification.

NOTE The value for KeyID has been stored in IAM1 (see 9.5.3).

Cryptographic errors should only be returned after all checks have been completed.

If ProtMode is "0000_b", the Tag shall check if the value of C_IAM3[11:0] is equal to C_IAM3_0. In case of mismatch, the Tag shall return a Cryptographic Error" error condition.

If ProtMode is "0001_b", the Tag shall check if the value of C_IAM3[11:0] is equal to C_IAM3_1. In case of mismatch, the Tag shall return a Cryptographic Error" error condition.

If ProtMode is "0010_b", the Tag shall check if the value of C_IAM3[11:0] is equal to C_IAM3_2. In case of mismatch, the Tag shall return a Cryptographic Error" error condition.

If ProtMode is "0011_b", the Tag shall check if the value of C_IAM3[11:0] is equal to C_IAM3_3. In case of mismatch, the Tag shall return a Cryptographic Error" error condition.

If the value of *Purpose_IAM3*[3:0] is <> "0000_b" and not supported by the Tag, then the Tag shall return a “Cryptographic Error” error condition.

If the value for *TChallenge_IAM1*[79:0] is not equal to the copy of *TChallenge_IAM1*[79:0] that has been stored in IAM1 (see 9.5.3), then the Tag shall return a “Cryptographic Error” error condition.

If all verifications have been completed successfully, the Tag shall further process IResponse based on the value of ProtMode.

 © ISO/IEC 2017 – All rights reserved

If ProtMode is "0000_b", the Tag shall process IResponse as described in 9.5.8.2.

If ProtMode is "0001_b", the Tag shall process IResponse as described in 9.5.8.3.

If ProtMode is "0010_b", the Tag shall process IResponse as described in 9.5.8.4.

If ProtMode is "0011_b", the Tag shall process IResponse as described in 9.5.8.5.

9.5.8.2 IResponse if ProtMode = "0000_b": Plaintext

The Interrogator has added custom data to the authentication block in plaintext as specified in Table 19.

Table 19 — IResponse if ProtMode = "0000_b": Plaintext

	IResponse
# of bits	128 + *D**128
Description	*AuthenticationBlock* || HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*)

The tag shall retrieve HEADER from the IResponse[*H*:0] and verify that it is valid. In case of mismatch, the Tag shall return a “Cryptographic Error” error condition.

The Tag shall retrieve the custom data from IResponse[(*D**128-H-1):0] and store it in *CUSTOMDATA*(*D**128-*H*).

9.5.8.3 IResponse if ProtMode = "0001_b": CBC encryption only

The Interrogator has added custom data with confidentiality protection to the authentication block as specified in Table 20.

Table 20 — IResponse if ProtMode = "0001_b": CBC encryption only

	IResponse
# of bits	128 + *D**128
Description	*AuthenticationBlock* || CBC_{ENC}_AES_{INV} (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*))

The Tag shall recover the header and the custom data by encrypting CBC_{ENC}_AES (IV= IResponse[(*LoI*-1):(*LoI*-128)], Key[KeyID].*ENC_key*, IResponse[(*D**128-1):0]).

Then the Tag shall retrieve HEADER from the previous result[*H*:0] and verify that it is valid. In case of mismatch, the Tag shall return a “Cryptographic Error” error condition.

Finally, the Tag should retrieve the customer data from the previous result [(*D**128-H-1):0] and store it in *CUSTOMDATA*(*D**128-*H*).

NOTE ProtMode = 0001_b offers only data encryption, no protection of the integrity of the custom data.

9.5.8.4 IResponse if ProtMode = "0010_b": CMAC message authentication only

The Interrogator has added custom data with integrity protection to the authentication block as specified in Table 21.

© ISO/IEC 2017 – All rights reserved

Table 21 — IResponse if ProtMode = "0010$_b$": CMAC message authentication only

	IResponse
# of bits	128 + *D**128 + 96
Description	*AuthenticationBlock* || HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*) || AES-CMAC-96(Key[KeyID].*MAC_key*, *AuthenticationBlock* || HEADER(*H*) *CUSTOMDATA*(*D**128-*H*))

The Tag shall use AES-CMAC-96(Key[KeyID].*MAC_key*, IResponse[(*LoI*-1):96]) to calculate the truncated 96-bit CMAC over *the authentication block* and the plaintext custom data HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*).

The Tag shall compare the result with IResponse[95:0] and return a “Cryptographic Error” error condition if the values are not identical.

The Tag shall retrieve the data from IResponse[(*D**128+95):96]. From this data, the Tag shall extract HEADER and verify it. In case of mismatch, the Tag shall return a “Cryptographic Error” error condition.

Finally, the Tag shall extract the custom data and store it in *CUSTOMDATA*(*D**128-*H*).

9.5.8.5 IResponse if ProtMode = "0011$_b$": CBC encryption with CMAC message authentication

The Interrogator has added custom data with confidentiality and integrity protection to the authentication block as specified in Table 22.

Table 22 — IResponse if ProtMode = "0011$_b$": CBC encryption with CMAC message authentication

	IResponse
# of bits	128 + *D**128 + 96
Description	*AuthenticationBlock* || CBC$_{ENC}$_AES$_{INV}$ (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*)) || AES-CMAC-96(Key[KeyID].*MAC_key*, *AuthenticationBlock* || CBC$_{ENC}$_AES$_{INV}$ (IV= *AuthenticationBlock*, Key[KeyID].*ENC_key*, HEADER(*H*) || *CUSTOMDATA*(*D**128)-*H*))

The Tag shall use AES-CMAC-96(Key[KeyID].*MAC_key*, IResponse[(*LoI*-1):96]) to calculate the truncated 96-bit CMAC over *the authentication block* and the encrypted data HEADER(*H*) || *CUSTOMDATA*(*D**128-*H*).

The Tag shall compare the result with IResponse[95:0] and return a “Cryptographic Error” error condition if the values are not identical.

The Tag shall recover the data by encrypting CBC$_{ENC}$_AES$_{INV}$ (IV= IResponse[(*LoI*-1): (*LoI*-128)], Key[KeyID].*ENC_key*, IResponse[(*D**128+95):96].

From the previous result, the Tag shall extract HEADER and *CUSTOMDATA*(*D**128-*H*). Then the Tag shall verify the HEADER. In case of mismatch, the Tag shall return a “Cryptographic Error” error condition.

If HEADER is valid, the Tag shall store the custom data in *CUSTOMDATA*(*D**128-*H*).

9.5.9 IAM3 Response

If the Interrogator Authentication and the verification of the custom data has been completed successfully, the Tag shall write the value *CUSTOMDATA*(*D**128-*H*), as specified by the parameters BlockSize, Profile, Offset and BlockCount, to the Tag’s memory.

If writing the custom data *CUSTOMDATA*(*D**128-*H*) to the specified memory area results in an error, then the Tag shall return the “Memory Write Error” error condition.

 © ISO/IEC 2017 – All rights reserved

The Tag shall respond with an IAM3 Response that shall be empty (zero bits).

After sending the IAM3 Response, the Tag shall transition to the **IA_OK** state.

9.5.10 Final Interrogator processing IAM3

The reception of the IAM3 Response (with zero bits) is the acknowledgement for the Interrogator (or the external application controlling the Interrogator) that the Tag has processed the custom data and has transitioned to the **IA-OK** state.

9.6 Mutual authentication (Method “10” = MAM)

9.6.1 General

The following subclauses describe the formatting of Message and Response for Mutual Authentication.

The MAM Messages are distinguished by the value of Step, the next 2-bit parameter in the Message after AuthMethod.

If Step = "00_b", the Tag shall parse the MAM1 Message as described in 9.6.2.

If Step = "01_b", the Tag shall parse the MAM2 Message as described in 9.6.5.

If Step = "10_b", the Tag shall return a "Not Supported" error condition.

If Step = "11_b", the Tag shall return a "Not Supported" error condition.

9.6.2 MAM1 Message

The Interrogator shall generate an 80-bit random number for use as IChallenge_MAM1. To initiate the mutual authentication, the Interrogator sends a request to get a challenge from the Tag.

The MAM1 Message format is specified in Table 23 and has the following parameters:

— AuthMethod[1:0]: value "10_b" specifies the use for MAM;

— Step[1:0]: value "00_b" specifies the use of MAM1;

— MAM1_RFU[3:0]: value "0000_b" makes the total length of the MAM1 Message a multiple of 8-bits and might be used for future extensions of this document;

— KeyID[7:0]: identifier of the key in Table 27;

— IChallenge_MAM1[79:0]: random challenge that the Interrogator has generated for use in MAM1.

Table 23 — MAM1 Message format

	AuthMethod	Step	MAM1_RFU	KeyID	IChallenge_MAM1
# of bits	2	2	4	8	80
Description	10_b	00_b	0000_b	[7:0]	random Interrogator challenge

The Tag shall accept this message only in the **Initial** or the **IA-OK** state (unless occupied by internal processing and not capable of receiving messages). If the parameters of the message are invalid, then the Tag shall transition to the **Initial** state, thereby aborting any cryptographic protocol that has not yet been completed.

If the length of the MAM1 message is <> 96 bits, then the Tag shall return an "Other Error" error condition.

If the value of MAM1_RFU[3:0] is <> "0000_b", then the Tag shall return a "Not Supported" error condition.

© ISO/IEC 2017 – All rights reserved

If the Tag does not support key[KeyID].*ENC_key*, then it shall return a "Not Supported" error condition.

9.6.3 MAM1 Response

The Tag shall store a copy of IChallenge_MAM1[31:0] for subsequent verification (see 9.6.5).

The Tag shall store a copy of KeyID for use in 9.6.5.

The Tag shall generate a random challenge *TChallenge_MAM1*[79:0] and store a copy of *TChallenge_MAM1*[79:0] for subsequent verification (see 9.6.5).

The Tag shall encrypt a concatenation of the constant *C_MAM1*(DA83$_h$), *TChallenge_MAM1*[31:0] and IChallenge_MAM1[79:0] using Key[KeyID].*ENC_key* and concatenate *TChallenge_MAM1*[79:32] to the result.

Table 24 — MAM1 Response format

	TResponse
# of bits	128 + 48
Description	AES-ENC(Key[KeyID].*ENC_key*, *C_MAM1*[15:0] // *TChallenge_MAM1*[31:0] || IChallenge_MAM1[79:0]) || *TChallenge_MAM1*[79:32]

After returning MAM1 Response (TResponse), the Tag shall transition to the **MAM-Init** state.

9.6.4 Final Interrogator processing MAM1

The Interrogator (or the external application controlling the Interrogator) decrypts the MAM1 Response (TResponse) and shall verify whether *C_MAM1* and IChallenge_MAM1 have the correct value. In that case, the Interrogator (or the external control application) can accept the Tag as genuine and it can store *TChallenge_MAM1* for further processing.

NOTE The Interrogator here uses decryption because this allows the use of an encryption-only implementation on the Tag.

9.6.5 MAM2 Message

The MAM2 Messages allows the Tag to authenticate the Interrogator after the Interrogator correctly replies to *TChallenge_MAM1*.

The MAM2 Message format is specified in Table 25 has the following parameters:

— AuthMethod[1:0]: value "10$_b$" specifies the use for MAM2;

— Step[1:0]: value "01$_b$" specifies the use of MAM2;

— MAM2_RFU[3:0]: value "0000$_b$" makes the total length of the MAM2 Message a multiple of 8-bits and will be used for future extensions of this document;

— IResponse[127:0]: bit string with decrypted concatenation of *C_MAM2*, *Purpose_MAM2*, IChallenge_MAM1 and *TChallenge_MAM1*;

  C_MAM2 (DA8$_h$) and *Purpose_MAM2*[3:0] are described in Table 1;

  IChallenge_MAM1[31:0] is generated by the Interrogator for use in MAM1 (see 9.6.2) *TChallenge_MAM1*[79:0] is generated by the Tag in the response for MAM1 (see 9.6.3).

© ISO/IEC 2017 – All rights reserved

Table 25 — MAM2 Message format

	Auth Method	Step	MAM2_ RFU	IResponse
# of bits	2	2	4	128
Description	10_b	01_b	0000_b	AES-DEC(Key[KeyID].*ENC_key*, *C_MAM2*[11:0] || *Purpose_MAM2*[3:0] || IChallenge_MAM1[31:0] || *TChallenge_MAM1*[79:0])

The Tag shall accept this message only in the **MAM-Init** state (unless occupied by internal processing and not capable of receiving messages). If the Tag is not in the **MAM-Init** state, it shall abort any cryptographic protocol that has not yet been completed and shall transition to the **Initial** state.

If the length of the MAM2 message is <> 136 bits, then the Tag shall return an "Other Error" error condition.

If the value of MAM2_RFU[2:0] is <> "000_b", then the Tag shall return a "Not Supported" error condition.

If the verification of MAM2_RFU is completed, the Tag shall encrypt the Interrogator message IResponse[127:0] to retrieve *C_MAM2*[11:0], *Purpose_MAM2*[3:0], IChallenge_MAM1[31:0] and *TChallenge_MAM1*[79:0]).

Cryptographic errors shall only be returned after all checks have been completed.

If the value of *C_MAM2*[11:0] is <> "$DA8_h$", then the Tag shall return a “Cryptographic Error” error condition.

If the value of *Purpose_MAM2*[3:0] is <> "0000_b" and not supported, then the Tag shall return a “Cryptographic Error” error condition.

If the value for IChallenge_MAM1[31:0] is not equal to the copy of IChallenge_MAM1[31:0] that has been stored in 9.6.3, then the Tag shall return a “Cryptographic Error” error condition.

If the value for *TChallenge_MAM1*[79:0] is not equal to the copy of *TChallenge_MAM1*[79:0] that has been stored in 9.6.3, then the Tag shall return a “Cryptographic Error” error condition.

9.6.6 MAM2 Response

If the Mutual Authentication has been completed successfully, the Tag shall respond with an MAM2 Response that shall be empty (zero bits).

After returning MAM2 Response (TResponse), the Tag shall transition to the **IA-OK** state.

9.6.7 Final Interrogator processing MAM2

The reception of the MAM2 Response (with zero bits) is the acknowledgement for the Interrogator (or the external application controlling the Interrogator) that the Tag has processed the MAM2 Message and has transitioned to the **IA-OK** state.

10 Communication

This document does not support secure communication.

11 Key Table and KeyUpdate

A Tag shall store one or more keys in the Key Table as specified in Table 27.

This document does not support KeyUpdate.

© ISO/IEC 2017 – All rights reserved

A key is identified by the KeyID, the identification number of the key within the Key Table. KeyID shall start with "00_h" and increment with one for every next key in the Key Table.

Each key shall contain an encryption key (*ENC_key*).

Each key may contain a message authentication key (*MAC_key*).

Encryption keys shall be exclusively used for Tag authentication, Interrogator authentication, Mutual authentication and encryption of custom data.

Message authentication keys shall be exclusively used for the authentication of custom data.

The Tag shall maintain a record in the Key Table for each key.

A record of the Key Management Table is specified in Table 27 and shall have the following parameters for every key:

— KeyID[7:0]: identifier of the key in Table 27;

— RFU_ENC[0:0]: reserved for future use;

— *ENC_key*[127:0]: Encryption Key that is used for Tag authentication, Interrogator authentication, Mutual authentication and for the confidentiality protection (encryption/decryption) of custom data.

A record of the Key Management Table may have the following parameters for every key:

— *MPI*[15:0]: Memory Profile Indicator.

Each key may be linked to a memory profile that is supported by the Tag. The links are stored in the MPI parameter. The MPI parameter contains 16 bits that correspond to a memory profile that is supported on the Tag and specifies if a security command of the air interface has the right to use that key to authenticate the Tag, encrypt the custom data and/or authenticate the custom data for the specified memory profile. MPI[0:0] to MPI[15:15] refers to memory profile 0 to 15, respectively, as far as they are supported by the Tag. If the value of an MPI bit is "0_b", this key shall not be used by the related profile. If the value of an MPI bit is "1_b", this key may be used by the related profile.

Table 26 describes the link of each bit of MPI with a memory profile.

Table 26 — Link of MPI bits with memory profiles

MPI bit	Function
MPI[0:0]	Memory profile 00 has the right to use this key
MPI[1:1]	Memory profile 01 has the right to use this key
MPI[2:2]	Memory profile 02 has the right to use this key
MPI[3:3]	Memory profile 03 has the right to use this key
....	....
MPI[14:14]	Memory profile 14 has the right to use this key
MPI[15:15]	Memory profile 15 has the right to use this key

The MPI bit or bits for non-existing profile on a Tag shall be permalocked to zero (bit "0_b") by the Tag manufacturer.

MPI is an optional parameter, but it shall be supported if the Tag supports the TAM2 mode (with custom data).

— RFU_MAC[0:0]: reserved for future use.

— *MAC_key*[127:0]: Message Authentication Key that may be used for the computation and validation of message authentication codes (MAC).

 © ISO/IEC 2017 – All rights reserved

It is recommended to generate the ENC_key and MAC_key independently to support separation of different cryptographic functions.

Table 27 — Key Management Table

KeyID[a]	**RFU_ENC**	**ENC_key**	**MPI** (optional)	**RFU_MAC**	**MAC_key** (optional)
00$_h$	[0:0]	key[127:0]	MPI[15:0]	[0:0]	key[127:0]
01$_h$	[0:0]	key[127:0]	MPI[15:0]	[0:0]	key[127:0]
02$_h$	[0:0]	key[127:0]	MPI[15:0]	[0:0]	key[127:0]
....	....	....		....	....
nn$_h$	[0:0]	key[127:0]	MPI[15:0]	[0:0]	key[127:0]

a See the definition of KeyID in Clause 3.

The size and initial values in Table 27 and its mapping to their respective physical memory locations on the Tag shall be defined by the manufacturer.

© ISO/IEC 2017 – All rights reserved

Annex A
(normative)

Crypto suite state transition table

Table A.1 — Crypto suite state transition table

Start state	Response	Action	Next state
Any	TAM1	Verify Tag's secret key	**Initial**
Any	TAM2	Verify Tag's secret key and request inclusion custom data	**Initial**
Initial	IAM1	Generate and return Tag challenge	**IAM-Init**
IA-OK	IAM1	Generate and return Tag challenge	**IAM-Init**
IAM-Init	IAM2	Verify Interrogator's secret key	**IA-OK**
IAM-Init	IAM3	Verify Interrogator's secret key and receive Custom Data	**IA-OK**
Initial	MAM1	Generate and return Tag challenge	**MAM-Init**
IA-OK	MAM1	Generate and return Tag challenge	**MAM-Init**
MAM-Init	MAM2	Verify Interrogator's secret key	**IA-OK**

Any combination of Start States and Transitions not listed in Table A.1 shall result in an error condition and consequently a transition to the **Initial** state.

All other errors resulting from the execution of commands shall result in an error and consequently a transition to the **Initial** state.

© ISO/IEC 2017 – All rights reserved

Annex B
(normative)

Error conditions and error handling

A Tag that encounters an error during the execution of a crypto suite operation might send an error reply to the Interrogator. The details of these error replies are defined in the respective air interface standards.

Table B.1 specifies a list of the error conditions that may result from the operation of this crypto suite. Annex E specifies how to translate this error condition into an error code for the air interface.

Table B.1 — Error conditions

Crypto suite error condition	Description
Cryptographic error	Cryptographic error detected.
Memory overrun	The command attempted to access a non-existent memory location.
Memory write error	An error occurred during writing the memory.
Not supported	The requested functionality is not supported by this crypto suite.
Other error	Miscellaneous error.

© ISO/IEC 2017 – All rights reserved

Annex C
(normative)

Cipher description

C.1 Description AES encryption algorithm

The Advanced Encryption Standard (AES) block cipher is described in detail in ISO/IEC 18033-3.

C.2 Terminology for cipher block chaining

C.2.1 General

The term encryption is used to transform a plaintext (something intelligible) into a ciphertext (something unintelligible). The mode of operation adopts the terms encryption and decryption because at this level, ciphertext messages and plaintext replies between Interrogator and Tag are being dealt with. Inside these modes of operation, a cryptographic engine with two functions is used, coined as forward cipher and inverse cipher.

Since FIPS 197 and NIST/SP 800-38A use the term "inverse cipher" function, this document uses the following terminology:

— **$\mathbf{CBC_{ENC}}$_AES** for the Tag processing in TAM2;

— **$\mathbf{CBC_{DEC}}$_$\mathbf{AES_{INV}}$** for the final Interrogator processing in TAM2;

— **$\mathbf{CBC_{ENC}}$_$\mathbf{AES_{INV}}$** for the Interrogator processing in IAM3;

— **$\mathbf{CBC_{DEC}}$_AES** for the final Tag processing in IAM3.

 © ISO/IEC 2017 – All rights reserved

C.2.2 CBC for TAM2 rev "0"

The Tag uses an ENcryption scheme (as it starts with the XOR) and uses the AES in encryption mode.

The Interrogator uses a DEcryption scheme (as it finishes with the XOR) and uses the AES INV in decryption mode.

Figure C.1 shows the CBC operation for TAM2 rev "0".

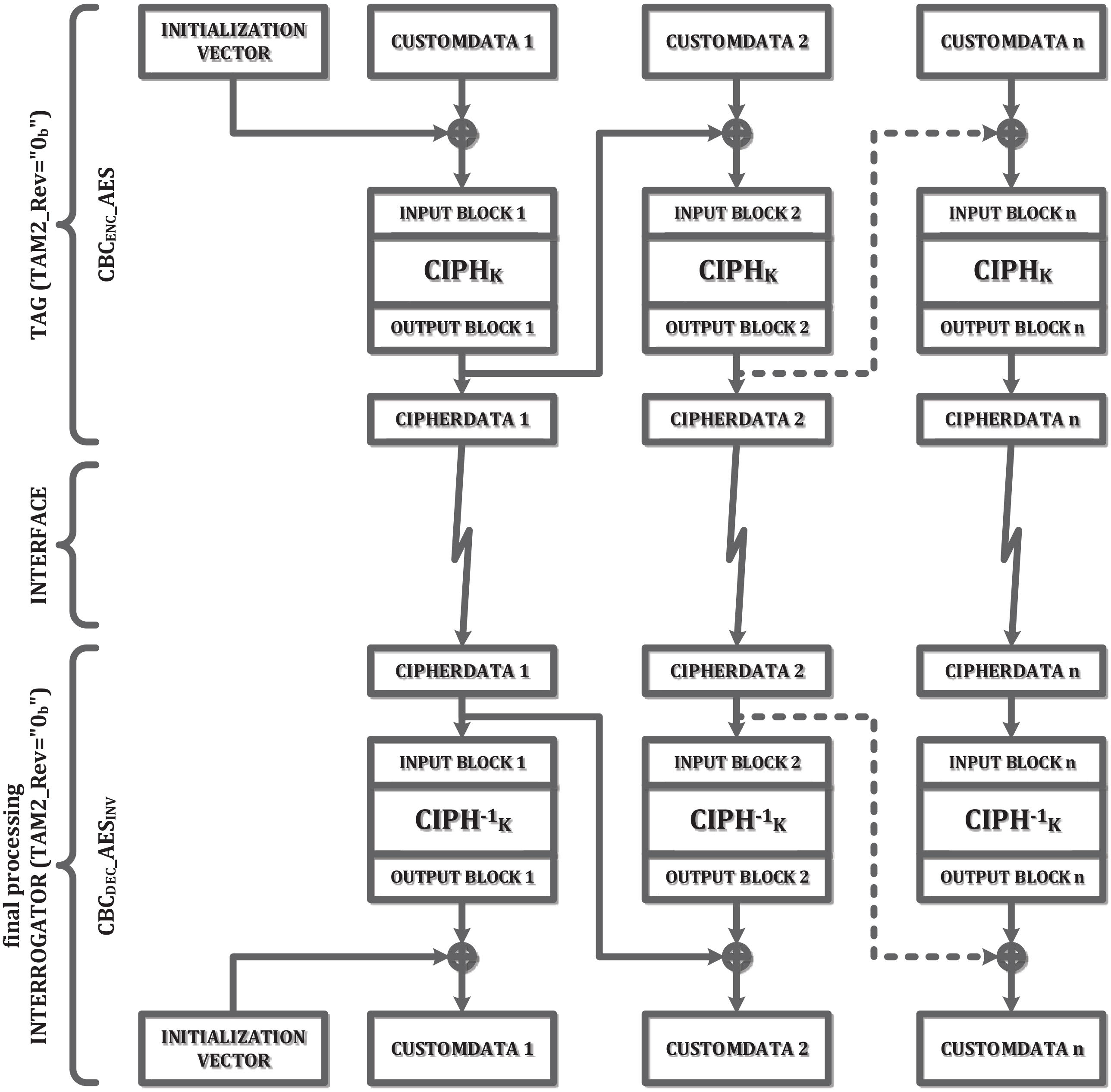


Figure C.1 — CBC operation for TAM2 rev "0"

C.2.3 CBC for TAM2 rev "1"

The Tag uses an ENcryption scheme (as it starts with the XOR) and uses the AES in encryption mode.

The Interrogator uses a DEcryption scheme (as it finishes with the XOR) and uses the AES INV in decryption mode.

Figure C.2 shows the CBC operation for TAM2 rev "1".

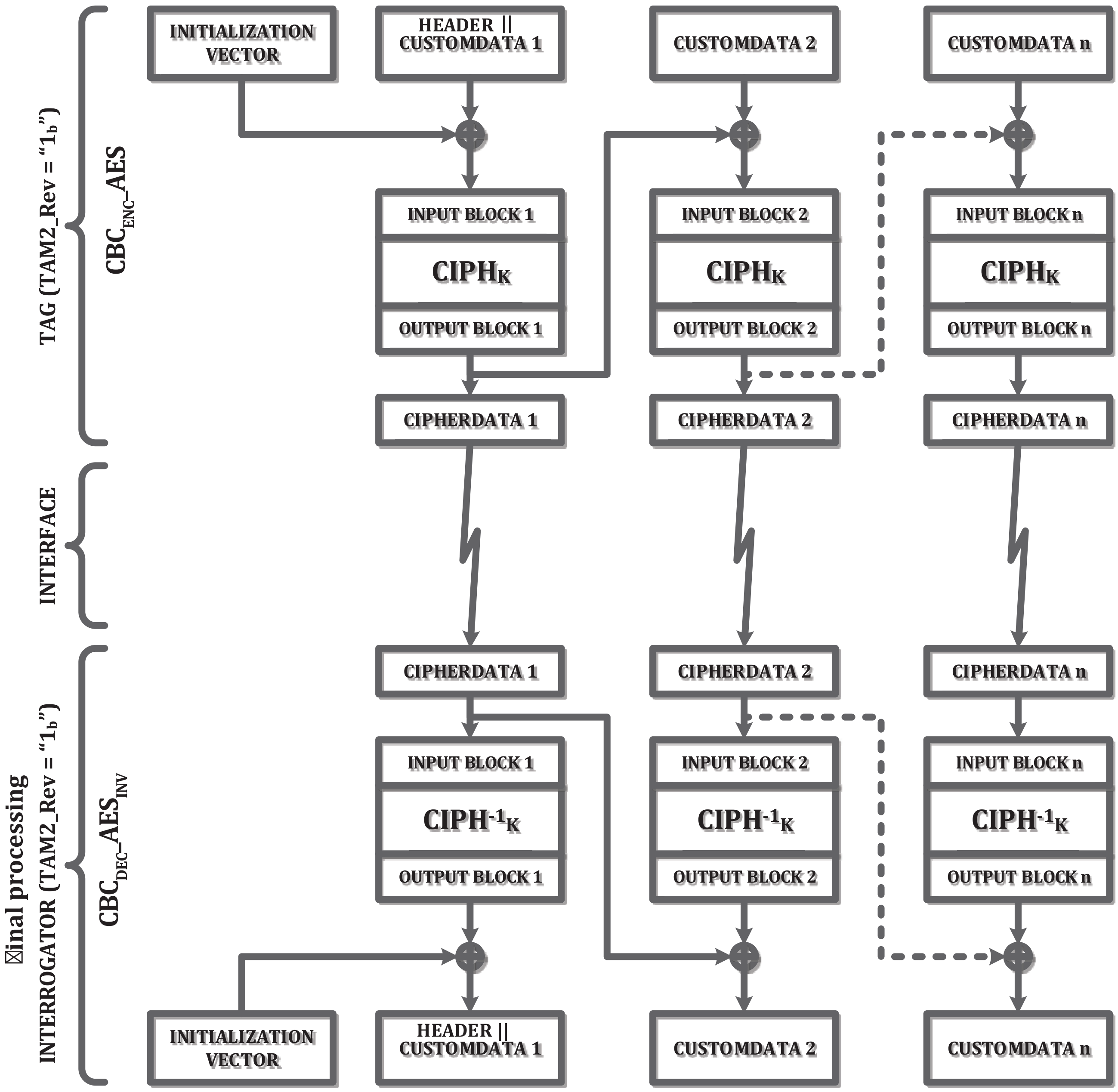


Figure C.2 — CBC operation for TAM2 rev "1"

© ISO/IEC 2017 – All rights reserved

C.2.4 CBC for IAM3

The Interrogator uses an ENcryption scheme (as it starts with the XOR) and uses the AES INV in decryption mode.

The Tag uses a DEcryption scheme (as it finishes with the XOR) and uses the AES in encryption mode.

Figure C.3 shows the CBC operation for IAM3.

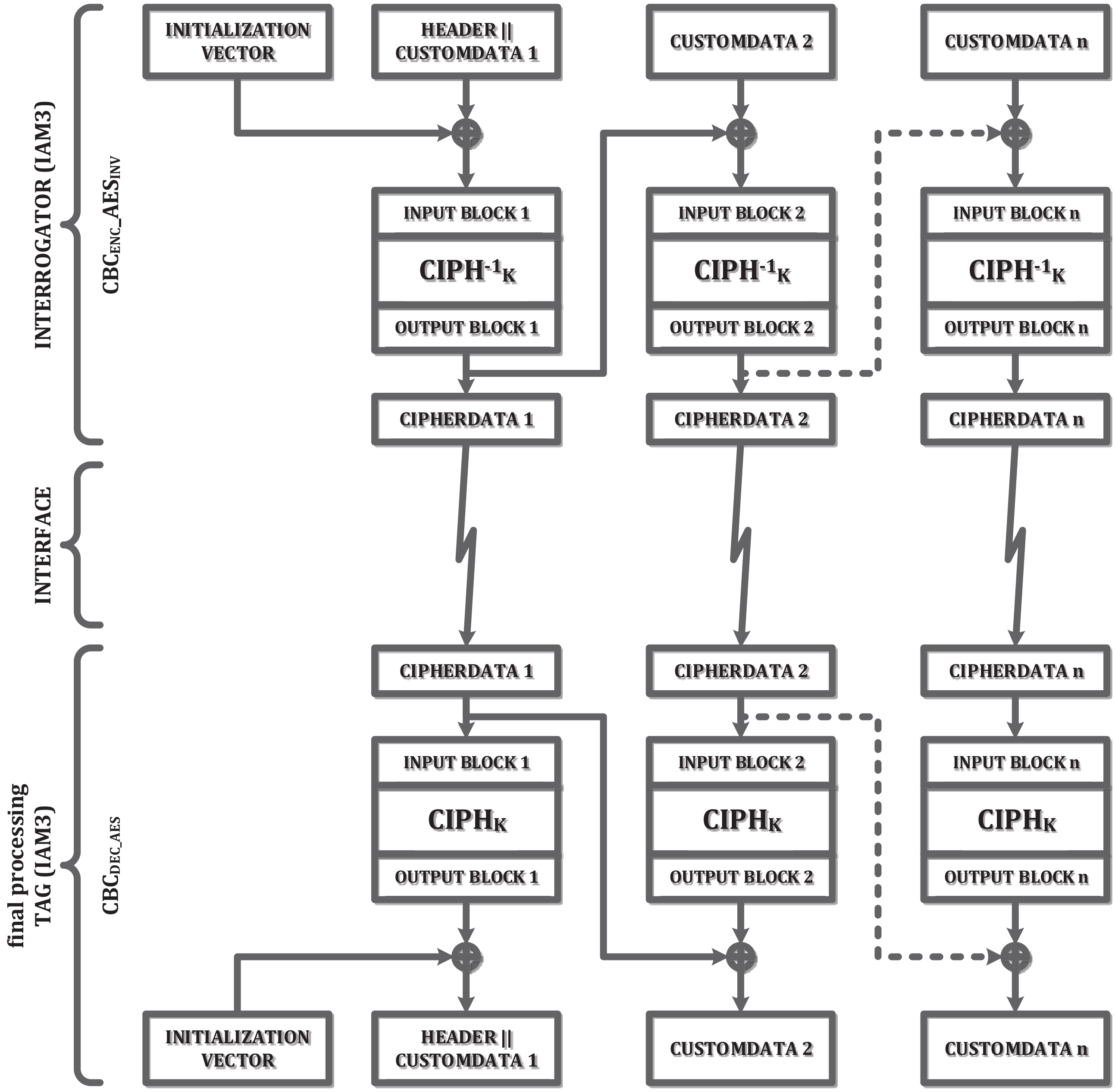


Figure C.3 — CBC operation for IAM3

© ISO/IEC 2017 – All rights reserved

Annex D
(informative)

Test vectors

D.1 References for AES test vectors

D.1.1 Test vectors for the AES algorithm

Test vectors for the AES algorithm can be found in http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

Additionally, the original submission to NIST included test vectors as well:

http://csrc.nist.gov/archive/aes/rijndael/rijndael-vals.zip

D.1.2 Online AES calculator

An online AES calculator can be found at http://testprotect.com/appendix/AEScalc.

© ISO/IEC 2017 – All rights reserved

Annex E
(normative)

Protocol specific information

E.1 General

E.1.1 Concept of exchanging Message and Response

For the implementation of this crypto suite, an air interface protocol shall support security commands that allow the exchange of data between the Interrogator and the Tag that has this crypto suite implemented. The security command contains a message with parameters for the crypto suite. The reply of the Tag contains a response with the data that is returned by the crypto suite. An example of such data exchange for this crypto suite is depicted in Figure E.1.

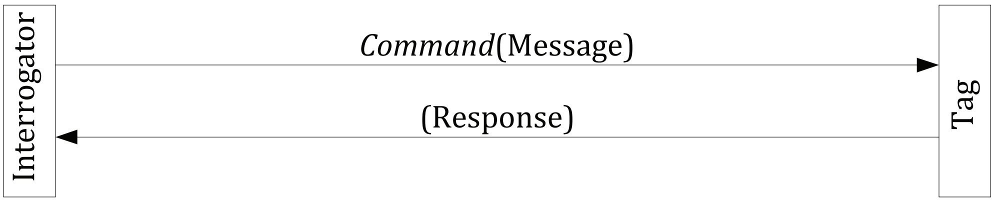


Figure E.1 — Message exchange for authentication process

The crypto suites that are defined by ISO/IEC 29167 can be defined by their Crypto Suite Identifier (CSI). According to ISO/IEC 29167-1, the CSI for this crypto suite shall be defined as the 6-bit value 000000_2. For use by the air interface protocols in this annex, the value is expanded to the 8-bit value 00_h.

This crypto suite is designed to provide security services for ISO/IEC 18000-3, Mode 1, ISO/IEC 18000-3, Mode 3 and ISO/IEC 18000-63. Details of the specific implementation for these air interface protocols are described in E.2, E.2.2 and E.4, respectively.

E.1.2 Supported security services

Table E.1 shows the security services that are supported by this crypto suite.

Table E.1 — Security services

Security services	Method	Mandatory, optional, prohibited, or not supported[a]
Authentication		Mandatory
Tag authentication (TA)	without custom data	Mandatory
Tag authentication (TA)	with custom data	Optional
Interrogator authentication (IAM)	All methods	Optional
Mutual Authentication (MAM)	All methods	Optional
Communication		Not supported
Authenticated Tag from TA	Authenticated communication (Tag => Interrogator)	Not supported

[a] A crypto suite shall identify for each security service above and method if it is mandatory, optional or prohibited.

© ISO/IEC 2017 – All rights reserved

Table E.1 *(continued)*

Security services	Method	Mandatory, optional, prohibited, or not supported[a]
	Secure authenticated communication (Tag => Interrogator)	Not supported
Authenticated Interrogator from IA	Authenticated communication (Interrogator => Tag)	Not supported
	Secure authenticated communication (Interrogator => Tag)	Not supported
[a] A crypto suite shall identify for each security service above and method if it is mandatory, optional or prohibited.		

E.2 Security services for ISO/IEC 18000-3, Mode 1

E.2.1 General

This subclause describes the implementation details for ISO/IEC 18000-3, Mode 1.

E.2.2 ISO/IEC 18000-3, Mode 1 protocol commands

A crypto suite supporting ISO/IEC 18000-3, Mode 1 shall fulfil the protocol security command requirements as defined in this subclause.

a) In accordance with the air interface standard, the Tag shall use the In-Process reply if the maximum execution time for an Authenticate command exceeds t1, as defined for the immediate reply.

NOTE This keeps the Interrogator informed about the ongoing processing on the Tag.

b) The Tag shall ignore commands from an Interrogator during execution of a cryptographic operation.

c) The Tag shall support sending the contents of the ResponseBuffer in the reply to a ReadBuffer command if a ResponseBuffer is supported by the Tag.

d) The Tag may support a security timeout following a crypto error. The length of the security timeout shall be <200 ms.

e) The Authenticate command shall be supported for all supported authentication methods.

f) The Challenge command may be supported for parts or all supported authentication methods.

g) A Tag in any cryptographic state shall ignore an invalid command and stay in the current state. (Invalid commands means crypto commands with non-matching UID or CRC error.)

h) For each error condition defined in the crypto suite,

— the Tag shall transition to the **Ready** state;

— the Tag shall send an error code in case of a transition to the **Ready** state;

— the Tag shall behave according to the error handling defined in ISO 18000-3, Mode 1;

— if the Tag is in **Selected Secure** state, it shall transition to the **Ready** state.

i) The Tag shall remain in its current state after a Tag Authentication. The Tag shall transition to **Selected Secure** state (corresponding to the **IA-OK state**) after processing successfully an Interrogator or Mutual Authentication.

j) This crypto suite does not support any encapsulation method.

k) This crypto suite does not support the KeyUpdate command.

© ISO/IEC 2017 – All rights reserved

E.2.3 Security commands in ISO/IEC 18000-3, Mode 1

In ISO/IEC 18000-3, Mode 1, the message to execute Tag authentication shall be transmitted to the Tag with the *Authenticate* or the *Challenge* command. The message to execute Interrogator Authentication or Mutual Authentication shall be transmitted to the Tag with the *Authenticate* command. The air interface shall return the response; it shall be backscattered immediately after the command and/or it shall be stored in the ResponseBuffer, from where it shall be returned to the Interrogator with the *ReadBuffer* command.

NOTE Information about the *Authenticate*, *Challenge* and *ReadBuffer* command and the ResponseBuffer for use in ISO/IEC 18000-3, Mode 1 can also be found in ISO/IEC 15693-3:2009/AMD4:2017.

ISO/IEC 18000-3, Mode 1 specifies an 8-bit CSI. For implementation of this document in ISO/IEC 18000-3, Mode 1, the CSI shall be expanded to the 8-bit value 00_h.

E.2.4 Implementation of crypto suite error conditions in ISO/IEC 18000-3 Mode 1

This document specifies error conditions when the authentication is not successful. The crypto suite shall return the error conditions for the error handling described in the base standard. Table E.2 shows the conversion of error conditions in the crypto suite to ISO/IEC 18000-3, Mode 1 error codes.

Table E.2 — Implementation of crypto suite error conditions as Tag error codes

Crypto suite error condition	Description	ISO/IEC 18000-3 Mode 1 error code	ISO/IEC 18000-3 Mode 1 error code name
Cryptographic error	Cryptographic error detected. This resets the cryptographic engine which results in a transition to the initial state	40_h	Generic cryptographic error
Memory overrun	The command attempted to access a non-existent memory location	10_h	The specified block is not available (does not exist)
Not supported	The requested functionality is not supported by this crypto suite	01_h	The command is not supported, i.e. the request code is not recognized
Other error	Miscellaneous error	$0F_h$	Error with no information given or a specific error code is not supported

E.2.5 Byte and bit order

The byte and bit order in ISO 18000-3, Mode 1 is defined as LSByte, LSbit first.

Transmission sequence of the payload:

Every payload parameter shall be transmitted LSBit and LSByte first.

The order of the payload parameters shall be transmitted in the sequence as defined in this document.

Bit field parameters shall be concatenated to achieve integer multiples of 8 bits.

Figure E.2 shows an example of the byte and bit order for the TAM2 message.

© ISO/IEC 2017 – All rights reserved

	Auth Method	Custom Data	Block Size	TAM2_ RFU	Key ID	IChallenge _TAM2	Pro file	Off set	Block Count	Prot Mode
# of bits	2	1	1	4	8	80	4	12	4	4
Description	00_b	1_b	[0:0]	0000_b	[7:0]	random Interrogator challenge	[3:0]	[11:0]	[3:0]	[3:0]

1 byte

Bit Field Parameters
[0] AuthMethode[0]
[1] AuthMethode[1]
[2] CustomData
[3] BlockSize]
[4] TAM1_RFU[0]
[5] TAM1_RFU[1]
[6] TAM1_RFU[2]
[7] TAM1_RFU[3]

10 bytes

Multiple Byte Parameter
Transmitted LSByte & LSBit first as defined in ISO 15693

2 bytes

Bit Field Parameters

Byte 0:
[0] Profile[0]
[1] Profile[1]
[2] Profile[2]
[3] Profile[3]
[4] Offset[0]
[5] Offset[1]
[6] Offset[2]
[7] Offset[3]

Byte 1:
[0] Offset[4]
[1] Offset[5]
[2] Offset[6]
[3] Offset[7]
[4] Offset[8]
[5] Offset[9]
[6] Offset[10]
[7] Offset[11]

Figure E.2 — Example byte and bit order for TAM2 message

Figure E.3 shows an example of the byte and bit order for the MAM1 response.

	TResponse
# of bits	128 + 48
description	AES-ENC(Key[KeyID].*ENC_key*, *C_MAM1*[15:0] || *TChallenge_MAM1*[31:0] || IChallenge_MAM1[79:0]) || *TChallenge_MAM1*[79:32]

22 bytes cipher text

Tresponse[175:0] Is transmitted as ONE parameter; LSByte and LSBit first

- TResponse[0]
- ...
- TResponse[47]
- TResponse[48]
- ...
- TResponse[175]

- TChallenge_MAM1[32]
- ...
- TChallenge_MAM1[79]
- Ciphertext[0]
- ...
- Ciphertext[127]

Figure E.3 — Example byte and bit order for MAM1 response

Figure E.4 shows an example of the byte and bit order for the MAM2 message.

 © ISO/IEC 2017 – All rights reserved

	Auth Method	Step	MAM2_ RFU	IResponse
# of bits	2	2	4	128
description	10_b	01_b	0000_b	AES-DEC(, Key[KeyID].*ENC_key*, *C_MAM2*[11:0] || *Purpose_MAM2*[3:0] || IChallenge_MAM1[31:0] || *TChallenge_MAM1*[79:0])

16 bytes

Ciphertext

AES calculation is done as defined in Crypto Suite:

- C_MAM2[11:0]
- Purpose_MAM2[3:0]
- IChallenge_MAM1[31:0]
- TChallenge_MAM1[79:0]

Ciphertext [127:0]

- transmitted in the following order:
- Ciphertext [0]

...

- Ciphertext [127]

Figure E.4 — Example byte and bit order for the ciphertext in MAM2 message

E.2.6 Key properties

ISO/IEC 18000-3, Mode 1 does not support the definition of key properties.

E.2.7 Memory profiles

Table E.3 shall contain zero or more pointers to an area with custom data within the Tag's memory. The maximum number of pointers is 16.

Table E.3 — Description of ISO/IEC 18000-3, Mode 1 specific memory profiles for Profile

Value	Description
0000	globally defined memory profile: UID (64 bit)
0001	globally defined memory profile: user memory (starting at word position 0)
0010	manufacturer-defined memory profile
0011	manufacturer-defined memory profile
....	....
1000	manufacturer-defined memory profile
1001	reserved for future use
....	....
1111	reserved for future use

The memory profiles specified by the Profile parameter may allow the use of one or more of the encryption algorithms and/or message authentication algorithms listed in Table 3 for the protection of the custom data. The chip manufacturer shall define which modes a particular Tag model supports for which memory profiles.

© ISO/IEC 2017 – All rights reserved

E.3 Security services for ISO/IEC 18000-3, Mode 3

Reserved for the implementation for ISO/IEC 18000-3, Mode 3.

E.4 Security services for ISO/IEC 18000-63

E.4.1 ISO/IEC 18000-63 protocol commands

A crypto suite supporting ISO/IEC 18000-63 shall fulfil the protocol security command requirements as defined in this subclause.

Optional choices shall be accepted for one-to-one communication. Reason: Since the Tag is singulated and the TID is known, supported options can be derived from it.

a) The Tag shall use the In-Process reply if the maximum execution time for an *Authenticate* command exceeds 20 ms.

b) The Tag shall ignore commands from an Interrogator during execution of a cryptographic operation.

c) The Tag may support sending the contents of the ResponseBuffer in the reply to an ACK command.

d) The Tag shall support sending the contents of the ResponseBuffer in the reply to a *ReadBuffer* command.

e) The Tag may support a security timeout following a crypto error. The length of the security timeout shall be <200 ms.

f) The *Authenticate* command shall be supported for all supported authentication methods.

g) The *Challenge* command may be supported for parts or all supported authentication methods.

h) A Tag in any cryptographic state other than the **initial** state (i.e. state after power-up) shall reset its cryptographic engine and transition to the open state upon receiving an invalid command. (Invalid commands means crypto commands with incorrect handle or CRC error.)

i) For each error condition defined in the crypto suite,

   — the Tag shall transition to the arbitrate state;

   — the Tag shall send an error code in case of a transition to the arbitrate state.

j) The Tag shall remain in its current state after a Tag Authentication. The Tag shall transition to the **secured** state after processing successfully an interrogator or mutual authentication.

k) This crypto suite does not support any encapsulation method.

l) This crypto suite does not support the KeyUpdate command.

E.4.2 Security commands in ISO/IEC 18000-63

In ISO/IEC 18000-63, the message to execute any authentication shall be transmitted to the Tag with the *Authenticate* or the *Challenge* command. The air interface shall return the response, either it shall be backscattered immediately after the command or it shall be stored in the ResponseBuffer, from where it shall be returned to the Interrogator with the *ReadBuffer* command.

NOTE Information about the *Authenticate*, *Challenge* and *ReadBuffer* command and the ResponseBuffer can also be found in ISO/IEC 19762.

ISO/IEC 18000-63 specifies an 8-bit CSI. For implementation of this document in ISO/IEC 18000-63, the CSI shall be expanded to the 8-bit value 00_h.

© ISO/IEC 2017 – All rights reserved

E.4.3 Implementation of crypto suite error conditions in ISO/IEC 18000-63

This document specifies error conditions when the authentication is not successful. The error conditions of the crypto suite shall be returned to the Interrogator as error codes for the air interface. Table E.4 shows the conversion of error conditions in the crypto suite to ISO/IEC 18000-63 error codes.

Table E.4 — Implementation of crypto suite error conditions as Tag error codes

Crypto suite error condition	Description	ISO/IEC 18000-63 error code	ISO/IEC 18000-63 error code name
Cryptographic error	Cryptographic error detected. This triggers a reset	00000101_b	Crypto suite error
Memory overrun	The command attempted to access a non-existent memory location	00000011_b	Memory overrun
Not supported	The requested functionality is not supported by this crypto suite	00000001_b	Not supported
Other error	Miscellaneous error	00000000_b	Other error

E.4.4 Key properties

ISO/IEC 18000-63 requires the definition of key properties. If an implementation does provide key properties for a key belonging to this crypto suite, it shall set the key properties to 0000_b.

E.4.5 Memory profiles

Table E.5 shall contain zero or more pointers to an area with custom data within the Tag's memory. The maximum number of pointers is 16.

Table E.5 — Description of ISO/IEC 18000-63 specific memory profiles for Profile

Value	Description
0000	globally defined memory profile: UII memory bank (starting at word position 0)
0001	globally defined memory profile: TID memory bank (starting at word position 0)
0010	globally defined memory profile: USER memory bank requested (file 0) (starting at word position 0)
0011	manufacturer-defined memory profile
....	....
1000	manufacturer-defined memory profile
1001	reserved for future use
....	....
1111	reserved for future use

The memory profiles specified by the Profile parameter may allow the use of one or more of the encryption algorithms and/or message authentication algorithms listed in Table 3 for the protection of the custom data. The chip manufacturer shall define which modes a particular Tag model supports for which memory profiles.

E.4.6 Interaction with untraceability feature

When a ProtMode is selected that specifies time-variant data encryption (ProtMode "0001_b" and "0011_b"), the Tag may assert the Untraceable privilege for the duration of the execution of the command when ProtMode specifies data encryption ("0001_b" and "0011_b"). Asserting the Untraceable privilege enables reading or writing of portions of UII, TID and User memory that are untraceably hidden and

© ISO/IEC 2017 – All rights reserved

enables reading or writing of User-memory files for which it has read or write privileges if User memory is untraceably hidden.

E.4.7 Use of purpose flags

The interrogator and mutual authentication result in the Tag being in the Secured state with an authenticated Interrogator. As this crypto suite does not support *AuthComm* and *SecureComm*, no further access commands are possible other than *Authenticate*. Consequently, the only use for interrogator and mutual authentication is to provide a standardized way to invoke manufacturer defined functionality identified by the Purpose flags. The only exception is IAM3 which also performs a write operation to Tag memory as part of the interrogator authentication.

© ISO/IEC 2017 – All rights reserved

Annex F
(informative)

Examples

F.1 General

This annex provides examples of messages and responses for the implementation of the TAM1, TAM2, MAM1 and MAM2 as they are specified in this document.

F.2 Assumed content of the Tag's Key Management Table and memory profile

F.2.1 Content of Key Management Table

Table F.1 shows the content of the key management table that is assumed to be present for the examples in this annex.

Table F.1 — Content of Key Management Table

KeyID	RFU	ENC_key	MPI	RFU	MAC_key
00_h	1_b	0001020304050607_h || $08090A0B0C0D0E0F_h$	0002_h	1_b	$2B7E151628AED2A6_h$ || $ABF7158809CF4F3C_h$
01_h	1_b	$0A1B2C3D4E5F6A7B_h$ || $8C9D0E1F2AB3C4D5_h$	0007_h	1_b	$889900AABBCCDDEE_h$ || $FFAABBCCDDEEFF00_h$

F.2.2 Memory profile configuration

Table F.2 shows the Tag's memory profile that is assumed to be configured for the examples in this annex.

Table F.2 — Configured Memory Profile = 0001_b

Bit Address	**00_h**	**10_h**	**20_h**	**30_h**	**40_h**	**50_h**	**60_h**	**70_h**	**80_h**
Content	$A16A_h$	7665_h	$6E69_h$	$617A_h$	5265_h	$676E_h$	$756D_h$	5475_h	$756D_h$

F.3 Example for TAM1

This subclause shows an example for TAM1.

Table F.3 shows an example for a TAM1 message using the first key of the Key Management Table, see F.2.1.

Table F.3 — Example TAM1 Message

	AuthMethod	CustomData	TAM1_RFU	KeyID	IChallenge_TAM1
# of bits	2	1	5	8	80
Description	00_b	0_b	00000_b	00_h	$96564402375796C69664_h$

Table F.4 shows the TAM1 response with *TRnd_TAM1* = $6D696372_h$.

© ISO/IEC 2017 – All rights reserved

Table F.4 — TAM1 Response

	TResponse
# of bits	128
Description	E920530CC781B20CFE1AB4A0144E7335$_h$[a]

[a] "E9" is the MSB and "35" is the LSB.

F.4 Examples for TAM2

F.4.1 Use of TAM2 Rev0

TAM2 is used to add custom data to the Tag authentication procedure. This subclause shows three different examples of message and response exchanges for TAM2 Rev0.

F.4.1.1 Example 1 for TAM2 Rev0

Table F.5 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero with two 64-bit blocks of custom data using CBC mode for data encryption.

Table F.5 — TAM2 Message Example 1 for TAM2 Rev0

	Auth Meth-od	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Descrip-tion	00$_b$	1$_b$	0$_b$	0$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0001$_b$	0001$_b$

Table F.6 shows the TAM2 response with TRnd_TAM2 = 72666964$_h$.

Table F.6 — TAM2 Response — Example 1 for TAM2 Rev0

	TResponse
# of bits	256
Description	C67D0EF2B1BA176DF9C0BD226212F142$_h$ || 46E79BF351E3D2A3CB50BEF0F1A917B3$_h$[a]

[a] "C6" is the MSB and "B3" is the LSB.

F.4.1.2 Example 2 for TAM2 Rev0

Table F.7 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero with one 64-bit block of custom data using CMAC mode for message authentication.

Table F.7 — TAM2 Message — Example 2 for TAM2 Rev0

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Descrip-tion	00$_b$	1$_b$	0$_b$	0$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0000$_b$	0010$_b$

Table F.8 shows the TAM2 response with TRnd_TAM2 = 72666964$_h$.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is FBEED61835713366_7C85E08F7236A8DE$_h$.

© ISO/IEC 2017 – All rights reserved

Table F.8 — TAM2 Response — Example 2 for TAM2 Rev0

	TResponse
# of bits	256+96
Description	C67D0EF2B1BA176DF9C0BD226212F142$_h$ || A16A76656E69617A0000000000000000$_h$ || 0DD2C5DEEC5EA202C41DFEE8$_h$[a]

[a] "C6" is the MSB and "E8" is the LSB.

F.4.1.3 Example 3 for TAM2 Rev0

Table F.9 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero and two 64-bit blocks of custom data using CBC mode for encryption plus CMAC mode for message authentication.

Table F.9 — TAM2 Message — Example 3 for TAM2 Rev0

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00$_b$	1$_b$	0$_b$	0$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0001$_b$	0011$_b$

Table F.10 shows the TAM2 response with TRnd_TAM2 = 72666964$_h$.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is FBEED61835713366_7C85E08F7236A8DE$_h$.

Table F.10 — TAM2 Response — Example 3 for TAM2 Rev0

	TResponse
# of bits	256+96
Description	C67D0EF2B1BA176DF9C0BD226212F142$_h$ || 46E79BF351E3D2A3CB50BEF0F1A917B3$_h$ || 1FD222AEDA9F7FF0DEB86509$_h$[a]

[a] "C6" is the MSB and "09" is the LSB.

F.4.2 Use of TAM2 Rev1

TAM2 is used to add custom data to the Tag authentication procedure. This subclause shows different examples of message and response exchanges for TAM2 with *TRnd_TAM2* = 72666964$_h$.

F.4.2.1 Example 1 for TAM2 Rev1

Table F.11 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero with two 64-bit blocks of custom data using CBC mode for encryption.

Table F.11 — TAM2 Message — Example 1 for TAM2 Rev1

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00$_b$	1$_b$	0$_b$	1$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0001$_b$	0001$_b$

Table F.12 shows the TAM2 response with *TRnd_TAM2* = 72666964$_h$. The Header is 0800080000000000$_h$

© ISO/IEC 2017 – All rights reserved

Table F.12 — TAM2 Response — Example 1 for TAM2 Rev1

	TResponse
# of bits	384
Description	B06628168448AF275D67756D4D17A0EE$_h$ || 33D7DC1DE1F6CF25CD92AAB5AF137091$_h$ || 08205C9947BA986D62D98A795D43B539$_h$[a]

[a] "B0" is the MSB and "39" is the LSB.

F.4.2.2 Example 2 for TAM2 Rev1

Table F.13 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero with one 64-bit block of custom data using CMAC mode for message authentication.

Table F.13 — TAM2 Message — Example 2 for TAM2 Rev1

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00$_b$	1$_b$	0$_b$	1$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0000$_b$	0010$_b$

Table F.14 shows the TAM2 response with *TRnd_TAM2* = 72666964$_h$. The Header is 0800000000000000$_h$

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is FBEED61835713366_7C85E08F7236A8DE$_h$.

Table F.14 — TAM2 Response — Example 2 for TAM2 Rev1

	TResponse
# of bits	256+96
Description	27CA8EFD714DE5C0B3F7C62619D4E204$_h$ || 0800000000000000A16A76656E69617A$_h$ || 0DFF99A6D097FF9028E0FEA4$_h$[a]

[a] "27" is the MSB and "A4" is the LSB.

F.4.2.3 Example 3 for TAM2 Rev1

Table F.15 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero and two 64-bit blocks of custom data using CBC mode for encryption plus CMAC mode for message authentication.

Table F.15 — TAM2 Message — Example 3 for TAM2 Rev1

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00$_b$	1$_b$	0$_b$	1$_b$	000$_b$	00$_h$	9656440237 5796C69664$_h$	0001$_b$	000$_h$	0001$_b$	0011$_b$

Table F.16 shows the TAM2 response with *TRnd_TAM2* = 72666964$_h$. The Header is 0800080000000000$_h$

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is FBEED61835713366_7C85E08F7236A8DE$_h$.

© ISO/IEC 2017 – All rights reserved

Table F.16 — TAM2 Response — Example 3 for TAM2 Rev1

	TResponse
# of bits	384+96
Description	$67039773F2C3C8E8B1D29254BADEF7C9_h$ || $B86966552DEE705111D4C077009D777C_h$ || $6D725BCD8DC8FA5C83960148241958C2_h$ || $E3546DEA93F24EE9B9799A43_h$[a]

[a] "67" is the MSB and "43" is the LSB.

F.4.2.4 Example 4 for TAM2 Rev1

Table F.17 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero and five 16-bit blocks of custom data using CBC mode for encryption plus CMAC mode for message authentication.

Table F.17 — TAM2 Message — Example 4 for TAM2 Rev1

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00_b	1_b	1_b	1_b	000_b	00_h	$96564402375796C69664_h$	0001_b	000_h	0100_b	0011_b

Table F.18 shows the TAM2 response with *TRnd_TAM2* = 72666964_h. The Header is 88002000_h.

Remark: The data are padded before entering in the CMAC computation. Therefore, the last block of CMAC is complete and K1-MAC is used.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is FBEED61835713366_7C85E08F7236A8DEh.

Table F.18 — TAM2 Response — Example 4 for TAM2 Rev1

	TResponse
# of bits	256+96
Description	$67039773F2C3C8E8B1D29254BADEF7C9_h$ || $0E0B9231A0CF8735023ED8430A9EE40C_h$ || $93B88C6C88230ADEF9D7B62A_h$

[a] "67" is the MSB and "2A" is the LSB.

F.4.2.5 Example 5 for TAM2 Rev1

Table F.19 shows a TAM2 message that specifies the use of memory profile #1, an offset of zero and eight 16-bit blocks of custom data using plaintext mode (i.e. no encryption and no message authentication).

Table F.19 — TAM2 Message — Example 5 for TAM2 Rev1

	Auth Method	Custom Data	Block Size	TAM2_ Rev	TAM2_ RFU	Key ID	IChallenge _TAM2	Profile	Offset	Block Count	Prot Mode
# of bits	2	1	1	1	3	8	80	4	12	4	4
Description	00_b	1_b	1_b	1_b	000_b	00_h	$96564402375796C69664_h$	0001_b	000_h	0111_b	0000_b

Table F.20 shows the TAM2 response with *TRnd_TAM2* = 72666964_h. The Header is 88003800_h.

© ISO/IEC 2017 – All rights reserved

Table F.20 — TAM2 Response — Example 5 for TAM2 Rev1

	TResponse
# of bits	384
Description	D43EF22E787383D87DAFD67617A7D6EE$_h$ || 88003800A16A76656E69617A5265676E$_h$ || 756D5475000000000000000000000000$_h$
[a] "D4" is the MSB and "00" is the LSB.	

F.5 Example for IAM2

This subclause shows an example for IAM2 with *IRnd_IAM2* = 8852BE01$_h$ and with *Purpose_IAM2=0000$_b$*

Table F.21 shows the IAM1 message.

Table F.21 — IAM1 message example

	AuthMethod	Step	IAM1_RFU	KeyID
# of bits	2	2	4	8
Description	01$_b$	00$_b$	0000$_b$	01$_h$

Table F.22 shows the IAM1 response.

Table F.22 — IAM1 response example

	TResponse
# of bits	80
Description	9786CAFE01BB65DC2300$_h$

Table F.23 shows the IAM2 message.

Table F.23 — IAM2 message example

	Auth Method	Step	Custom Data	IAM2 _RFU	IResponse
# of bits	2	2	1	3	128
Description	01$_b$	01$_b$	0$_b$	000$_b$	2C79B76E2EF8B47F6DC4E861EA2F3D5E$_h$

F.6 Example for IAM3

F.6.1 Use of IAM3

IAM3 is used to add custom data to the Interrogator authentication procedure.

This subclause shows examples for IAM3 with *IRnd_IAM3* = 8852BE01$_h$, *Purpose_IAM3=0000$_b$* and *CUSTOMDATA* = AABBCCDDEEFF00010203040506070809$_h$.

Table F.24 shows the IAM1 message.

Table F.24 — IAM1 message example

	AuthMethod	Step	IAM1_RFU	KeyID
# of bits	2	2	4	8
Description	01b	00b	0000b	01h

Table F.25 shows the IAM1 response.

 © ISO/IEC 2017 – All rights reserved

Table F.25 — IAM1 response example

	TResponse
# of bits	80
Description	9786CAFE01BB65DC2300h

F.6.2 Example 1

Table F.26 shows an IAM3 message that specifies the use of memory profile #1, an offset of zero and two 64-bit blocks of custom data using CBC mode encryption. The Header is 0800080000000000_h.

Table F.26 — IAM3 message example

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	384
Descrip-tion	01_b	01_b	1_b	0_b	00_b	0001_b	000_h	0001_b	0001b	AEB86FD061758873D80B-9416C84E3692$_h$ || 2FA00D4893FCBDF-38D05E97AF1238C9A$_h$ || A6DBD-01167E8D88572AF307F8E-7025B4$_h$

F.6.3 Example 2

Table F.27 shows an IAM3 message that specifies the use of memory profile #1, an offset of zero and two 64-bit blocks of custom data using CMAC mode for message authentication. The Header is 0800080000000000_h.

Remark: The data are padded before entering in the CMAC computation. Therefore, the last block of CMAC is complete and K1-MAC is used.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is $19D0CA0C238737F6A68572F59AF062FA_h$.

Table F.27 — IAM3 message example

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	384+96
De-scrip-tion	01_b	01_b	1_b	0_b	00_b	0001_b	000_h	0001_b	0010_b	D1A9761C9A8565A-CA926891E912275E3$_h$ || 0800080000000000AABBCCD-DEEFF0001$_h$ || 0203040506070809000000000000000$_h$ || 88BE5739F14DD1647E758E6E$_h$

F.6.4 Example 3

Table F.28 shows an IAM3 message that specifies the use of memory profile #1, an offset of zero and two 64-bit blocks of custom data using CBC mode encryption and CMAC mode for message authentication. The Header is 0800080000000000_h.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is $19D0CA0C238737F6A68572F59AF062FA_h$.

© ISO/IEC 2017 – All rights reserved

Table F.28 — IAM3 message example

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	384+96
Description	01_b	01_b	1_b	0_b	00_b	0001_b	000_h	0001_b	0011_b	5D3F017B042E02228D26C90DBDB-513DA$_h$ || 4363BE97E6CD0126036A742F-4C1E9810$_h$ || 78CBCEE527213D6ABE22B537CD-426D63$_h$ || 921E18FC050A4A572389499E$_h$

F.6.5 Example 4

Table F.29 shows an IAM3 message that specifies the use of memory profile #1, an offset of two and two 16-bit blocks of custom data (CUSTOMDATA = AABBCCDD$_h$) using CBC mode encryption and CMAC mode for message authentication. The Header is 88010800$_h$.

Remark: The data are padded before entering in the CMAC computation. Therefore, the last block of CMAC is complete and K1-MAC is used.

NOTE The internally derived intermediate K1-MAC key (according to NIST CMAC specification) is 19D0CA0c238737F6A68572F59AF062FA$_h$.

Table F.29 — IAM3 message example

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	256+96
Description	01_b	01_b	1_b	1_b	00_b	0001_b	002_h	0001_b	0011_b	5D3F017B042E02228D-26C90DBDB513DA$_h$ || 49E163B8C7A3316B-A0A1E3EB62088358$_h$ || 624C8B0CDBE35E089F1C355F$_h$

F.6.6 Example 5

Table F.30 shows an IAM3 message that specifies the use of memory profile #1, an offset of zero and one 16-bit block of custom data (CUSTOMDATA = AABB$_h$) using Plaintext mode (i.e. no encryption and no message authentication). The Header is 88000000$_h$.

Table F.30 — IAM3 message example

	Auth Method	Step	Custom Data	Block Size	IAM3_ RFU	Profile	Offset	Block Count	Prot Mode	IResponse
# of bits	2	2	1	1	2	4	12	4	4	256
Description	01_b	01_b	1_b	1_b	00_b	0001_b	000_h	0000_b	0000_b	2C79B76E2EF8B47F6D-C4E861EA2F3D5E$_h$ || 88000000AAB-B00000000000000000000$_h$

F.7 Example for MAM1, MAM2

This subclause shows an example for MAM1 and MAM2 with *TChallenge_MAM1* = 566F6E20427261756E20$_h$ and *Purpose_MAM2=0000$_b$*.

Table F.31 shows the MAM1 message.

© ISO/IEC 2017 – All rights reserved

Table F.31 — MAM1 message example

	AuthMethod	Step	MAM1_RFU	KeyID	IChallenge_MAM1
# of bits	2	2	4	8	80
Description	10_b	00_b	0000_b	01_h	$96564402375796C69664_h$

Table F.32 shows the MAM1 response.

Table F.32 — MAM1 response example

	TResponse
# of bits	**176**
Description	$47D77088E9143699215C9D162D8A310A_h$ || $566F6E204272_h$

Table F.33 shows the MAM2 message.

Table F.33 — MAM2 message example

	Auth Method	Step	Custom Data	MAM2 _RFU	IResponse
# of bits	2	2	1	3	128
Description	10_b	01_b	0_b	000_b	$F537798C9E30CF1999E3C52A7994BD93_h$

© ISO/IEC 2017 – All rights reserved

Bibliography

[1] ISO/IEC 15693-3:2009/AMD4:2017, *Identification cards — Contactless integrated circuit cards — Vicinity cards — Part 3: Anticollision and transmission protocol — AMENDMENT 4: Security framework*

[2] ISO/IEC 18033-3, *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers*

[3] ISO/IEC 18000-3, *Information technology — Radio frequency identification for item management — Part 3: Parameters for air interface communications at 13,56 MHz*

[4] EPC™ Radio-Frequency Identity Protocols, UHF RFID Generation-2 Version 2.0.1, Specification for RFID Air Interface Protocol for Communications at 860 MHz – 960 MHz; GS1 EPCglobal™ Inc.

[5] ERC REC 70-03, Relating to the Use of Short Range Devices (SRD), Annex 1, Band e; European Radio-communications Committee (ERC)

[6] NIST Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication

[7] NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation: Methods and Techniques

© ISO/IEC 2017 – All rights reserved

ICS 35.040.50
Price based on 58 pages

© ISO/IEC 2017 – All rights reserved