BS ISO/IEC 9594-11:2025

BSI Standards Publication

# Information technology — Open systems interconnection directory

Part 11: Protocol specifications for secure operations

bsi.

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2025

# National foreword

This British Standard is the UK implementation of ISO/IEC 9594-11:2025. It supersedes BS ISO/IEC 9594-11:2020, which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee IST/6, Data communications.

A list of organizations represented on this committee can be obtained on request to its committee manager.

**Contractual and legal considerations**

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2025
Published by BSI Standards Limited 2025

ISBN 978 0 539 36055 4

ICS 35.100.70

**Compliance with a British Standard cannot confer immunity from legal obligations.**

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 September 2025.

**Amendments/corrigenda issued since publication**

| Date | Text affected |
|---|---|

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2025

International Standard

ISO/IEC 9594-11

Second edition
2025-08

# Information technology — Open systems interconnection directory —

## Part 11: Protocol specifications for secure operations

Reference number
ISO/IEC 9594-11:2025(en)

© ISO/IEC 2025

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2025

## COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2025

All rights reserved. Unless otherwise specified, or required in the context of its implementation, 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
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland



Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2025

# 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.

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.

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

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

For an explanation of 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 www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by ITU-T as ITU-T X.510 (10/2023) and drafted in accordance with its editorial rules, in collaboration with Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*.

This second edition cancels and replaces the first edition (ISO/IEC 9594-11:2020), which has been technically revised.

A list of all parts in the ISO/IEC 9594 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

© ISO 2025 – All rights reserved



Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

**CONTENTS**

*Page*

SECTION 1 – GENERAL ..... 1

1 Scope ..... 1

2 Normative references ..... 1
2.1 Identical Recommendations | International Standards ..... 1
2.2 Paired Recommendations | International Standards equivalent in technical content ..... 1
2.3 International Standards ..... 2
2.4 Other references ..... 2

3 Definitions ..... 2
3.1 OSI reference model definitions ..... 2
3.2 Directory model definitions ..... 2
3.3 Public-key and attribute certificate definitions ..... 2
3.4 Terms defined in this Recommendation | International Standard ..... 3

4 Abbreviations ..... 4

5 Conventions ..... 5

6 Communication model ..... 5

7 Common data types and special cryptographic algorithms ..... 5
7.1 Introduction ..... 5
7.2 ASN.1 information object class specification tool ..... 5
7.3 Parameterized data types ..... 7
7.4 Multiple cryptographic algorithm specifications ..... 9
7.5 Parameterized data types for providing multiple-cryptographic algorithm-values ..... 15
7.6 Formal specification of encipherment ..... 16

8 Symmetric-key algorithms ..... 17
8.1 Introduction to symmetric-key algorithms ..... 17
8.2 Advance encryption standard (AES) – symmetric-key algorithms ..... 18
8.3 Camellia symmetric-key algorithms ..... 21
8.4 SEED – symmetric-key algorithms ..... 24
8.5 SM4 – symmetric-key algorithms ..... 26

9 Public-key and digital signature algorithms ..... 28

10 Key establishment algorithms ..... 28
10.1 General ..... 28
10.2 Diffie-Hellman over prime field ..... 28
10.3 Elliptic curve Diffie-Hellman ..... 30
10.4 Key derivation ..... 31

11 General concepts for securing protocols ..... 32
11.1 Introduction ..... 32
11.2 Protected protocol plug-in concept ..... 32
11.3 Communication structure ..... 32
11.4 Another view of the relationship between the wrapper protocol and the protected protocol ..... 33
11.5 Structure of the application protocol data unit ..... 33
11.6 Exception conditions ..... 33

SECTION 2 – THE WRAPPER PROTOCOL ..... 34

12 Wrapper protocol general concepts ..... 34
12.1 Introduction ..... 34
12.2 UTC time specification ..... 34
12.3 Use of alternative cryptographic algorithms ..... 34
12.4 Establishment of symmetric keys ..... 34
12.5 Sequence numbers ..... 35
12.6 Use of invocation identification in the wrapper protocol ..... 35
12.7 Mapping to underlying services ..... 35
12.8 Addressing of communicating entities ..... 35


© ISO/IEC 2025 – All rights reserved


Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

12.9 Definition of protected protocols ... 35
12.10 Overview of wrapper protocol data units ... 35

13 Association management ... 36
13.1 Introduction to association management ... 36
13.2 Association handshake request ... 36
13.3 Association handshake accept ... 37
13.4 Association reject due to security issues ... 38
13.5 Association reject by the protected protocol ... 39
13.6 Handshake security abort ... 40
13.7 Handshake abort by protected protocol ... 40
13.8 Data transfer security abort ... 41
13.9 Abort by protected protocol ... 41
13.10 Release request WrPDU ... 42
13.11 Release response WrPDU ... 42
13.12 Release collision ... 43

14 Data transfer phase ... 43
14.1 Symmetric keys renewal ... 43
14.2 Data transfer by the client ... 44
14.3 Data transfer by the server ... 45

15 Information flow ... 48
15.1 Purpose and general model ... 48
15.2 Protected protocol SAOC ... 49
15.3 Wrapper SAOC ... 49

16 Wrapper error handling ... 52
16.1 General ... 52
16.2 Checking of a wrapper handshake request ... 52
16.3 Checking of a wrapper handshake accept ... 53
16.4 Checking of data transfer WrPDUs ... 54
16.5 Wrapper diagnostic codes ... 56

17 End-to-end communications ... 57

SECTION 3 – PROTECTED PROTOCOLS ... 58

18 Authorization and validation list management ... 58
18.1 General on authorization and validation management ... 58
18.2 Defined protected protocol data unit types ... 58
18.3 Authorization and validation management protocol initialization request ... 59
18.4 Authorization and validation management protocol initialization accept ... 59
18.5 Authorization and validation management protocol initialization reject ... 59
18.6 Authorization and validation management protocol initialization abort ... 59
18.7 Add authorization and validation list request ... 60
18.8 Add authorization and validation list response ... 61
18.9 Replace authorization and validation list request ... 61
18.10 Replace authorization and validation list response ... 61
18.11 Delete authorization and validation list request ... 62
18.12 Delete authorization and validation list response ... 62
18.13 Authorization and validation list abort ... 63
18.14 Authorization and validation list error codes ... 63

19 Certification authority subscription protocol ... 64
19.1 Certification authority subscription introduction ... 64
19.2 Defined protected protocol data unit types ... 64
19.3 Certification authority subscription protocol initialization request ... 65
19.4 Certification authority subscription protocol initialization accept ... 65
19.5 Certification authority subscription protocol initialization reject ... 65
19.6 Certification authority subscription protocol initialization abort ... 65
19.7 Public-key certificate subscription request ... 66
19.8 Public-key certificate subscription response ... 66
19.9 Public-key certificate un-subscription request ... 67
19.10 Public-key certificate un-subscription response ... 68
19.11 Public-key certificate replacements request ... 69
19.12 Public-key certificate replacement response ... 69


© ISO/IEC 2025 – All rights reserved


AVA Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

19.13 End-entity public-key certificate updates request ..... 70
19.14 End-entity public-key certificate updates response ..... 71
19.15 Certification authority subscription abort ..... 72
19.16 Certification authority subscription error codes ..... 72

20 Trust broker protocol ..... 72
20.1 Introduction ..... 72
20.2 Defined protected protocol data unit types ..... 73
20.3 Trust broker protocol initialization request ..... 73
20.4 Trust broker protocol initialization accept ..... 73
20.5 Trust broker protocol initialization reject ..... 73
20.6 Trust broker protocol initialization abort ..... 74
20.7 Trust broker request syntax ..... 74
20.8 Trust broker response syntax ..... 74
20.9 Trust broker error information ..... 75

Annex A Crypto Tools in ASN.1 ..... 76

Annex B General cryptographic algorithms ..... 81

Annex C Wrapper protocol in ASN.1 ..... 92

Annex D Protected protocol interface to the wrapper protocol ..... 97

Annex E Authorization and validation list management in ASN.1 ..... 99

Annex F Certification authority subscription in ASN.1 ..... 102

Annex G Trust broker in ASN.1 ..... 106

Annex H Migration of cryptographic algorithms ..... 108
H.1 Migration of cryptographic algorithms ..... 108
H.2 Migration tools or migration approaches ..... 109
H.3 Migration of public-key certificates and other data types using the extension mechanism ..... 110
H.4 General migration approach for communication protocols ..... 110
H.5 Use of multiple and choice cryptographic algorithms ..... 111

Annex I Auxiliary specifications ..... 114

Annex J Amendments and corrigenda ..... 119

Bibliography ..... 120


© ISO/IEC 2025 – All rights reserved


Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

**Introduction**

The Internet Engineering Task Force (IETF) maintains a substantial set of protocols for supporting public-key infrastructure (PKI). Recommendation ITU-T X.510 | ISO/IEC 9594-11 provides protocols to supplement those protocols developed by IETF, especially for:

a) supporting new functions specified by Rec. ITU-T X.509 | ISO/IEC 9594-8, for which IETF has not provided support, e.g., support for authorization and validation list (AVL) maintenance;

b) constraint environments, where lean protocols are required.

In addition, it specifies:

c) a wrapper protocol that provides security services for other protocols.

This Recommendation | International Standard consist of three sections as follows.

Section 1 gives general specifications for this Recommendation | International Standard.

Section 2 is the wrapper protocol specification.

Section 3 specifies some protocols to be protected by the wrapper protocol:

a) a protocol for maintaining authorization and validation lists (AVLs);

b) a protocol for subscribing public-key certificate status information from certification authorities (CAs); and

c) a protocol for accessing a trust broker.

The following annexes are included:

Annex A, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for specifications to be imported by protocols providing a migration path for cryptographic algorithms.

Annex B, which is an integral part of this Recommendation | International Standard, provides cryptographic algorithm specification.

Annex C, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the wrapper protocol.

Annex D, which is an integral part of this Recommendation | International Standard, provides specifications for how a protected protocol is wrapped by the wrapper protocol.

Annex E, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for maintenance of the authorization and validation lists (AVLs) protocol.

Annex F, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for certification authority subscription protocol.

Annex G, which is an integral part of this Recommendation | International Standard, provides the ASN.1 module for the trust broker protocol.

Annex H, which is not an integral part of this Recommendation | International Standard, provides guidance for cryptographic algorithm migration.

Annex I, which is not an integral part of this Recommendation | International Standard, reproduces the ASN.1 specification from other Specifications in the Rec. ITU X.500 Series | ISO/IEC 9594-all parts.

Annex J, which is not an integral part of this Recommendation | International Standard, provides lists amendments and corrigenda included in this Specification.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2025

**INTERNATIONAL STANDARD**
**ITU-T RECOMMENDATION**

# Information Technology – Open systems Interconnection – The Directory – Protocol specifications for secure operations

## SECTION 1 – GENERAL

## 1 Scope

This Recommendation | International Standard provides guidance on how to prepare new and old protocols for cryptographic algorithm migration and defines auxiliary cryptographic algorithms to be used for migration purposes.

This Recommendation | International Standard specifies a general wrapper protocol that provides authentication, integrity and confidentiality (encryption) protection for other protocols. This wrapper protocol includes a migration path for cryptographic algorithms allowing for smooth migration to stronger cryptographic algorithms as such requirements evolve. This will allow migration to quantum-safe cryptographic algorithms. Protected protocols can then be developed without taking security and cryptographic algorithms into consideration.

This Recommendation | International Standard also includes some protocols to be protected by the wrapper protocol primarily for support of public-key infrastructure (PKI). Other specifications, e.g., Recommendations or International Standards, may also develop protocols designed to be protected by the wrapper protocol.

## 2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

### 2.1 Identical Recommendations | International Standards

- Recommendation. ITU-T X.501 (2019) | ISO/IEC 9594-2:2020, *Information technology – Open Systems Interconnection – The Directory: Models*.
- Recommendation ITU-T X.509 (2019) | ISO/IEC 9594-8:2020, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.*
- Recommendation ITU-T X.680 (2021) | ISO/IEC 8824-1:2021, *Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation*.
- Recommendation ITU-T X.681 (2021) | ISO/IEC 8824-2:2021, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.
- Recommendation ITU-T X.682 (2021) | ISO/IEC 8824-3:2021, *Information technology - Abstract Syntax Notation One (ASN.1): Constraint specification*.
- Recommendation ITU-T X.683 (2021) | ISO/IEC 8824-4:2021, *Information technology - Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.
- Recommendation ITU-T X.690 (2021) | ISO/IEC 8825-1:2021, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.
- Recommendation ITU-T X.691 (2021) | ISO/IEC 8825-2:2021, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER).*

### 2.2 Paired Recommendations | International Standards equivalent in technical content

- Recommendation ITU-T X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications*.

  ISO 7498-2:1989, *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture*.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

### 2.3 International Standards

- ISO/IEC 10116:2017, *Information technology – Security techniques – Modes of operation for a n-bit block cipher*.
- ISO/IEC 18033-3:2010, *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers*.
- ISO/IEC 18033-3:2010/Amd.1:2021, *Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers, Amendment 1: SM4*.

### 2.4 Other references

- IETF RFC 793 (1981), *Transmission Control Protocol*.
- IETF RFC 2104 (1997), *HMAC: Keyed-Hashing for Message Authentication*.
- IETF RFC 3526 (2003), *More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*.
- IETF RFC 5084 (2007), *Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS)*.
- IETF RFC 5114 (2008), *Additional Diffie-Hellman Groups for Use with IETF Standards*.
- IETF RFC 5869 (2010), *HMAC-based Extract-and-Expand Key Derivation Function (HKDF)*.
- IETF RFC 6932 (2013), *Brainpool Elliptic Curves for the Internet Key Exchange (IKE) Group Description Registry*.

## 3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply:

### 3.1 OSI reference model definitions

The following terms are defined in Rec. ITU-T X.800 | ISO 7498-2:

a) confidentiality;
b) cryptography;
c) digital signature.

### 3.2 Directory model definitions

The following term is defined in Rec. ITU-T X.501 | ISO/IEC 9594-2:

a) attribute.

### 3.3 Public-key and attribute certificate definitions

The following terms are defined in Rec. ITU-T X.509 | ISO/IEC 9594-8:

a) attribute certificate;
b) authorization and validation list (AVL);
c) authorization and validation list entity (AVL entity);
d) authorizer;
e) certification authority (CA);
f) certification path;
g) end entity;
h) end-entity public-key certificate;
i) hash function;
j) key agreement;
k) private key;
l) public key;
m) public-key certificate;


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

n) public-key infrastructure (PKI);

o) relying party.

### 3.4 Terms defined in this Recommendation | International Standard

This Recommendation | International Standard defines the following terms:

**3.4.1 abstract syntax**: A specification of application-protocol-data-units by using notation rules that are independent of the encoding technique used to represent them.

NOTE – The term abstract syntax is originally an OSI term but is extended here to be general applicable.

**3.4.2 alternative cryptographic algorithm**: A cryptographic algorithm to which migration is wanted.

**3.4.3 application entity**: An active element embodying a set of capabilities which is pertinent to communication systems, and which is defined for the application layer.

NOTE – The term application entity is originally an open systems interconnection (OSI) term, but is extended here to be generally applicable.

**3.4.4 application protocol data unit (APDU)**: Data that is transmitted as a single unit at the application layer between two application entities.

**3.4.5 association**: A cooperative relationship between two application entities, which enables the communication of information and the coordination of their joint operation for an instance of communication.

**3.4.6 block**: A string of bits of defined length.

**3.4.7 block cipher**: Symmetric encipherment system with the property that the encryption algorithm operates on a block of plain text, i.e., a string of bits of a defined length, to yield a block of ciphertext.

**3.4.8 client**: The entity that initiates an association.

**3.4.9 cryptographic algorithm pluck-in concept**: The ability to specify the use of a cryptographic algorithm in the base protocol, but the exact selection of the cryptographic algorithm is determined later for a particular instance of communication.

**3.4.10 cryptographic check algorithm**: Either an integrity check value (ICV) algorithm or a digital signature algorithm, i.e., an algorithm that takes as an input a message and a symmetric or private key and returns a string of bits that can be used to verify the origin and integrity of the message.

**3.4.11 cryptographic check value**: Output of a cryptographic check function invocation.

**3.4.12 data transfer phase**: The phase from the completion of the establishment of an association to the termination of the association.

**3.4.13 digital signature**: The result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for origin authentication, data integrity and signatory non-repudiation.

**3.4.14 inner cryptographic algorithm**: A cryptographic algorithm that included as a parameter of an outer cryptographic algorithm.

**3.4.15 native cryptographic algorithm**: A cryptographic algorithm used prior to a migration period.

**3.4.16 protected protocol data unit (PrPDU)**: Application protocol data unit defined by an application protocol to be protected by the wrapper protocol.

**3.4.17 protected protocol pluck-in concept**: The ability to specify the inclusion of a communications protocol to be protected in the base protocol, but the exact identification of a specific protocol is determined later for a particular instance of communication.

**3.4.18 outer cryptographic algorithm**: A cryptographic algorithm that has as parameters one or more other cryptographic algorithms, and in this way lets multiple inner cryptographic algorithms have the appearance of a single cryptographic algorithm.

**3.4.19 server**: The entity that accepts or rejects an association request.

**3.4.20 symmetric key**: A cryptographic key used for both encryption of plaintext and decryption of ciphertext.

**3.4.21 wrapper protocol data unit (WrPDU)**: An application protocol data unit carrying security protocol control information and, when relevant, carrying a protected protocol data unit.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

## 4 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply:

| | |
|---|---|
| AEAD | Authenticated Encryption with Associated Data |
| AES | Advanced Encryption Standard |
| APDU | Application Protocol Data Unit |
| ASN.1 | Abstract Syntax Notation One |
| AVL | Authorization and Validation List |
| AVMP | Authorization and Validation Management Protocol |
| BER | Basic Encoding Rules |
| CA | Certification Authority |
| CAM | CAMELLIA |
| CASP | Certification Authority Subscription Protocol |
| CBC | Cipher Block Chaining |
| CFB | Cipher FeedBack |
| CMAC | Cipher-based Message Authentication Code |
| CTR | Counter |
| DER | Distinguished Encoding Rules |
| DH | Diffie-Hellman |
| DNS | Domain Name System |
| ECB | Electronic CodeBook |
| ECDH | Elliptic Curve Diffie-Hellman |
| FQDN | Fully Qualified Domain Name |
| HKDF | HMAC-based extract-and-expand Key Derivation Function |
| HMAC | keyed-Hash Message Authentication Code |
| ICV | Integrity Check Value |
| ICT | Information and Communications Technology |
| ID | Identifier |
| LoA | Loss of Alignment |
| MAC | Message Authentication Code |
| MODP | Modular exponential |
| OFB | Output Feedback |
| OKM | Output Keying Material |
| OSI | Open Systems Interconnection |
| PDU | Protocol Data Unit |
| PKI | Public-Key Infrastructure |
| PMI | Privilege Management Infrastructure |
| PRK | Pseudorandom Key |
| PrPDU | Protected protocol Data Unit |
| RAOC | Receive Application Object Class |
| SAOC | Send Application Object Class |
| SIV | Synthetic Initialization Vector |
| TCP | Transmission Control Protocol |
| UTC | Coordinated Universal Time |
| WrPDU | Wrapper Protocol Data Unit |


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

# 5 Conventions

The term "Specification" (as in "this Specification") shall be taken to mean this Recommendation | International Standard.

If an International Standard or ITU-T Recommendation is referenced within normal text without an indication of the edition, the edition shall be taken to be the one specified in the normative references clause.

This Specification makes extensive use of the abstract syntax notation one (ASN.1) for the formal specification of data types and values, as it is specified in Rec. ITU-T X.680 | ISO/IEC 8824-1, Rec. ITU-T X.681 | ISO/IEC 8824-2, Rec. ITU-T X.682 | ISO/IEC 8824-3, Rec. ITU-T X.683 | ISO/IEC 8824-4, Rec. ITU-T X.690 | ISO/IEC 8825-1 and Rec. ITU-T X.691 | ISO/IEC 8825-2.

This Specification presents ASN.1 notation in the **`bold Courier New`** typeface. When ASN.1 types and values are referenced in normal text, they are differentiated from normal text by presenting them in the **`bold Courier New`** typeface.

# 6 Communication model

Figure 1 illustrates the Internet communication model and its relationship to the OSI reference model. The communication model illustrates that an entity communicates with another entity at different layers.

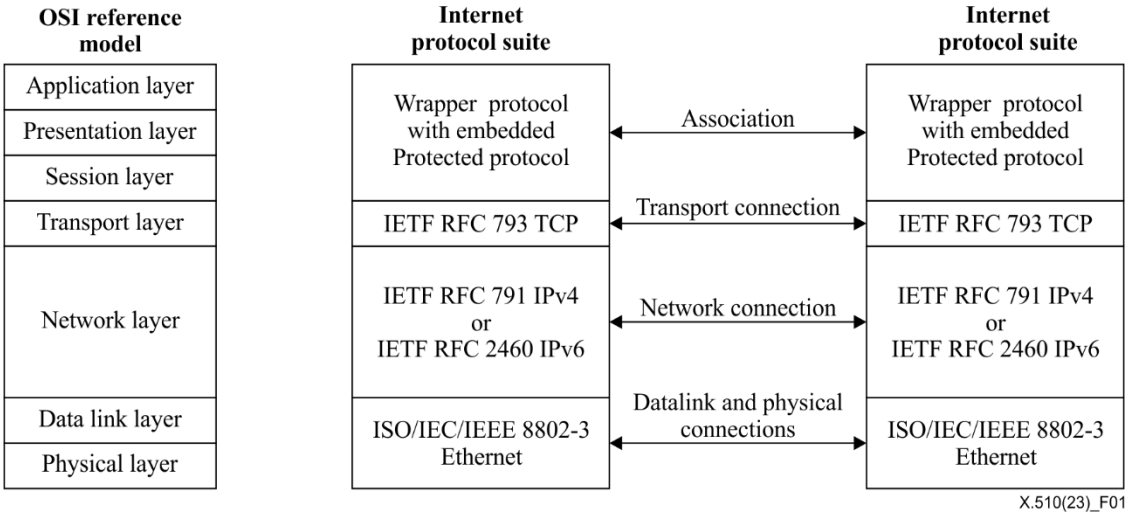


**Figure 1 – Communication model**

# 7 Common data types and special cryptographic algorithms

## 7.1 Introduction

This clause defines some auxiliary cryptographic specification as follows:

a) ASN.1 information object classes are heavily used for protocol design. The **`ALGORITHM`** information object class is important for this Specification. This is further expanded in clause 7.2.

b) Multiple cryptographic algorithms of a specific class may be specified by using a single containing cryptographic algorithm. This is done by utilizing the flexibility provided by the **`AlgorithmIdentifier`** parameterized data type defined in clause 6.2.2 of Rec. ITU-T X.509 | ISO/IEC 9594-8. This is further described in clause 7.3.

c) Some parameterized data types are defined in clause 7.3 for flexible protocol design.

d) Some formal specifications for encipherment are given in clause 7.6.

## 7.2 ASN.1 information object class specification tool

### 7.2.1 General information object class concept

The concept of ASN.1 information object class is specified in Rec. ITU-T X.681 | ISO/IEC 8824-2. It is, together with the parameterized data type concept, vital for protocol design. For that reason, a short introduction is included here to encourage increased use of this facility. ASN.1 information object classes are widely used by the ITU-T X.500 series of


© ISO/IEC 2025 – All rights reserved

Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

Recommendations | ISO/IEC 9594-all parts for defining attribute types, matching rules, extensions to public-key and attribute certificates, etc.

Figure 2 illustrates the general ASN.1 information object class concept. An ASN.1 information object class specifies the general syntax for a class of information objects, e.g., attribute types as defined by Rec. ITU-T X.501 | ISO/IEC 9594-2. From this general specification, specifications for specific attribute types may be derived, e.g., an attribute type for e-mail addresses. From this general e-mail specification, instances of e-mail address attributes may be generated. Instances are what is transferred in protocols or stored in a database (directory).

NOTE – The concept of object classes used in clause 15 is somewhat different from the concept of information object class described above and defined in Rec. ITU-T X.681 | ISO/IEC 8824-2.

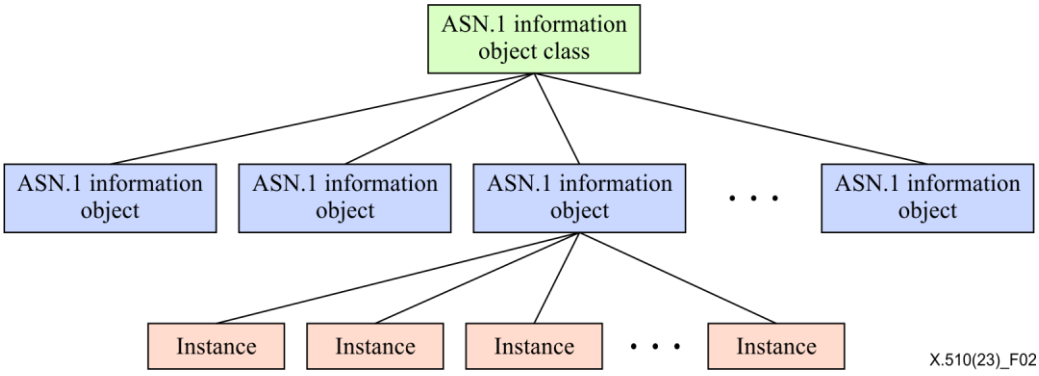


**Figure 2 – ASN.1 information object class concept**

### 7.2.2 The ALGORITHM information object class

The information object class concept is used to define cryptographic algorithms. The **ALGORITHM** information object class is defined in clause 6.2.2 of Rec. ITU-T X.509 | ISO/IEC 9594-8. The specification of this information object class is reproduced below for easy reference. The **ALGORITHM** information object class is different from most other information object classes in the sense that an instance of an information object is an invocation of the algorithm rather than specifying a value identifying something, like an e-mail address.

The following ASN.1 information object class is used for specifying cryptographic algorithms.

```
ALGORITHM ::= CLASS {
  &Type           OPTIONAL,
  &DynParms       OPTIONAL,
  &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
  [PARMS          &Type]
  [DYN-PARMS      &DynParms ]
  IDENTIFIED BY   &id }
```

The **ALGORITHM** information object class has the following fields:

a) The **&Type** field is used to specify those fixed parameters that are necessary for specifying the exact procedure for implementation of the cryptographic algorithm being defined. Not all cryptographic algorithms require such parameters. The field is then absent or has the value **NULL**, as determined by the individual cryptographic algorithm specifications. This field may take data of any syntax.

b) The **&DynParms** field is used to specify those dynamic parameters that determine the value(s) to be exchanged between two communicating entities when invoking the cryptographic algorithm. Not all cryptographic algorithms require dynamic parameters. In this case the **&DynParms** field shall be absent. This field may take data of any syntax.

c) The **&id** field is used to uniquely identify the class of cryptographic algorithm being defined using a unique object identifier.

The **ALGORITHM** information object class defined in Rec. ITU-T X.509 (2016) | ISO/IEC 9594-8:2017 for cryptographic algorithms did not make a distinction between cryptographic algorithm between fixed and dynamic parameters. The **ALGORITHM** information object class has therefore been extended in Rec. ITU-T X.509 (2019) | ISO/IEC 9594-6:2020 to make that distinction. The **ALGORITHM** information object class has been extended in such a way that only cryptographic algorithms having dynamic parameters need to be redefined using the extended information object class, while cryptographic algorithms having only fixed parameters do not need any change.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

When a cryptographic algorithm needs to be redefined using the extended **ALGORITHM** information object class, it requires a new object identifier to be allocated.

Cryptographic algorithms that have never been defined using the **ALGORITHM** information object class may also have object identifiers allocated by this Specification.

## 7.3 Parameterized data types

### 7.3.1 The parameterized data type concept

A parameterized data type is a data type whose syntax is only partially specified and is supplemented by one or more parameters as input to the parameterized data type.

The parameter(s) may be data of any syntax, or it may be limited to a particular type of information. The SIGNED parameterized data type described in clause 6.2.1 of Rec. ITU-T X.509 | ISO/IEC 9594-8, the ICV-Invoke is described in clause 7.5.3, and AUTHEN-ENCRYPT described in clause 7.6.2 are examples of parameterized data types that take data of any syntax as parameter(s). The parameterized data types discussed in clause 7.3.2 require the parameter to be an algorithm specification.

### 7.3.2 Parameterized data types for cryptographic algorithms representation

The different ways of representing a cryptographic algorithm in a communications protocol based on the ALGORITHM information object class are shown below.

The **AlgorithmWithInvoke** parameterized data type defined is used in situations where the type of cryptographic algorithm is signalled together with its invocation.

```
AlgorithmWithInvoke{ALGORITHM:SupportedAlgorithms} ::= SEQUENCE {
  algorithm        ALGORITHM.&id({SupportedAlgorithms}),
  parameters  [0] ALGORITHM.&Type({SupportedAlgorithms}{@algorithm}) OPTIONAL,
  dynamParms         [1]    ALGORITHM.&DynParms({SupportedAlgorithms}{@algorithm})
OPTIONAL,
  ... }
```

The **AlgorithmWithInvoke** parameterized data type has the following components:

a) The **algorithm** component shall hold the object identifier that uniquely identify the cryptographic algorithm being defined.

b) The **parameters** component, when present, shall hold the values of the fixed parameters that further identify the cryptographic algorithm in question. This component shall be present when the **&Type** field is present in the information object for the cryptographic algorithm in question. Otherwise, it shall be absent.

c) The **dynamParms** component, when present, shall hold the value(s) required by the dynamic parameters for the cryptographic algorithm. This component shall be present when the **&DynParms** field is present in the information object for the cryptographic algorithm. Otherwise, it shall be absent.

The **AlgorithmIdentifier** parameterized data type defined as follows is used in situations where the type of cryptographic algorithm is signalled without a corresponding invocation.

```
AlgorithmIdentifier{ALGORITHM:SupportedAlgorithms} ::= SEQUENCE {
  algorithm        ALGORITHM.&id({SupportedAlgorithms}),
  parameters       ALGORITHM.&Type({SupportedAlgorithms}{@algorithm}) OPTIONAL,
  ... }
```

The components of **AlgorithmIdentifier** data type shall be as specified for the corresponding components of the **AlgorithmWithInvoke** parameterized data type.

The **AlgoInvoke** parameterized data type defined as follows is used when the cryptographic algorithm has previously been determined and where only invocation information is required.

```
AlgoInvoke{ALGORITHM:SupportedAlgorithms} ::=
    ALGORITHM.&DynParms({SupportedAlgorithms})
```

### 7.3.3 Relationship between Information object class and parameterized data type

Figure 3 illustrates how the information provided in an **ALGORITHM** object is reflected in a protocol. Figure 3 illustrates the case where both the type of algorithm is signalled including fixed parameter, and the algorithm is also invoked. One example could be the following authenticated encryption with associated data (AEAD) algorithm:


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
aes256-GCM ALGORITHM ::= {
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY {joint-iso-itu-t ds(5) algo(44) ska(2) gcm256(23) aes(1)} }

GCM-ICVlen ::= INTEGER (16)
GCM-nonce  ::= OCTET STRING (SIZE (12))
```

In this case the following sequence is transmitted:

a) the encoded object identifier { joint-iso-itu-t ds(5) algo(44) aead(3) 3 };

b) the encoded integer 16; and

c) an encoded random octet string.

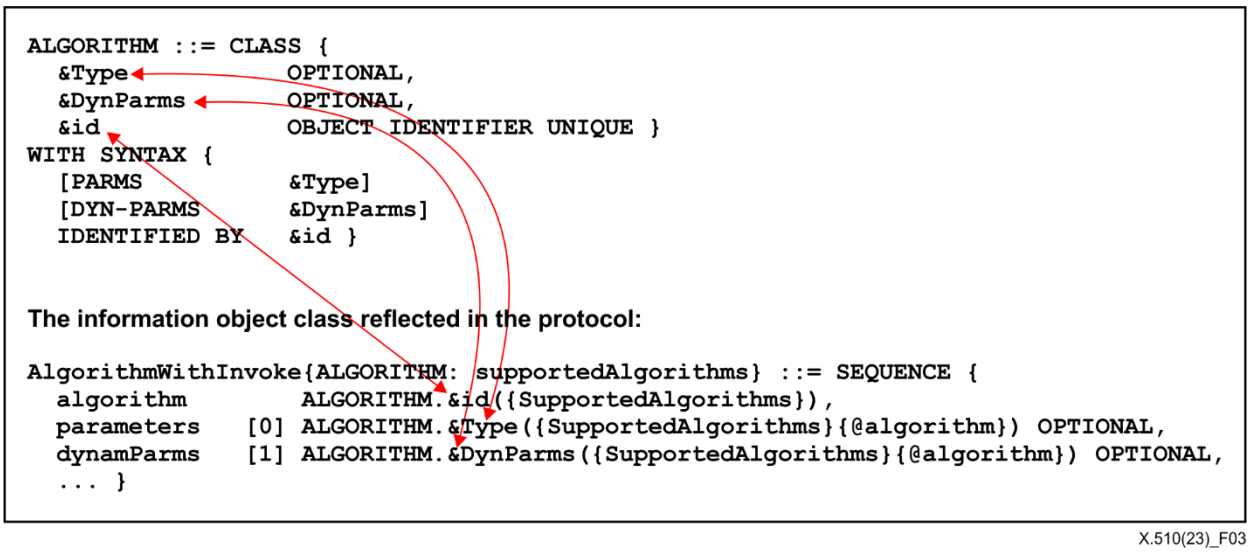


**Figure 3 – How it looks in the communication protocol**

In addition, some information is encrypted and an ICV is generated as specified by the **AUTHEN-ENCRYPT** parameterized data type (see clause 7.6.2).

### 7.3.4 Cryptographic algorithm pluck-in concept

The above-described capabilities may be used to define secure communication protocols without specifying in the base protocol what cryptographic algorithms are to be used. This concept is illustrated in Figure 4.

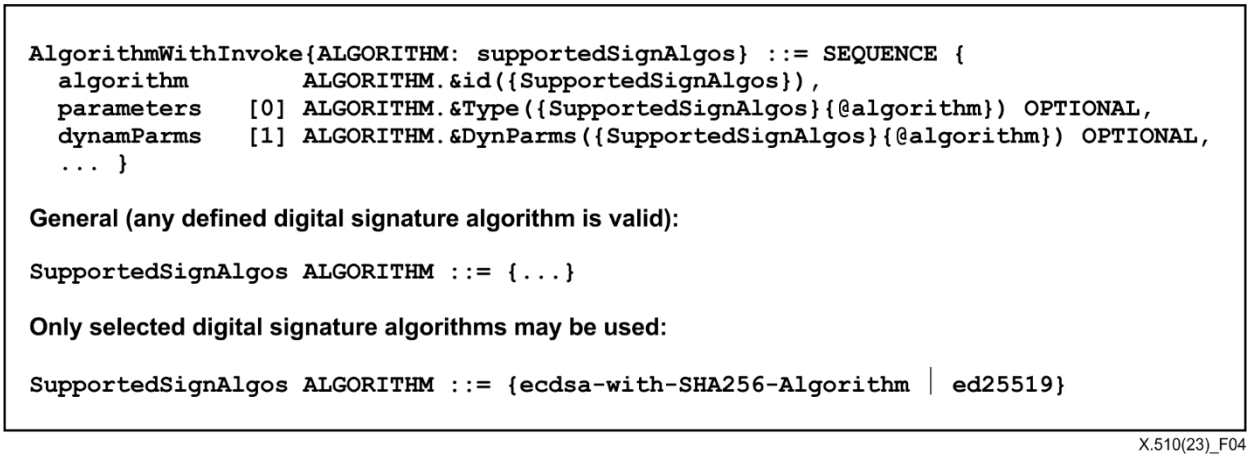


**Figure 4 – Pluck-in concept**


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

The figure illustrates the case where somewhere in a communication protocol a signature algorithm is specified and used. The latter implies that a digital signature is to be generated.

The statement:

```
SupportedSignatureAlgorithms ALGORITHM ::= {...}
```

signals that it is undetermined what digital signature algorithm is to be used, but that any approved digital signature may be used.

The statement:

```
SupportedSignatureAlgorithms ALGORITHM ::= {ecdsa-with-SHA256-Algorithm | ed25519}
```

signals that either the ecdsa-with-SHA256-Algorithm or the ed25519 algorithm may be used and that not any other algorithm is allowed. A referencing standard may require mandatory support for the listed algorithms.

## 7.4 Multiple cryptographic algorithm specifications

### 7.4.1 General

The **&Type** field of the **ALGORITHM** information object class allows any data type to be specified. This is utilized to define **ALGORITHM** information objects that allow for multiple-cryptographic algorithm specifications within a single algorithm specification. This is done by using the **&Type** field to hold a sequence of cryptographic algorithms, which in the following is limited to two elements, where one element is the native algorithm, and the other one optionally holds the alternative algorithm. This technique provides migration facilities as described in Annex H.

### 7.4.2 Algorithm information object sets

The **ALGORITHM** information objects in clauses 7.4.4 to 7.4.7 make reference to open **ALGORITHM** information object sets, i.e., sets that include all defined cryptographic algorithms of a specific type. The intention of providing information object sets here is to make them visible and available for other cybersecurity ASN.1 modules to have a consistent terminology.

The below specification is also included in the formal ASN.1 module **CryptoTools** (see Annex A). This allows organizations, different areas of the world, etc. to customise these object set definitions as explained in clause 7.3.4.

```
SupportedSignatureAlgorithms         ALGORITHM ::= {...}
SupportedAltSignatureAlgorithms      ALGORITHM ::= {...}

SupportedSymmetricKeyAlgorithms      ALGORITHM ::= {...}
SupportedAltSymmetricKeyAlgorithms   ALGORITHM ::= {...}

SupportedPublicKeyAlgorithms         ALGORITHM ::= {...}
SupportedAltPublicKeyAlgorithms      ALGORITHM ::= {...}

SupportedHashAlgorithms              ALGORITHM ::= {...}
SupportedAltHashAlgorithms           ALGORITHM ::= {...}

SupportedKeyEstablishmentAlgos       ALGORITHM ::= {...}
SupportedAltKeyEstablishmentAlgos    ALGORITHM ::= {...}

SupportedAuthenEncryptAlgorithms     ALGORITHM ::= {...}
SupportedAltAuthenEncryptAlgorithms  ALGORITHM ::= {...}

SupportedIcvAlgorithms               ALGORITHM ::= {...}
SupportedAltIcvAlgorithms            ALGORITHM ::= {...}
```

### 7.4.3 Object identifier allocation

The following specifies object identifier arcs for different types of cryptographic algorithms.

Rec. ITU-T X.501 | ISO/IEC 9594-2 allocates the following object identifier for cryptographic algorithms:

```
id-algo OBJECT IDENTIFIER ::= {joint-iso-itu-t(2) ds(5) algorithm(44)}
```

This arc is then further divided into arcs for each of the cryptographic algorithm types.

The following arc is allocated to the multiple-cryptographic algorithms defined in clause 7.4.4:

```
id-algo-mca OBJECT IDENTIFIER ::= {id-algo mca(1)}
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

The following arc is allocated to symmetric key algorithms:

```
id-algo-ska OBJECT IDENTIFIER ::= {id-algo ska(2)} as specified in clause 8.
```

The following arc is allocated to public-key algorithms:

```
id-algo-pka OBJECT IDENTIFIER ::= {id-algo 3}
```

The following arc is allocated to hash algorithms:

```
Id-algo-ha OBJECT IDENTIFIER ::= {id-algo 4}
```

The following arc is allocated to digital signature algorithms:

```
id-algo-dsa OBJECT IDENTIFIER ::= {id-algo 5}
```

The following arc is allocated to key establishment algorithms:

```
id-algo-kem OBJECT IDENTIFIER ::= {id-algo 6}
```

### 7.4.4 Multiple-cryptographic algorithm specifications without invoke

#### 7.4.4.1 General

When cryptographic algorithm specifications are signalled in a protocol request but are not invoked, the following specifications may be used for migration of cryptographic algorithms. Each of the defined outer algorithms allows two inner algorithms to be specified. One of these inner algorithms is designated as the native algorithm and the other, which is an optional algorithm, is designated as the alternative algorithm to be used during migration (see Annex H).

#### 7.4.4.2 Multiple digital signature algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two digital signature algorithms to be specified.

```
multipleSignaturesAlgo ALGORITHM ::= {
  PARMS         MultipleSignaturesAlgo
  IDENTIFIED BY {id-algo-mca 1} }

MultipleSignaturesAlgo ::= SEQUENCE {
  native  AlgorithmIdentifier{{SupportedSignatureAlgorithms}},
  alt     AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}} OPTIONAL }
```

#### 7.4.4.3 Multiple public-key algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two public-key algorithms to be specified.

```
multiplePublicKeyAlgo ALGORITHM ::= {
  PARMS         MultiplePublicKeyAlgo
  IDENTIFIED BY {id-algo-mca 2} }

MultiplePublicKeyAlgo ::= SEQUENCE {
  native  AlgorithmIdentifier{{SupportedPublicKeyAlgorithms}},
  alt     AlgorithmIdentifier{{SupportedAltPublicKeyAlgorithms}} OPTIONAL }
```

#### 7.4.4.4 Multiple hash algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two hash algorithms to be specified.

```
multipleHashAlgo ALGORITHM ::= {
  PARMS         MultipleHashAlgo
  IDENTIFIED BY {id-algo-mca 3} }

MultipleHashAlgo ::= SEQUENCE {
  native  AlgorithmIdentifier{{SupportedHashAlgorithms}},
  alt     AlgorithmIdentifier{{SupportedAltHashAlgorithms}} OPTIONAL }
```

#### 7.4.4.5 Multiple key establishment algorithm

The following is a specification of an **ALGORITHM** information object that allows one or two key establishment algorithms to be specified.

```
multipleKeyEstablishmentAlgo ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  PARMS  MultipleKeyEstablishmentAlgo
  IDENTIFIED BY {id-algo-mca 4} }

MultipleKeyEstablishmentAlgo ::= SEQUENCE {
  native  AlgorithmIdentifier{{SupportedKeyEstablishmentAlgos}},
  alt     AlgorithmIdentifier{{SupportedAltKeyEstablishmentAlgos}} OPTIONAL }
```

#### 7.4.4.6 Multiple sets for data transfer protection algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two sets of cryptographic algorithms to be specified for protecting the data transfer.

Each set of cryptographic algorithms allows a choice between using authenticated encryption with associated data (AEAD) or using a combination of optional encryption together with ICV. The two sets can make different choices.

```
multipleDataPhaseProtection ALGORITHM ::= {
  PARMS         MultipleDataPhaseProtection
  IDENTIFIED BY {id-algo-mca 5} }

MultipleDataPhaseProtection ::= SEQUENCE {
  native        CHOICE {
    aead      [0] AlgorithmIdentifier{{SupportedAuthenEncryptAlgorithms}},
    non-aead  [1] SEQUENCE {
      encr            [0]  AlgorithmIdentifier{{SupportedSymmetricKeyAlgorithms}}
OPTIONAL,
      icv         [1] AlgorithmIdentifier{{SupportedIcvAlgorithms}},
      ... },
    ... },
  alt           CHOICE {
    aead      [0] AlgorithmIdentifier{{SupportedAltAuthenEncryptAlgorithms}},
    non-aead  [1] SEQUENCE {
      encr            [0]  AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}}
OPTIONAL,
      icv         [1] AlgorithmIdentifier{{SupportedAltIcvAlgorithms}},
      ... },
    ... } OPTIONAL }
```

The **native** component reflects the way the data transfer is protected outside migration periods. The component specifies two alternatives:

a) When the **aead** alternative is taken, it shall specify the AEAD algorithm being used.

b) When the **non-aead** alternative is taken, it shall hold the following components:

- The **encr** component, when present, shall hold the symmetric key algorithm to be used for encryption. It shall be present if encryption is required. Otherwise, it shall be absent.
- The **icv** component shall hold the ICV algorithm to be used.

The **alt** component, when present, reflects the way the data transfer shall be protected after a migration period. It shall be present during the migration period if supported by the sending application entity. Otherwise, it shall be absent. It specifies the same alternatives as specified by the **native** component.

### 7.4.5 Multiple choice cryptographic algorithm specifications without invoke

#### 7.4.5.1 General

When cryptographic algorithm specifications are signalled in a protocol request but not invoked, the following specifications may be used for migration of cryptographic algorithms. Each of the defined outer algorithms allows for the choice between two inner algorithms to be specified. One alternative is designated as the native algorithm and the other alternative is designated as the alternative algorithm to be used during migration (see Annex H).

#### 7.4.5.2 Choice digital signature algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two digital signature algorithms.

```
choiceSignatureAlgo ALGORITHM ::= {
  PARMS         ChoiceSignatureAlgo
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  IDENTIFIED BY {id-algo-mca 10} }

ChoiceSignatureAlgo ::= CHOICE {
  native  [0] AlgorithmIdentifier{{SupportedSignatureAlgorithms}},
  alt     [1] AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}} }
```

#### 7.4.5.3 Choice public-key algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two public-key algorithms.

```
choicePublicKeyAlgo ALGORITHM ::= {
  PARMS         ChoicePublicKeyAlgo
  IDENTIFIED BY {id-algo-mca 11} }

ChoicePublicKeyAlgo ::= CHOICE {
  native  [0] AlgorithmIdentifier{{SupportedPublicKeyAlgorithms}},
  alt     [1] AlgorithmIdentifier{{SupportedAltPublicKeyAlgorithms}} }
```

#### 7.4.5.4 Choice hash algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two hash algorithms.

```
choiceHashAlgo ALGORITHM ::= {
  PARMS         ChoiceHashAlgo
  IDENTIFIED BY {id-algo-mca 12} }

ChoiceHashAlgo ::= CHOICE {
  native  [0] AlgorithmIdentifier{{SupportedHashAlgorithms}},
  alt     [1] AlgorithmIdentifier{{SupportedAltHashAlgorithms}} }
```

#### 7.4.5.5 Choice key-establishment algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two key-establishment algorithms.

```
choiceKeyEstablishmentAlgo ALGORITHM ::= {
  PARMS         ChoiceKeyEstablishmentAlgo
  IDENTIFIED BY {id-algo-mca 13} }

ChoiceKeyEstablishmentAlgo ::= CHOICE {
  native  [0] AlgorithmIdentifier{{SupportedKeyEstablishmentAlgos}},
  alt     [1] AlgorithmIdentifier{{SupportedAltKeyEstablishmentAlgos}} }
```

#### 7.4.5.6 Choice sets for data transfer protection algorithms

The following is a specification of the **choiceDataPhaseProtection** algorithm information object that allows a choice of two sets of cryptographic algorithms to be specified for protecting the data transfer. It is only relevant to include it in a handshake accept when the corresponding handshake request included a **multipleDataPhaseProtection** algorithm instance.

Each set of cryptographic algorithms allows a choice between using authenticated encryption with associated data (AEAD) or using a combination of optional encryption and ICV. The same choice shall be made as done for the corresponding **multipleDataPhaseProtection** algorithm.

```
choiceDataPhaseProtection ALGORITHM ::= {
  PARMS         ChoiceDataPhaseProtection
  IDENTIFIED BY {id-algo-mca 14} }

ChoiceDataPhaseProtection ::= CHOICE {
  native    [0] CHOICE {
    aead      [0] AlgorithmIdentifier{{SupportedAuthenEncryptAlgorithms}},
    non-aead  [1] SEQUENCE {
      encr            [0]  AlgorithmIdentifier{{SupportedSymmetricKeyAlgorithms}}
OPTIONAL,
      icv         [1] AlgorithmIdentifier{{SupportedIcvAlgorithms}},
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
      ... }
    },
  alt       [1] CHOICE {
    aead      [0] AlgorithmIdentifier{{SupportedAuthenEncryptAlgorithms}},
    non-aead  [1] SEQUENCE {
      encr            [0]  AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}}
OPTIONAL,
      icv       [1] AlgorithmIdentifier{{SupportedAltIcvAlgorithms}},
      ... }
    } }
```

**ChoiceDataPhaseProtection** data type has two alternatives:

a) The **native** alternative shall be taken outside migration periods or when the application entity has received a request where the **MultipleDataPhaseProtection** data value had the **alt** component included but the application entity has not yet completed migration. The alternative again specifies two alternatives where the same alternative shall be taken as for the corresponding **MultipleDataPhaseProtection** data value:

– When the **aead** alternative is taken, it shall specify the same AEAD algorithm as specified in the corresponding **MultipleDataPhaseProtection** data value.

– When the **non-aead** alternative is taken, it shall hold the following components:

i) The **encr** component shall be present if the corresponding component was present in the corresponding **MultipleDataPhaseProtection** data value, and it shall specify the same value. Otherwise, it shall be absent.

ii) The **icv** component shall hold the same value as in the corresponding component of the **MultipleDataPhaseProtection** data value.

b) The **alt** alternative shall be taken when the application entity has received a request where the **MultipleDataPhaseProtection** data value had the **alt** component included and the application entity has completed migration. This alternative has the same alternatives as the **native** alternative.

### 7.4.6 Multiple-cryptographic algorithm specifications with invoke

#### 7.4.6.1 General

When cryptographic algorithm specifications are not only signalled in a protocol request but are also invoked, the following specifications may be used for the migration of cryptographic algorithms. Each of the defined outer algorithms allows two inner algorithms to be specified. One of these inner algorithms is designated as the native algorithm, and the other, which is an optional algorithm, is designated as the alternative algorithm to be used during migration (see Annex H).

#### 7.4.6.2 Multiple digital signatures algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two digital signature algorithms to be specified.

```
multipleSignaturesAlgoInvoke ALGORITHM ::= {
  PARMS         MultipleSignaturesAlgo
  IDENTIFIED BY {id-algo-mca 20} }

MultipleSignaturesAlgoInvoke ::= SEQUENCE {
  native  AlgorithmWithInvoke{{SupportedSignatureAlgorithms}},
  alt     AlgorithmWithInvoke{{SupportedAltSignatureAlgorithms}} OPTIONAL }
```

#### 7.4.6.3 Multiple public-key algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two public-key algorithms to be specified.

```
multiplePublicKeyAlgoInvoke ALGORITHM ::= {
  PARMS         MultiplePublicKeyAlgoInvoke
  IDENTIFIED BY {id-algo-mca 21} }

MultiplePublicKeyAlgoInvoke ::= SEQUENCE {
  native  AlgorithmWithInvoke{{SupportedPublicKeyAlgorithms}},
  alt     AlgorithmWithInvoke{{SupportedAltPublicKeyAlgorithms}} OPTIONAL }
```


© ISO/IEC 2025 – All rights reserved

Full document details: BS ISO/IEC 9594-11:2025

#### 7.4.6.4 Multiple hash algorithms

The following is a specification of an **ALGORITHM** information object that allows one or two hash algorithms to be specified.

```
multipleHashAlgoInvoke ALGORITHM ::= {
  PARMS         MultipleHashAlgoInvoke
  IDENTIFIED BY {id-algo-mca 22} }

MultipleHashAlgoInvoke ::= SEQUENCE {
  native  AlgorithmWithInvoke{{SupportedHashAlgorithms}},
  alt     AlgorithmWithInvoke{{SupportedAltHashAlgorithms}} OPTIONAL }
```

#### 7.4.6.5 Multiple key establishment algorithm

The following is a specification of an **ALGORITHM** information object that allows one or two integrity check value (ICV) algorithms to be specified.

```
multipleKeyEstablishmentAlgoInvoke ALGORITHM ::= {
  PARMS  MultipleKeyEstablishmentAlgoInvoke
  IDENTIFIED BY {id-algo-mca 23} }

MultipleKeyEstablishmentAlgoInvoke ::= SEQUENCE {
  native  AlgorithmWithInvoke {{SupportedKeyEstablishmentAlgos}},
  alt AlgorithmWithInvoke {{SupportedAltKeyEstablishmentAlgos}} OPTIONAL }
```

### 7.4.7 Multiple choice cryptographic algorithm specifications with invoke

#### 7.4.7.1 General

When cryptographic algorithm specifications are both signalled in a protocol request and invoked, the following specifications may be used for migration of cryptographic algorithms. Each of the defined outer algorithms allows for the choice between two inner algorithms to be specified. One alternative is designated as the native algorithm and the other, which is an optional algorithm, is designated as the alternative algorithm to be used during migration (see Annex H).

#### 7.4.7.2 Choice digital signature algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two digital signature algorithms.

```
choiceSignatureAlgoInvoke ALGORITHM ::= {
  PARMS         ChoiceSignatureAlgoInvoke
  IDENTIFIED BY {id-algo-mca 30} }

ChoiceSignatureAlgoInvoke ::= CHOICE {
  native  [0] AlgorithmWithInvoke{{SupportedSignatureAlgorithms}},
  alt     [1] AlgorithmWithInvoke{{SupportedAltSignatureAlgorithms}} }
```

#### 7.4.7.3 Choice public-key algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two public-key algorithms.

```
choicePublicKeyAlgoInvoke ALGORITHM ::= {
  PARMS         ChoicePublicKeyAlgoInvoke
  IDENTIFIED BY {id-algo-mca 31} }

ChoicePublicKeyAlgoInvoke ::= CHOICE {
  native  [0] AlgorithmWithInvoke{{SupportedPublicKeyAlgorithms}},
  alt     [1] AlgorithmWithInvoke{{SupportedAltPublicKeyAlgorithms}} }
```

#### 7.4.7.4 Choice hash algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two hash algorithms.

```
choiceHashAlgoInvoke ALGORITHM ::= {
  PARMS         ChoiceHashAlgoInvoke
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  IDENTIFIED BY {id-algo-mca 32} }

ChoiceHashAlgoInvoke ::= CHOICE {
  native   [0] AlgorithmWithInvoke{{SupportedHashAlgorithms}},
  alt      [1] AlgorithmWithInvoke{{SupportedAltHashAlgorithms}} }
```

#### 7.4.7.5 Choice key establishment algorithm

The following is a specification of an **ALGORITHM** information object that allows a choice between two key establishment algorithms.

```
choiceKeyEstablishmentAlgoInvoke ALGORITHM ::= {
  PARMS          ChoiceKeyEstablishmentAlgoInvoke
  IDENTIFIED BY {id-algo-mca 33} }

ChoiceKeyEstablishmentAlgoInvoke ::= CHOICE {
  native    [0] AlgorithmWithInvoke{{SupportedKeyEstablishmentAlgos}},
  alt       [1] AlgorithmWithInvoke{{SupportedAltKeyEstablishmentAlgos}} }
```

### 7.5 Parameterized data types for providing multiple-cryptographic algorithm-values

#### 7.5.1 Multiple digital signatures attached to data

The **MULTI_SIGNED** parameterized data type is an expansion of the **SIGNED** data type defined in clause 6.2.1 of Rec. ITU-T X.509 | ISO/IEC 9594-8. It allows for multiple signatures to be attached to data to be digitally signed.

```
MULTY-SIGNED{ToBeSigned} ::= SEQUENCE {
  toBeSigned  ToBeSigned,
  algorithm   ALGORITHM.&id({multipleSignaturesAlgo}),
  parmeters   SEQUENCE SIZE (1..MAX) OF
    sign        SEQUENCE {
      algo        AlgorithmIdentifier{{SupportedSignatureAlgorithms}},
      signature   BIT STRING,
      ... },
  ... }
```

Depending on the deployment, it may be required that all signatures be verified. This requires that the recipient supports all the involved digital signature algorithms. In other cases, a single signature is sufficient for verification.

#### 7.5.2 Double digital signature attached to data

The **Signed** parameterized data type is used instead of the **SIGNED** data type, when signalling of digital signature algorithms is not necessary.

```
Signed{ToBeSigned} ::= SEQUENCE {
  toBeSigned   ToBeSigned,
  signature    BIT STRING,
  altSignature BIT STRING OPTIONAL,
  ... }
```

The **Signed** data type has the following components:

a) the **toBeSigned** component shall hold the value of the data type to be signed;

b) the **signature** component shall hold what is called the native digital signature to be attach to the data – the digital signature shall be generated according to a digital signature algorithm that has previously been signalled;

c) the **altSignature** component may be present if an alternative digital signature algorithm previously has been signalled – otherwise, it shall be absent.

#### 7.5.3 Duplicate integrity check values attached to data

The **ICV** parameterized data types are counterparts to the **SIGNED** parameterized data type as defined in clause 6.2.1 of Rec. ITU-T X.509 | ISO/IEC 9594-8. They provide for attaching one or two ICVs to data whose integrity is to be protected.

The **ICV-Total** parameterized data type is used where the ICV algorithm has not previously been agreed between two communication entities.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
ICV-Total{ToBeProtected} ::= SEQUENCE {
  toBeProtected               ToBeProtected,
  algorithmIdentifier         AlgorithmWithInvoke{{SupportedIcvAlgorithms}},
  icv                         OCTET STRING,
  altAlgorithmIdentifier      [0]   AlgorithmWithInvoke{{SupportedIcvAlgorithms}}
OPTIONAL,
  altIcv                  [1] OCTET STRING OPTIONAL,
  ... }
   (WITH COMPONENTS {..., altAlgorithmIdentifier PRESENT, altIcv PRESENT } |
    WITH COMPONENTS {..., altAlgorithmIdentifier ABSENT,  altIcv ABSENT } )
```

The **ICV-Total** data type has the following components:

a) The **toBeProtected** component shall specify the data value of what is passed in a parameter, being the data value to be protected.

b) The **algorithmIdentifier** component consists of the following components (see clause 7.2.2):

- the **algorithm** component shall hold the object identifier assigned to the ICV algorithm in question;
- the **parameters** component, when present, shall hold the value of the fixed parameters – this component shall be present when the **&Type** field is present in the information object for the ICV algorithm in question – otherwise, it shall be absent;
- The **dynamParms** component, when present, shall hold the value of the dynamic parameters – this component shall be present when the **&DynParms** field is present in the information object for the ICV algorithm in question – otherwise, it shall be absent.

c) The **icv** component shall hold the value of the generated ICV using the ICV algorithm identified by the **algorithm** and **parameters** components listed in b) and by applying the **dynamParms** component, if relevant.

d) The **altAlgorithmIdentifier** component, when present, consists of the same components as the **algorithmIdentifier** component.

e) The **altIcv** component, when present, shall hold the value of the generated ICV using ICV algorithm identified by the **algorithm** and **parameters** components of the **altAlgorithmIdentifier** component.

The **altAlgorithmIdentifier** and **altIcv** components shall either both be present, or both be absent.

The **ICV-Invoke** parameterized data type is used where the ICV algorithm has previously been agreed between two communication entities, e.g., during a handshake.

```
ICV-Invoke{ToBeProtected} ::= SEQUENCE {
  toBeProtected       ToBeProtected,
  dynParms        [0] AlgoInvoke{{SupportedIcvAlgorithms}} OPTIONAL,
  icv                 BIT STRING,
  ... }
```

The **ICV-Total** data type has the following components:

a) The **toBeProtected** component shall specify the data value of what is passed in a parameter, being the data value to be protected.

b) The **dynParms** component, when present, shall hold the value of the dynamic parameters associated with the ICV algorithm in question. This component shall be present when the **&DynParms** field is present in the information object for that ICV algorithm. Otherwise, it shall be absent.

c) The **icv** component shall hold the generated ICV using the previously agreed ICV algorithm and, if relevant, by applying the **dynamParms** component.

## 7.6 Formal specification of encipherment

### 7.6.1 Formal specification of encryption

```
ENCIPHERED{ToBeEnciphered} ::= OCTET STRING (CONSTRAINED BY {
   -- shall be the result of applying an encipherment procedure
   -- to the encoded octets of a value of -- ToBeEnciphered } )
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

### 7.6.2 Formal specification of authenticated encryption with associated data

The following parameterized data type is used to specify AEAD.

```
AUTHEN-ENCRYPT{ToBeAuth, ToBeEnciphered} ::= SEQUENCE {
  aad  ToBeAuth,
  encr ToBeEnciphered,
  ... }
```

IETF RFC 5084 is a specification for the use of AEAD. IETF RFC 5084 has associated data as optional, while it is mandatory in this Specification.

## 8 Symmetric-key algorithms

### 8.1 Introduction to symmetric-key algorithms

The concept of symmetric-key algorithms covers not only the use of symmetric keys for encryption but also authenticated encryption and ICV.

Encryption is conversion of the original representation of the plaintext information into ciphertext using a symmetric key. The recipient of a ciphertext message can convert this message back into plaintext by use of the same symmetric key that was used for producing the ciphertext.

Encryption techniques distinguish between stream ciphers and block ciphers. Block cipher divides a message into blocks of a certain size, typically 64 or 128 bits, and then encrypts each block to generate a block of ciphertext. A similar technique is used for decryption. In stream cipher, one bit or one octet is encoded/decoded as the stream passes by.

Currently, this Specification only considers block ciphers. There are two aspects to consider when it comes to block ciphers. They are encryption/decryption of the individual blocks, and how the different blocks are combined for encryption/decryption of a complete message. Those two aspects are independent of each other. Therefore, ISO/IEC 10116 defines operation modes that are independent of the encryption techniques.

From a communication protocol perspective, only the combination of an encryption technique and an operational mode constitute a symmetric key algorithm.

There are five block cipher operation modes:

a) Electronic codebook (ECB) mode:

   Each block is encrypted independently of other blocks. The same applies for decryption. If the same plaintext is encrypted multiple times, it always results in the same ciphertext. This is considered a security issue. Use of this mode is therefore not recommended.

b) Cipher block chaining (CBC) mode:

   Each block is XOR-ed before encryption with the previous ciphertext block. The first block is XOR-ed with an initialization vector (IV) of the same size as the block size. The same IV is used for encryption and decryptions of a message of data. The IV shall be generated by a good random number generator. For security, a new random IV is used for each message to be encrypted.

c) Cipher feedback (CFB) mode:

   The encryption operations start by placing an IV with the same length of a block in a shift register and encrypting it. This results in an encrypted string where only some $s$ bits (typically 8 bits) are used. These bits are XOR-ed with the first $s$ bits of the message to be encrypted to generate the first $s$ bits of the ciphertext. For the next part to be encrypted, the shift register is shifted $s$ bits to the left and then concatenated with the encrypted $s$ bits of the previously encrypted $s$ bits. The result is then encrypted, where again only $s$ bits are XOR-ed with the next $s$ bits of the input message to create the next $s$ bits of the ciphertext.

d) Output feedback (OFB) mode:

   An IV with the same size as a block is the input to the encryption of the first block. The IV is encrypted and the result is XOR-ed with the first block of plain text to yield the first block of ciphertext. The resulting block of ciphertext is XOR-ed with the next block of plaintext to produce a ciphertext. This procedure is repeated until all plaintext blocks have been enciphered.

e) Counter mode (CRT) mode:

   In counter mode, sequence numbers, called counters, are used together with a block of plain text, one at a time as. An initial counter value is used for the first block of plain text. This counter is the size of half the block size. It is concatenated with a nonce also having the size of half the block. The result is encrypted,


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

and then XORed with the first plain text block to create the first ciphertext block. Subsequent blocks are treated in the same way, except that the counter is increased by a fixed value, e.g., 1, for each block processed. The same nonce is used for the entire process and for decryption. However, it is important that a new nonce is generated for each message to be encrypted.

These modes are described in more detail in Rec. ITU-T X.508 | ISO/IEC 9594-12.

Clauses 8.2 to 8.5 list algorithm definitions for different combinations of encryption techniques and operation modes, which include not only the five modes listed above, but also modes that relate to authenticated encryption and ICVs.

## 8.2 Advance encryption standard (AES) – symmetric-key algorithms

### 8.2.1 General

The National Institute of Standards and Technology has specified the AES algorithms using the original **ALGORITHM** information object class. These algorithms are redefined here using the extended **ALGORITHM** information object class.

AES is specified for key sizes of 128 bits, 192 bits and 256 bits. The block size is 128 bits for all three cases.

### 8.2.2 Advanced encryption standard – electronic codebook mode (AES-ECB)

The three AES-ECB symmetric-key algorithms are defined below:

```
aes-ecb128 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb128(1)} }

aes-ecb192 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb192(2)} }

aes-ecb256 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb256(3)} }
```

Electronic code book (ECB) is the simplest cipher operation mode.

It has the advantage that forward cipher operations can be performed in parallel for encryption and that the inverse cipher operation can also be performed in parallel for decryption.

It has the disadvantage that the same plaintext and the same symmetric key always produce the same ciphertext, and this may make this cipher operation mode subject for cryptanalysis. In applications where that could a problem, this operation mode should not be used.

### 8.2.3 Advanced encryption standard – cipher block chaining mode (AES-CBC)

The three AES-CBC symmetric-key algorithms are defined below:

```
aes-cbc128 ALGORITHM ::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc128(5)} }

aes-cbc192 ALGORITHM ::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc192(6)} }

aes-cbc256 ALGORITHM ::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc256(7)} }

AES-InitializationVector ::= OCTET STRING (SIZE (16))
```

The **AES-InitializationVector** does not have to be secret. It shall be generated as a random value by a high-quality random number generator. The same value for the **AES-InitializationVector** shall be used for encryption and decryption. It therefore needs to be transferred in the clear together with the encrypted value.

An instance of the **AES-InitializationVector** shall only be used for one encryption/decryption. A new value shall be generated for each encryption occurrence. In this way it is ensured that if one same data is encrypted several times, each occurrence results in a very different ciphertext.


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

### 8.2.4 Advanced encryption standard – cipher feedback mode (AES-CFB)

The three AES-CFB symmetric-key algorithms are defined below:

```
aes-cfb128 ALGORITHM ::= {
  PARMS          NumberOfBits
  DYN-PARMS      AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb128(13)} }

aes-cfb192 ALGORITHM ::= {
  PARMS          NumberOfBits
  DYN-PARMS      AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb192(14)} }

aes-cfb256 ALGORITHM ::= {
  PARMS          NumberOfBits
  DYN-PARMS      AES-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb256(15)} }

NumberOfBits ::= INTEGER(1..128)
```

### 8.2.5 Advanced encryption standard – output feedback mode (AES-OFB)

The three AES-OFB symmetric-key algorithms are defined below:

```
aes-ofb128 ALGORITHM ::= {
  DYN-PARMS      AES-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb128(9)} }

aes-ofb192 ALGORITHM ::= {
  DYN-PARMS      AES-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb192(10)} }

aes-ofb256 ALGORITHM ::= {
  DYN-PARMS     AES-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb256(11)} }
```

For **AES-InitializationVector** handling see clause 8.4.3.

### 8.2.6 Advanced encryption standard – counter mode (AES-CTR)

The three AES-CTR symmetric-key algorithms are defined below:

```
aes-ctr128 ALGORITHM ::= {
  DYN-PARMS      CtrNonce
  IDENTIFIED BY {id-algo-ska ctr128(17)} }

aes-ctr192 ALGORITHM ::= {
  DYN-PARMS      CtrNonce
  IDENTIFIED BY {id-algo-ska ctr192(18)} }

aes-ctr256 ALGORITHM ::= {
  DYN-PARMS      CtrNonce
  IDENTIFIED BY {id-algo-ska ctr256(19)} }

CtrNonce ::= OCTET STRING (SIZE (8))
```

The initial counter block structure chosen here is the second approach specified in section B.2 of NIST SP 800-38A.

The size of a counter block is 16 octets. The left most (most significant) 8 octets hold the nonce, while the remaining 8 octets are used as counter field. The counter field of the first block of a message shall have the hex value 0x0000 0000 0000 0001. The counter shall be incremented by '1' for each subsequent block of the message.

### 8.2.7 Advanced encryption standard – Galois/counter mode (AES-GCM)

The three AES-GCM authenticated encryption with associated data (AEAD) algorithms are defined below:

```
aes-128gcm ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm128(21)} }

aes-192gcm ALGORITHM ::= {
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm192(22)} }

aes-256gcm ALGORITHM ::= {
  PARMS          GCM-ICVlen
  DYN-PARMS      GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm256(23)} }

GCM-ICVlen ::= INTEGER (16)
GCM-nonce  ::= OCTET STRING (SIZE (12))
```

IETF RFC 5116 specifies in section 5.1 that **GCM-ICVlen** be 16 octets long and recommends in section 3.2 a nonce of 12 octets.

```
GCM-ICVlen ::= INTEGER (16)
GCM-nonce  ::= OCTET STRING (SIZE (12))
```

The security of the AES-GCM algorithms requires never to use the same nonce more than once.

### 8.2.8 Advanced encryption standard – Galois/counter mode – synthetic initialization vector (AES-GCM-SIV)

The three AES-GCM-SIV authenticated encryption with associate data-key algorithms are defined below:

```
aes-128gcm-siv ALGORITHM ::= {
  PARMS          GCM-SIV-ICVlen
  DYN-PARMS      GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm128-siv(25)} }

aes-192gcm-siv ALGORITHM ::= {
  PARMS          GCM-SIV-ICVlen
  DYN-PARMS      GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm192-siv(26)} }

aes-256gcm-siv ALGORITHM ::= {
  PARMS          GCM-SIV-ICVlen
  DYN-PARMS      GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm256-siv(27)} }


GCM-SIV-ICVlen ::= INTEGER (16)
GCM-SIV-nonce  ::= OCTET STRING (SIZE (12))
```

For details, see IETF RFC 5297.

### 8.2.9 Advanced encryption standard (AES) – counter with CBC-MAC (AES-CCM)

The three AES-CCM ICV algorithms are defined below:

```
aes-128ccm ALGORITHM ::= {
  PARMS          CCM-ICVlen
  DYN-PARMS      CCM-nonce
  IDENTIFIED BY {id-algo-ska ccm128(29)} }

aes-192ccm ALGORITHM ::= {
  PARMS          CCM-ICVlen
  DYN-PARMS      CCM-nonce
  IDENTIFIED BY {id-algo-ska 192ccm(30)} }

aes-256ccm ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  PARMS          CCM-ICVlen
  DYN-PARMS      CCM-nonce
  IDENTIFIED BY {id-algo-ska 256ccm(31)} }


CCM-ICVlen ::= INTEGER (16)
CCM-nonce ::= OCTET STRING (SIZE (12))
```

For details, see IETF RFC 5084.

#### 8.2.10 Advanced encryption standard (AES) – Galois Message Authentication Code (AES-GMAC)

The three AES-GMAC ICV algorithms are defined below:

```
aes-128gmac ALGORITHM ::= {
  PARMS          MACLength
  DYN-PARMS      GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac128(33)} }

aes-192gmac ALGORITHM ::= {
  PARMS          MACLength
  DYN-PARMS      GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac192(34) } }

aes-256gmac ALGORITHM ::= {
  PARMS          MACLength
  DYN-PARMS      GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac256(35)} }

MACLength  ::= INTEGER (12 | 13 | 14 | 15 | 16)
GMAC-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.2.11 Advanced encryption standard – cipher-based message authentication code (AES-CMAC)

The three AES-CMAC ICV algorithms are defined below:

```
aes-128cmac ALGORITHM ::= {
  PARMS          CMACLength
  IDENTIFIED BY {id-algo-ska cmac128(37)} }

aes-192cmac ALGORITHM ::= {
  PARMS          CMACLength
  IDENTIFIED BY {id-algo-ska cmac192(38)} }

aes-256cmac ALGORITHM ::= {
  PARMS          CMACLength
  IDENTIFIED BY {id-algo-ska cmac256(39)} }

CMACLength ::= INTEGER
```

### 8.3 Camellia symmetric-key algorithms

#### 8.3.1 General

The Camellia (CAM) encryption technique is specified in ISO/IEC 18033-3, but only for the encryption of a single block. As for AES it is defined for key sizes of 128 bits, 192 bits and 256 bits. The block size is 128 bits.

#### 8.3.2 Camellia – electronic codebook (CAM-ECB) mode

The three CAM-ECB symmetric-key algorithms are defined below:

```
cam-ecb128 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb128(1) cam(2)} }

cam-ecb192 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb192(2) cam(2)} }
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
cam-ecb256 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb256(3) cam(2)} }
```

See clause 8.2.2 for some considerations regarding the use of this mode.

### 8.3.3 Camellia – cipher block chaining (CAM-CBC) mode

The three CAM-CBC symmetric-key algorithms are defined below:

```
cam-cbc128 ALGORITHM ::= {
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc128(5) cam(2)} }

cam-cbc192 ALGORITHM ::= {
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc192(6) cam(2)} }

cam-cbc256 ALGORITHM ::= {
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc256(7) cam(2)} }

CAM-InitializationVector ::= OCTET STRING (SIZE (16))
```

The requirements on the CAM-InitializationVector are the same as specified for AES in clause 8.2.3.

### 8.3.4 Camellia – cipher feedback (CAM-CFB) mode

The three CAM-CFB symmetric-key algorithms are defined below:

```
cam-cfb128 ALGORITHM ::= {
  PARMS         CamNumberOfBits
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb128(13) cam(2)} }

cam-cfb192 ALGORITHM ::= {
  PARMS         CamNumberOfBits
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb192(14) cam(2)} }

cam-cfb256 ALGORITHM ::= {
  PARMS         CamNumberOfBits
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb256(15) cam(2)} }

CamNumberOfBits ::= INTEGER(1..128)
```

### 8.3.5 Camellia – output feedback (CAM-OFB) mode

The three CAM-OFB symmetric-key algorithms are defined below:

```
cam-ofb128 ALGORITHM ::= {
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb128(9) cam(2)} }

cam-ofb192 ALGORITHM ::= {
  DYN-PARMS     CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb192(10) cam(2)} }

cam-ofb256 ALGORITHM ::= {
  DYN-PARMS    CAM-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb256(11) cam(2)} }
```

### 8.3.6 Camellia – counter (CAM-CTR) mode

The three CAM-CTR symmetric-key algorithms are defined below:

```
cam-ctr128 ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  DYN-PARMS     CAM-CtrNonce
  IDENTIFIED BY {id-algo-ska ctr128(17) cam(2)} }

cam-ctr192 ALGORITHM ::= {
  DYN-PARMS     CAM-CtrNonce
  IDENTIFIED BY {id-algo-ska ctr192(18) cam(2)} }

cam-ctr256 ALGORITHM ::= {
  DYN-PARMS     CAM-CtrNonce
  IDENTIFIED BY {id-algo-ska ctr256(19) cam(2)} }

CAM-CtrNonce ::= OCTET STRING (SIZE (8))
```

The requirements for CAM-CtrNonce are the same as for CtrNonce in clause 8.2.6.

### 8.3.7 Camellia – Galois/counter mode (CAM-GCM)

The three CAM-GCM authenticated encryption with associated data (AEAD) algorithms are defined below:

```
cam-128gcm ALGORITHM ::= {
  PARMS         CAM-GCM-ICVlen
  DYN-PARMS     CAM-GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm128(21) cam(2)} }

cam-192gcm ALGORITHM ::= {
  PARMS         CAM-GCM-ICVlen
  DYN-PARMS     CAM-GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm192(22) cam(2)} }

cam-256gcm ALGORITHM ::= {
  PARMS         CAM-GCM-ICVlen
  DYN-PARMS     CAM-GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm256(23) cam(2)} }

CAM-GCM-ICVlen ::= INTEGER (16)
CAM-GCM-nonce  ::= OCTET STRING (SIZE (12))
```

### 8.3.8 Camellia – Galois/counter mode – synthetic initialization vector (CAM-GCM-SIV)

The three CAM-GCM-SIV authenticated encryption with associate data-key algorithms are defined below:

```
cam-128gcm-siv ALGORITHM ::= {
  PARMS         CAM-GCM-SIV-ICVlen
  DYN-PARMS     CAM-GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm128-siv(25) cam(2)} }

cam-192gcm-siv ALGORITHM ::= {
  PARMS         CAM-GCM-SIV-ICVlen
  DYN-PARMS     CAM-GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm192-siv(26) cam(2)} }

cam-256gcm-siv ALGORITHM ::= {
  PARMS         CAM-GCM-SIV-ICVlen
  DYN-PARMS     CAM-GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm256-siv(27) cam(2)} }

CAM-GCM-SIV-ICVlen ::= INTEGER (16)
CAM-GCM-SIV-nonce  ::= OCTET STRING (SIZE (12))
```

### 8.3.9 Camellia – Counter with CBC-MAC (CAM-CCM)

The three CAM-CCM ICV algorithms are defined below.

```
cam-128ccm ALGORITHM ::= {
  PARMS         CAM-CCM-ICVlen
  DYN-PARMS     CAM-CCM-nonce
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  IDENTIFIED BY {id-algo-ska ccm128(29) cam(2)} }

cam-192ccm ALGORITHM ::= {
  PARMS          CAM-CCM-ICVlen
  DYN-PARMS      CAM-CCM-nonce
  IDENTIFIED BY {id-algo-ska ccm192(30) cam(2)} }

cam-256ccm ALGORITHM ::= {
  PARMS          CAM-CCM-ICVlen
  DYN-PARMS      CAM-CCM-nonce
  IDENTIFIED BY {id-algo-ska ccm256(31) cam(2)} }

CAM-CCM-ICVlen ::= INTEGER (16)
CAM-CCM-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.3.10 Camellia – Galois Message Authentication Code (CAM-GMAC)

The three CAM-GMAC ICV algorithms are defined below.

```
cam-128gmac ALGORITHM ::= {
  PARMS          CAM-MACLength
  DYN-PARMS      CAM-GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac128(33) cam(2)} }

cam-192gmac ALGORITHM ::= {
  PARMS          CAM-MACLength
  DYN-PARMS      CAM-GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac192(34) cam(2)} }

cam-256gmac ALGORITHM ::= {
  PARMS          CAM-MACLength
  DYN-PARMS      CAM-GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac256(35) cam(2)} }

CAM-MACLength  ::= INTEGER (12 | 13 | 14 | 15 | 16)
CAM-GMAC-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.3.11 Camellia – Cipher-based message authentication code (CAM-CMAC)

The three CAM-CMAC ICV algorithms are defined below.

```
cam-128cmac ALGORITHM ::= {
  PARMS          CAM-CMACLength
  IDENTIFIED BY {id-algo-ska cmac128(37) cam(2)} }

cam-192cmac ALGORITHM ::= {
  PARMS          CAM-CMACLength
  IDENTIFIED BY {id-algo-ska cmac192(38) cam(2)} }

cam-256cmac ALGORITHM ::= {
  PARMS          CAM-CMACLength
  IDENTIFIED BY {id-algo-ska cmac256(39)cam(2) } }

CAM-CMACLength ::= INTEGER
```

### 8.4 SEED – symmetric-key algorithms

#### 8.4.1 General

The SEED encryption technique is specified in ISO/IEC 18033-3, but only for the encryption of a single block. It is only defined for a key size of 128 bits. The block size is 128 bits.

#### 8.4.2 SEED – electronic codebook (SEED-ECB) mode

The SEED-ECB symmetric-key algorithm is defined below:


© ISO/IEC 2025 – All rights reserved

Full document details: BS ISO/IEC 9594-11:2025

```
seed-ecb128 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb128(1) seed(3)} }
```

See clause 8.2.2 for some consideration for using this mode.

### 8.4.3 SEED – cipher block chaining (SEED-CBC) mode

The SEED-CBC symmetric-key algorithm is defined below:

```
seed-cbc128 ALGORITHM ::= {
  DYN-PARMS     SEED-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc128(5) seed(3)} }

SEED-InitializationVector ::= OCTET STRING (SIZE (16))
```

The requirements on the SEED-InitializationVector are the same as specified for AES in clause 8.2.3.

### 8.4.4 SEED – cipher feedback (SEED-CFB) mode

The SEED-CFB symmetric-key algorithm is defined below:

```
seed-cfb128 ALGORITHM ::= {
  PARMS         SEED-NumberOfBits
  DYN-PARMS     SEED-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb128(13) seed(3)} }

SEED-NumberOfBits ::= INTEGER(1..128)
```

### 8.4.5 SEED – output feedback (SEED-OFB) mode

The SEED-OFB symmetric-key algorithm is defined below:

```
seed-ofb128 ALGORITHM ::= {
  DYN-PARMS     SEED-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb128(9) seed(3)} }
```

### 8.4.6 SEED – counter (SEED-CTR) mode

The SEED-CTR symmetric-key algorithm is defined below:

```
seed-ctr128 ALGORITHM ::= {
  DYN-PARMS     SEED-CtrNonce
  IDENTIFIED BY {id-algo-ska ctr128(17) seed(3)} }

SEED-CtrNonce ::= OCTET STRING (SIZE (8))
```

The requirements for SEED-CtrNonce are the same as for CtrNonce in clause 8.2.6.

### 8.4.7 SEED – Galois/counter mode (SEED-GCM)

The SEED-GCM authenticated encryption with associated data (AEAD) algorithm is defined below:

```
seed-128gcm ALGORITHM ::= {
  PARMS         SEED-GCM-ICVlen
  DYN-PARMS     SEED-GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm128(21) seed(3)} }

SEED-GCM-ICVlen ::= INTEGER (16)
SEED-GCM-nonce  ::= OCTET STRING (SIZE (12))
```

### 8.4.8 SEED – Galois/counter mode – synthetic initialization vector (SEED-GCM-SIV)

The SEED-GCM-SIV authenticated encryption with associate data-key algorithm is defined below:

```
seed-128gcm-siv ALGORITHM ::= {
  PARMS         SEED-GCM-SIV-ICVlen
  DYN-PARMS     SEED-GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm128-siv(25) seed(3)} }
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
SEED-GCM-SIV-ICVlen ::= INTEGER (16)
SEED-GCM-SIV-nonce  ::= OCTET STRING (SIZE (12))
```

#### 8.4.9 SEED – Counter with CBC-MAC (SEED-CCM)

The SEED-CCM ICV algorithm is defined below:

```
seed-128ccm ALGORITHM ::= {
  PARMS          SEED-CCM-ICVlen
  DYN-PARMS      SEED-CCM-nonce
  IDENTIFIED BY {id-algo-ska ccm128(29) seed(3)} }

SEED-CCM-ICVlen ::= INTEGER (16)
SEED-CCM-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.4.10 SEED – Galois Message Authentication Code (SEED-GMAC)

The SEED-GMAC ICV algorithm is defined below:

```
seed-128gmac ALGORITHM ::= {
  PARMS          SEED-MACLength
  DYN-PARMS      SEED-GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac128(33) seed(3)} }

SEED-MACLength  ::= INTEGER (12 | 13 | 14 | 15 | 16)
SEED-GMAC-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.4.11 SEED – Cipher-based message authentication code (SEED-CMAC)

The SEED-CMAC ICV algorithm is defined below:

```
seed-128cmac ALGORITHM ::= {
  PARMS          SEED-CMACLength
  IDENTIFIED BY {id-algo-ska cmac128(37) seed(3)} }

SEED-CMACLength ::= INTEGER
```

### 8.5 SM4 – symmetric-key algorithms

#### 8.5.1 General

The SM4 encryption technique is specified in ISO/IEC 18033-3, but only for the encryption of a single block. It is only defined for a key size of 128 bits. The block size is 128 bits.

#### 8.5.2 SM4 – electronic codebook (SM4-ECB) mode

The SM4-ECB symmetric-key algorithm is defined below:

```
sm4-ecb128 ALGORITHM ::= {
  IDENTIFIED BY {id-algo-ska ecb128(1) sm4(4)} }
```

See clause 8.2.2 for some consideration for using this mode.

#### 8.5.3 SM4 – cipher block chaining (SM4-CBC) mode

The SM4-CBC symmetric-key algorithm is defined below:

```
sm4-cbc128 ALGORITHM ::= {
  DYN-PARMS      SM4-InitializationVector
  IDENTIFIED BY {id-algo-ska cbc128(5) sm4(4)} }

SM4-InitializationVector ::= OCTET STRING (SIZE (16))
```

#### 8.5.4 SM4 – cipher feedback (SM4-CFB) mode

The SM4-CFB symmetric-key algorithm is defined below:

```
sm4-cfb128 ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  PARMS          SM4-NumberOfBits
  DYN-PARMS      SM4-InitializationVector
  IDENTIFIED BY {id-algo-ska cfb128(13) sm4(4)} }

SM4-NumberOfBits ::= INTEGER(1..128)
```

### 8.5.5 SM4 – output feedback (SM4-OFB) mode

The SM4-OFB symmetric-key algorithm is defined below:

```
sm4-ofb128 ALGORITHM ::= {
  DYN-PARMS      SM4-InitializationVector
  IDENTIFIED BY {id-algo-ska ofb128(9) sm4(4)} }
```

### 8.5.6 SM4 – counter (SM4-CTR) mode

The SM4-CTR symmetric-key algorithm is defined below:

```
sm4-ctr128 ALGORITHM ::= {
  DYN-PARMS      SM4-CtrNonce
  IDENTIFIED BY {id-algo-ska ctr128(17) sm4(4)} }

SM4-CtrNonce ::= OCTET STRING (SIZE (8))
```

The requirements for SEED-CtrNonce are the same as for CtrNonce in clause 8.2.6.

### 8.5.7 SM4 – Galois/counter mode (SM4-GCM)

The SM4-GCM algorithm is defined below:

```
sm4-128gcm ALGORITHM ::= {
  PARMS          SM4-GCM-ICVlen
  DYN-PARMS      SM4-GCM-nonce
  IDENTIFIED BY {id-algo-ska gcm128(21) sm4(4)} }

SM4-GCM-ICVlen ::= INTEGER (16)
SM4-GCM-nonce  ::= OCTET STRING (SIZE (12))
```

### 8.5.8 SM4 – Galois/counter mode – synthetic initialization vector (SM4-GCM-SIV)

The SM4-GCM-SIV authenticated encryption with associated data (AEAD) algorithm is defined below:

```
sm4-128gcm-siv ALGORITHM ::= {
  PARMS          SM4-GCM-SIV-ICVlen
  DYN-PARMS      SM4-GCM-SIV-nonce
  IDENTIFIED BY {id-algo-ska gcm128-siv(25) sm4(4)} }

SM4-GCM-SIV-ICVlen ::= INTEGER (16)
SM4-GCM-SIV-nonce  ::= OCTET STRING (SIZE (12))
```

### 8.5.9 SM4 – Counter with CBC-MAC (SM4-CCM)

The SM4-CCM ICV algorithm is defined below.

```
sm4-128ccm ALGORITHM ::= {
  PARMS          SM4-CCM-ICVlen
  DYN-PARMS      SM4-CCM-nonce
  IDENTIFIED BY {id-algo-ska ccm128(29) sm4(4)} }

SM4-CCM-ICVlen ::= INTEGER (16)
SM4-CCM-nonce ::= OCTET STRING (SIZE (12))
```

### 8.5.10 SM4 – Galois Message Authentication Code (SM4-GMAC)

The SM4-GMAC ICV algorithm is defined below.

```
sm4-128gmac ALGORITHM ::= {
  PARMS          SM4-MACLength
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  DYN-PARMS      SM4-GMAC-nonce
  IDENTIFIED BY {id-algo-ska gmac128(33) sm4(4)} }

SM4-MACLength  ::= INTEGER (12 | 13 | 14 | 15 | 16)
SM4-GMAC-nonce ::= OCTET STRING (SIZE (12))
```

#### 8.5.11 SM4 – Cipher-based message authentication code (SM4-CMAC)

The SM4-CMAC ICV algorithm is defined below.

```
sm4-128cmac ALGORITHM ::= {
  PARMS          SM4-CMACLength
  IDENTIFIED BY {id-algo-ska cmac128(37) sm4(4)} }

SM4-CMACLength ::= INTEGER
```

## 9 Public-key and digital signature algorithms

None of the current public-key and digital signature algorithms have dynamic parameters. This is a place holder for possible new public-key and digital signature algorithms requiring an allocation of object identifiers.

## 10 Key establishment algorithms

### 10.1 General

Key establishment methods are used for establishing symmetric keys directly between two communicating entities in a secure way. The Diffie-Hellman technology, as specified in IETF RFC 2631, is an important example. The result of a Diffie-Hellman operation is a shared secret that a key derivation technology expands to generate the required symmetric keys.

Diffie-Helman and the key derivation technologies in their basic form are not well suited for the cryptographic algorithm migration techniques as described in Annex H. By defining different Diffie-Hellman and key derivation combinations as cryptographic algorithms using the **ALGORITHM** information object class, as it is specified in clause 7.2.2. In the following some key establishment information objects are defined.

### 10.2 Diffie-Hellman over prime field

#### 10.2.1 General

As discussed in Rec. ITU-T X.508 | ISO/IEC 9594-12, Diffie-Hellman methods may be performed over a finite field which forms a group. IETF RFC 3526 lists groups, and for each group it selects a generator. When selecting parameters to be used for generating a shared secret, it is only necessary to point to the group to be used.

#### 10.2.2 Diffie-Hellman group 14 algorithm with HKDF-256

The following is a specification for key establishment algorithm based on the Diffie-Hellman key agreement method.

```
dhModpGr14Hkdf256Algo ALGORITHM ::= {
  PARMS          Group14
  DYN-PARMS      Payload14
  IDENTIFIED BY id-algo-dhModpGr14Hkdf256Algo }

Group14 ::= INTEGER (14)

Payload14 ::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (256)),
  nonce       OCTET STRING (SIZE (32)) OPTIONAL,
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 14 for the 2048-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad14** data type.

The **PayLoad14** data type has the following components:


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

a) The **dhPublicKey** component shall specify the length in octets of the Diffie-Hellman public key to be used by the sender. For the used group, the size is always 256 octets.

c) The **nonce** component shall specify the length in octets of a random value to be used by the shared key derivation specified in clause 10.4.2. A new value shall be generated for each key establishment. The length is recommended to be length of the hash output. As the **sha256** algorithm is used for key derivation, the length shall be 32 octets.

From the established shared secret, the two partners shall use the HMAC-based Extract-and-Expand Key Derivation Function (HKDF) to create symmetric keys according to the requirement established by the encryption and ICV requirements.

The **hmacWithSHA256** algorithm shall be used for the HKDF derivation.

### 10.2.3 Diffie-Hellman group 15 algorithm with HKDF-384

```
dhModpGr15Hkdf384Algo ALGORITHM ::= {
  PARMS          Group15
  DYN-PARMS      Payload15
  IDENTIFIED BY id-algo-dhModpGr15Hkdf384Algo }

Group15 ::= INTEGER (15)

Payload15 ::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (384)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 15 for the 3072-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad15** data type.

The **PayLoad15** data type has the same components as for the **PayLoad14** data type as specified in clause 10.2.2 except that the Diffie-Hellman public key size is 384 octets.

### 10.2.4 Diffie-Hellman group 16 algorithm with HKDF-512

```
dhModpGr16Hkdf512Algo ALGORITHM ::= {
  PARMS          Group16
  DYN-PARMS      Payload16
  IDENTIFIED BY id-algo-dhModpGr16Hkdf512Algo }

Group16 ::= INTEGER (16)

Payload16 ::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (512)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 16 for the 4096-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad16** data type.

The **PayLoad16** data type has the same components as for the **PayLoad14** data type as specified in clause 10.2.2 except that the Diffie-Hellman public key size is 512 octets.

### 10.2.5 Diffie-Hellman group 17 algorithm with HKDF-768

```
dhModpGr17Hkdf768Algo ALGORITHM ::= {
  PARMS          Group17
  DYN-PARMS      Payload17
  IDENTIFIED BY id-algo-dhModpGr17Hkdf7686Algo }

Group17 ::= INTEGER (17)

Payload17 ::= SEQUENCE {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025

```
  dhPublicKey OCTET STRING (SIZE (768)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 17 for the 6144-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad16** data type.

The **PayLoad17** data type has the same components as for the **PayLoad14** data type as specified in clause 10.2.2 except that the Diffie-Hellman public key size is 768 octets.

#### 10.2.6 Diffie-Hellman group 18 algorithm with HKDF-1024

```
dhModpGr18Hkdf1024Algo ALGORITHM ::= {
  PARMS         Group18
  DYN-PARMS     Payload18
  IDENTIFIED BY id-algo-dhModpGr18Hkdf1024Algo }

Group18 ::= INTEGER (18)

Payload18 ::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (1024)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 18 for the 8192-bit MODP as specified in IETF RFC 3526.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad18** data type.

The **PayLoad18** data type has the same components as for the **PayLoad14** data type as specified in clause 10.2.2 except that the Diffie-Hellman public key size is 1024 octets.

### 10.3 Elliptic curve Diffie-Hellman

#### 10.3.1 General

The Diffie-Hellman method may be performed over a group as specified in IETF RFC 5114, where all the parameters for a given curve are established by a group specification. Different groups are defined for different curves. When selecting parameters to be used for generating a shared secret, it is only necessary to point to the group to be used.

#### 10.3.2 Diffie-Hellman group 23 algorithm with HKDF-256

```
dhModpGr23Hkdf256Algo ALGORITHM ::= {
  PARMS         Group23
  DYN-PARMS     Payload23
  IDENTIFIED BY id-algo-dhModpGr23Hkdf256Algo }

Group23 ::= INTEGER (23)

Payload23 ::= SEQUENCE {
  dhPublicKey OCTET STRING (SIZE (65)),
  nonce       OCTET STRING (SIZE (32)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 23, for the Elliptic curve Diffie-Hellman (ECDH) curve **secp256r1** as specified in IETF RFC 5114.

The **DYN-PARMS** token that the dynamic parameters shall be those specified by the **PayLoad23** data type.

The **PayLoad23** data type has the same components as the **PayLoad14** data type except that the **dhPublicKey** shall have length of 65 octets.

The key derivation shall be as specified in clause 10.4.

#### 10.3.3 Diffie-Hellman group 28 algorithm with HKDF-256

```
dhModpGr28Hkdf256Algo ALGORITHM ::= {
```


© ISO/IEC 2025 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2025