BS ISO/IEC 9594-11:2020

BSI Standards Publication

Information technology — Open systems interconnection directory

Part 11: Protocol specifications for secure operations

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2020

National foreword

This British Standard is the UK implementation of ISO/IEC 9594-11:2020.

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.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2021
Published by BSI Standards Limited 2021

ISBN 978 0 580 99878 2

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 31 January 2021.

Amendments/corrigenda issued since publication

Date	Text affected

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2020

INTERNATIONAL STANDARD

ISO/IEC 9594-11

First edition
2020-12

Information technology — Open systems interconnection directory —

Part 11: Protocol specifications for secure operations

Reference number
ISO/IEC 9594-11:2020(E)

Documents
Document Sharing Hub
© ISO/IEC 2020

Full document details: BS ISO/IEC 9594-11:2020

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2020

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

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2020

INTERNATIONAL STANDARD ISO/IEC 9594-11
RECOMMENDATION ITU-T X.510

Information technology – Open Systems Interconnection – The Directory: Protocol specifications for secure operations

Summary

Recommendation ITU-T X.510 | ISO/IEC 9594-11 specifies a general protocol, called the wrapper protocol, that provides cybersecurity for protocols designed for its protection. The wrapper protocol provides authentication, integrity and optionally confidentiality (encryption). The wrapper protocol allows cybersecurity to be provided independently of the protected protocols, which means that security may be enhanced without affecting protected protocol specifications.

The wrapper protocol is specified without specifying specific cryptographic algorithms, but is designed for plucking-in cryptographic algorithms as required.

The wrapper protocol is designed for easy migration of cryptographic algorithms, as stronger cryptographic algorithms become necessary.

Recommendation ITU-T X.510 | ISO/IEC 9594-11 contains recommendations for how other Recommendations and International Standards may include features for migration of cryptographic algorithms, and it includes ASN.1 specifications to be applied for that purpose.

Recommendation ITU-T X.510 | ISO/IEC 9594-11 also specifies three protocols that make use of the wrapper protocol protection. This includes a protocol for maintenance of authorization and validation lists (AVLs), a protocol for subscribing of public-key certificate status and a protocol for accessing a trust broker.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T X.510	2020-08-22	17	11.1002/1000/14320

Keywords

Certification authority, cryptography, cryptographic algorithm, digital signature, public-key certificate, PKI, quantum-safe, trust anchor, validation.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2020

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

© ISO/IEC 2020 – All rights reserved

Documents Document Sharing Hub

Full document details: BS ISO/IEC 9594-11:2020

CONTENTS

Page

SECTION 1 – GENERAL 1
1 Scope 1
2 Normative references 1
2.1 Identical Recommendations | International Standards 1
2.2 Other references 1
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 specified by this Recommendation | International Standard 2
4 Abbreviations 3
5 Conventions 4
6 Common data types and special cryptographic algorithms 4
6.1 Introduction 4
6.2 ASN.1 information object class specification tool 4
6.3 Multiple-cryptographic algorithm specifications 6
6.4 Key establishment algorithms 7
6.5 Multiple-cryptographic algorithm-value pairs 9
6.6 Formal specification of encipherment 11
7 General concepts for securing protocols 11
7.1 Introduction 11
7.2 Protected protocol plug-in concept 12
7.3 Communications structure 12
7.4 Another view of the relationship between the wrapper protocol and the protected protocol 12
7.5 Structure of application protocol data unit 13
7.6 Exception conditions 13
SECTION 2 – THE WRAPPER PROTOCOL 14
8 Wrapper protocol general concepts 14
8.1 Introduction 14
8.2 UTC time specification 14
8.3 Use of alternative cryptographic algorithms 14
8.4 General on establishing shared keys 14
8.5 Sequence numbers 15
8.6 Use of invocation identification in the wrapper protocol 15
8.7 Mapping to underlying services 15
8.8 Definition of protected protocols 15
8.9 Overview of wrapper protocol data units 15
9 Association management 16
9.1 Introduction to association management 16
9.2 Association handshake request 16
9.3 Association accept 18
9.4 Association reject due to security issues 19
9.5 Association reject by the protected protocol 20
9.6 Handshake security abort 21
9.7 Handshake abort by protected protocol 21
9.8 Data transfer security abort 22
9.9 Abort by protected protocol 22
9.10 Release request WrPDU 23
9.11 Release response WrPDU 23
9.12 Release collision 24

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

Page

10 Data transfer phase ... 24
10.1 Symmetric keys renewal ... 24
10.2 Data transfer by the client ... 24
10.3 Data transfer by the server ... 26
11 Information flow ... 28
11.1 Purpose and general model ... 28
11.2 Protected protocol SAOC ... 29
11.3 Wrapper SAOC ... 29
12 Wrapper error handling ... 32
12.1 General ... 32
12.2 Checking of a wrapper handshake request ... 32
12.3 Checking of a wrapper handshake accept ... 33
12.4 Checking of data transfer WrPDUs ... 34
12.5 Wrapper diagnostic codes ... 36
SECTION 3 – PROTECTED PROTOCOLS ... 37
13 Authorization and validation list management ... 37
13.1 General on authorization and validation management ... 37
13.2 Defined protected protocol data unit (PrPDU) types ... 37
13.3 Authorization and validation management protocol initialization request ... 38
13.4 Authorization and validation management protocol initialization accept9 ... 38
13.5 Authorization and validation management protocol initialization reject ... 38
13.6 Authorization and validation management protocol initialization abort ... 38
13.7 Add authorization and validation list request ... 39
13.8 Add authorization and validation list response ... 40
13.9 Replace authorization and validation list request ... 40
13.10 Replace authorization and validation list response ... 40
13.11 Delete authorization and validation list request ... 41
13.12 Delete authorization and validation list response ... 41
13.13 Authorization and validation list abort ... 42
13.14 Authorization and validation list error codes ... 42
14 Certification authority subscription protocol ... 43
14.1 Certification authority subscription introduction ... 43
14.2 Defined protected protocol data unit (PrPDU) types ... 43
14.3 Certification authority subscription protocol initialization request ... 43
14.4 Certification authority subscription protocol initialization accept ... 44
14.5 Certification authority subscription protocol initialization reject ... 44
14.6 Certification authority subscription protocol initialization abort ... 44
14.7 Public-key certificate subscription request ... 44
14.8 Public-key certificate subscription response ... 45
14.9 Public-key certificate un-subscription request ... 46
14.10 Public-key certificate un-subscription response ... 46
14.11 Public-key certificate replacements request ... 47
14.12 Public-key certificate replacement response ... 48
14.13 End-entity public-key certificate updates request ... 49
14.14 End-entity public-key certificate updates response ... 49
14.15 Certification authority subscription abort ... 50
14.16 Certification authority subscription error codes ... 50
15 Trust broker protocol ... 51
15.1 Introduction ... 51
15.2 Defined protected protocol data unit (PrPDU) types ... 51
15.3 Trust broker protocol initialization request ... 51
15.4 Trust broker protocol initialization accept ... 52
15.5 Trust broker protocol initialization reject ... 52

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

Page
15.6 Trust broker protocol initialization abort 52
15.7 Trust broker request syntax 52
15.8 Trust broker response syntax 53
15.9 Trust broker error information 53
Annex A − Crypto Tools in ASN.1 55
Annex B − Wrapper protocol in ASN.1 58
Annex C − Protected protocol interface to the wrapper protocol 63
Annex D − Cryptographic algorithms 65
Annex E − Authorization and validation list management in ASN.1 67
Annex F − Certification authority subscription in ASN.1 70
Annex G −Trust broker in ASN.1 74
Annex H − Migration of cryptographic algorithms 76
H.1 Introduction 76
H.2 Negotiation of cryptographic algorithms 76
H.3 Non-negotiable digital signature algorithms 77
Annex I − Auxiliary specifications 80
Bibliography 85

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

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 the ASN.1 module for the wrapper protocol.

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

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

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.

The content of this Rec. ITU-T X.510 | ISO/IEC 9594-11 was moved to here from Rec. ITU-T X.509 (2016) | ISO/IEC 9594-8:2017 and subsequently updated.

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

INTERNATIONAL STANDARD ISO/IEC 9594-11
RECOMMENDATION ITU-T X.510

Information technology – Open Systems Interconnection – The Directory: Protocol specifications for secure operations

SECTION 1 – GENERAL

1 Scope

The scope of this Recommendation | International Standard is threefold.

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.500 (2019) | ISO/IEC 9594-1:2020, *Information technology - Open Systems Interconnection - The Directory: Overview of concepts, models and services.*
- 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.511 (2019) | ISO/IEC 9594-3:2020, *Information technology - Open Systems Interconnection - The Directory: Abstract service definition*.
- Recommendation ITU-T X.518 (2019) | ISO/IEC 9594-4:2020, *Information technology - Open Systems Interconnection - The Directory: Procedures for distributed operation*.
- Recommendation ITU-T X.519 (2019) | ISO/IEC 9594-5:2020, *Information technology - Open Systems Interconnection - The Directory: Protocol specifications*.
- Recommendation ITU-T X.520 (2019) | ISO/IEC 9594-6:2020, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types*.
- Recommendation ITU-T X.521 (2019) | ISO/IEC 9594-7:2020, *Information technology - Open Systems Interconnection - The Directory: Selected object classes*.
- Recommendation ITU-T X.525 (2019) | ISO/IEC 9594-9:2020, *Information technology - Open Systems Interconnection - The Directory: Replication*.
- Recommendation ITU-T X.680 (2015)| ISO/IEC 8824-1:2015, *Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation*.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

– Recommendation ITU-T X.681 (2015) | ISO/IEC 8824-2:2015, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification*.

– Recommendation ITU-T X.682 (2015) | ISO/IEC 8824-3:2015, *Information technology - Abstract Syntax Notation One (ASN.1): Constraint specification*.

– Recommendation ITU-T X.683 (2015) | ISO/IEC 8824-4:2015, *Information technology - Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.

– Recommendation ITU-T X.690 (2015) | ISO/IEC 8825-1:2015, *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 (2015) | ISO/IEC 8825-2:2015, *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*.

2.3 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 terms are defined in Rec. ITU-T X.501 | ISO/IEC 9594-2:

a) attribute;

b) distinguished name (of an entry).

3.3 Public-key and attribute certificate definitions

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

a) authorization and validation list (AVL);

b) authorization and validation list entity (AVL entity);

c) authorizer;

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

d) certification authority (CA);
e) certification path;
f) end entity;
g) end-entity public-key certificate;
h) hash function;
i) key agreement;
j) private key;
k) public key;
l) public-key certificate;
m) public-key infrastructure (PKI);
n) relying party;
o) trust broker.

3.4 Terms defined in this Recommendation | International Standard

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 original 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 OSI term (see Rec. ITU-T X.519 | ISO/IEC 9594-5), 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 client: The entity that initiates an association.

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

3.4.8 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.

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

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

3.4.11 server: The entity that accepts or rejects an association.

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

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

4 Abbreviations

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

AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
AES-CBC	Advanced Encryption Standard-Cipher Block Chaining
APDU	Application Protocol Data Unit
ASN.1	Abstract Syntax Notation One

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

AVL	Authorization and Validation List
AVMP	Authorization and Validation Management Protocol
BER	Basic Encoding Rules
CA	Certification Authority
CASP	Certification Authority Subscription Protocol
DER	Distinguished Encoding Rules
DH	Diffie-Hellman
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
MODP	Modular exponential
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
TCP	Transmission Control Protocol
UTC	Coordinated Universal Time
WrPDU	Wrapper Protocol Data Unit

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 Common data types and special cryptographic algorithms

6.1 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 6.2.

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`**

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

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 6.3.

c) There are advantages to the definition of cryptographic algorithms for key agreement procedures. This has been done for a few cases in clause 6.4.

d) Parameterized data types are defined in clause 6.5 for flexible protocol design.

e) Some formal specifications for encipherment are given in clause 6.6.

6.2 ASN.1 information object class specification tool

6.2.1 General information object class concept

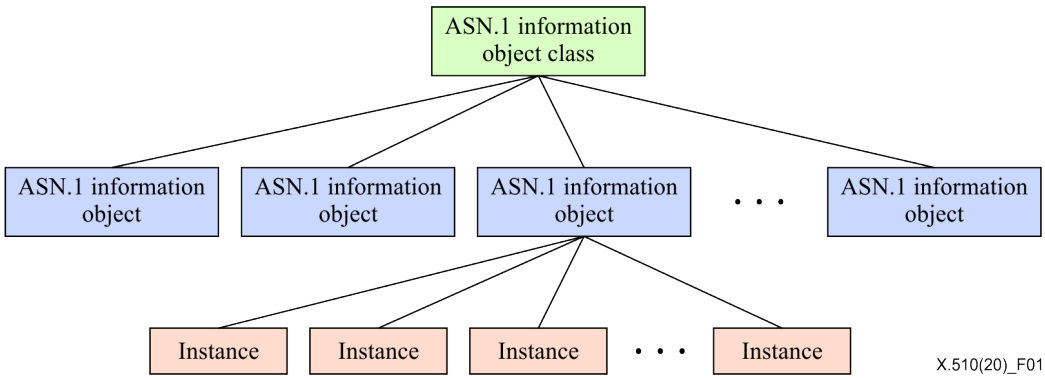

Figure 1 – ASN.1 information object class concept

The concept of ASN.1 information object class is specified in Rec. ITU-T X.681 | ISO/IEC 8824-2. It is 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 Recommendations | ISO/IEC 9594-all parts for defining attributes, matching rules, extensions to public-key and attribute certificates, etc.

Figure 1 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 are defined, e.g., an attribute type for e-mail addresses. From this specification, instances of e-mail address attributes may be generated. Instances may be transferred in the protocol or may be stored in a directory.

NOTE – The concept of object classes used in clause 11 is somewhat different from the concept of information object class defined in Rec. ITU-T X.681 | ISO/IEC 8824-2.

6.2.2 The ALGORITHM information object class

The information object class concept is also used to define cryptographic algorithms. The **ALGORITHM** information object class is defined in 6.2.2 of Rec. ITU-T X.509 | ISO/IEC 9594-8. The specification of this information is reproduced as follows 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 to specify 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 deploying the cryptographic algorithm being defined. Not all cryptographic algorithms

© ISO/IEC 2020 – All rights reserved

Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

require such parameters. The field is then absent or has the value **NULL**, as determined by the individual cryptographic algorithm specifications.

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.

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

The **AlgorithmWithInvoke** parameterized data type defined as follows 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 **AlgorithmWithIvoke** 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})
```

6.3 Multiple-cryptographic algorithm specifications

6.3.1 General

Multiple cryptographic algorithms of the same object class may be specified using a single outer algorithm and then used instead of a single algorithm as a tool for algorithm migration as discussed in Annex H.

The **PARMS** 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.

6.3.2 Multiple signatures algorithm

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

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

MultipleSignaturesAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo  AlgorithmIdentifier{{SupportedSignatureAlgorithms}}
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

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

6.3.3 Multiple symmetric key algorithm

The following is a specification of an **ALGORITHM** information object that allows multiple symmetric key algorithms to be specified.

```
multipleSymmetricKeyAlgo ALGORITHM ::= {
  PARMS         MultipleSymmetricKeyAlgo
  IDENTIFIED BY id-algo-multipleSymmetricKeyAlgo }

MultipleSymmetricKeyAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo  AlgorithmIdentifier{{SupportedSymmetricKeyAlgorithms}}

SupportedSymmetricKeyAlgorithms ALGORITHM ::= {...}
```

6.3.4 Multiple public-key algorithms

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

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

MultiplePublicKeyAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo  AlgorithmIdentifier{{SupportedPublicKeyAlgorithms}}

SupportedPublicKeyAlgorithms ALGORITHM ::= {...}
```

6.3.5 Multiple hash algorithm

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

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

MultipleHashAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo  AlgorithmIdentifier{{SupportedHashAlgorithms}}

SupportedHashAlgorithms ALGORITHM ::= {...}
```

6.3.6 Multiple authenticated encryption with associated data algorithm

The following is a specification of an **ALGORITHM** information object that allows multiple authenticated encryption with associated data (AEAD) algorithms to be specified.

```
multipleAuthenEncryptAlgo ALGORITHM ::= {
  PARMS         MultipleAuthenEncryptAlgo
  IDENTIFIED BY id-algo-multipleAuthenEncryptAlgo }

MultipleAuthenEncryptAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo       AlgorithmIdentifier{{SupportedAuthenEncryptAlgorithms}}

SupportedAuthenEncryptAlgorithms ALGORITHM ::= {...}
```

6.3.7 Multiple integrity check value algorithm

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

```
multipleIcvAlgo ALGORITHM ::= {
  PARMS         MultipleIcvAlgo
  IDENTIFIED BY id-algo-multipleIcvAlgo }

MultipleIcvAlgo ::= SEQUENCE SIZE (1..MAX) OF
  algo  AlgorithmIdentifier{{SupportedIcvAlgorithms}}

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

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

6.4 Key establishment algorithms

6.4.1 General

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

DH and the key derivation technologies in their basic form are not well suited to the cryptographic algorithm migration techniques described in Annex H. By defining different combination of DH and key derivation combinations as cryptographic algorithms using the **ALGORITHM** information object class, as it specified in clause 6.2.2, it is possible to establish a migration path. In the following, some key establishment information objects are defined.

Clause 6.4 is provided for the purpose of validating the wrapper protocol as a whole using DH technology, as specified in IETF RFC 2631. The choice of the cryptographic mechanism for the shared key generation in the wrapper protocol is made by end-users in accordance with the principles of national regulations.

6.4.2 Diffie-Hellman group 14 algorithm with HKDF-256

The following is a specification for key establishment algorithm based on the DH key agreement technique.

```
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)),
  ... }
```

The **PARMS** token specifies that the fixed parameters shall be those parameters specified for group number 14 for the 2048-bit modular exponential (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.

a) The **dhPublicKey** component shall hold the DH public key to be used by the sender. For the group used, the length of the key is always 256 octets. A different DH private and public key pair shall be generated for each new association establishment. For the duration of the association, the DH public key of the server shall be retained by both parties and the server shall retain its DH private key for later renewal of symmetric keys (see clause 10.2).

b) For each key renewal, the client shall generate a new DH private and DH public key pair and use it for the key renewal process.

c) The **nonce** component shall hold a random value to be used by the shared key derivation specified in clause 6.4.5. A new value shall be generated for each key establishment. The length is recommended to be that of the hash output. As the **sha256** algorithms 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 negotiation.

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

6.4.3 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 (512)),
  nonce       OCTET STRING (SIZE (32)),
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

```
  ... }
```

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

The **DYN-PARMS** token specifies 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 a length of 512 octets.

The key derivation shall be as specified in clause 6.4.5.

6.4.4 Diffie-Hellman group 28 algorithm with HKDF-256

```
dhModpGr28Hkdf256Algo ALGORITHM ::= {
  PARMS          Group28
  DYN-PARMS      Payload28
  IDENTIFIED BY id-algo-dhModpGr28Hkdf256Algo }

Group28 ::= INTEGER (28)

Payload28 ::= 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 28, for the ECDH curve **sbrainpoolP256r1** as specified in IETF RFC 6932.

The **DYN-PARMS** token specifies that the dynamic parameters shall be those specified by the **PayLoad28** data type.

The **PayLoad28** data type has the same components as the **PayLoad14** data type, except that the **dhPublicKey** shall have a length of 512 octets.

The key derivation shall be as specified in clause 6.4.5.

6.4.5 Key derivation

6.4.5.1 General

When the outcome of a key agreement algorithm is a shared secret, this shared secret shall be expanded by use of a key derivation function to provide sufficient material for the required symmetric keys.

6.4.5.2 HMAC-based extract-and-expand key derivation function

The HKDF is specified in IETF RFC 5869. It requires the use of the keyed-hash message authentication code (HMAC) algorithm as specified in IETF RFC 2104.

The HKDF consists of an HKDF-extract, resulting in a so-called pseudorandom key (PRK) and of an HKDF-expand, resulting in the output keying material (OKM).

The HKDF-extract may be expressed as follows.

PRK = HMAC-Hash(salt, IKM) where:

a) **salt** is a non-secret random value – it shall take the value of the **nonce** component of the key establishment algorithm instance in question that require the use of HKDF;

b) **IKM** stands for input keying material – it shall be the shared secret resulting from the DH key agreement;

c) The **OKM** is defined as:

OKM = HKDF-expand (PRK, info, L), where:

– **PRK** shall be the **PRK** generated by the **HKDF-expand** above,

– **info** shall be a zero-length string,

– **L** shall be the combined length of the keys to be generated;

If two keys are to be generated:

– the first part of the **OKM** is the key to be used by the client,

– the remaining part is the key to be used by the server;

If four keys are to be generated:

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

– the first key is the key used to generate an ICV by the client,
– the second key is the key used to generate an ICV by the server,
– the third key is the key used for encryption by the client,
– the fourth key is the key used for encryption by the server.

6.4.6 Special conditions

NOTE – For authenticated key agreement protocols, the absence of an explicit key validation phase [should] be compensated for by mixing up the certificates of both parties when generating the shared key. Such mixing up is not carried out in this protocol, which may lead to undesired consequences.

6.5 Multiple-cryptographic algorithm-value pairs

6.5.1 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.

6.5.2 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.

6.5.3 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.

```
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 } )
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

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 6.2.2):

– 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.

6.6 Formal specification of encipherment

6.6.1 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 } )
```

6.6.2 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.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

7 General concepts for securing protocols

7.1 Introduction

Protocols used in information and communications technology (ICT) infrastructures need to include cybersecurity capabilities to make them resistant to attacks. The intention of this Specification is to provide a structure where the cybersecurity aspects are clearly separated from the actual protocol requiring cybersecurity. This is done by a specification of a general protocol, called the wrapper protocol, which includes all the cybersecurity aspects. This wrapper protocol can then embed another protocol, called the protected protocol, and in this way supply cybersecurity to the embedded or wrapped protocol. A protocol that is structured according to this Specification can be protected by the wrapper protocol and needs no security capabilities of its own.

The wrapper protocol may protect protocols specific for maintenance of a PKI or a privilege management infrastructure (PMI), but may also be used for protocols with different purposes.

7.2 Protected protocol plug-in concept

Figure 2 is one way of illustrating the relationship between the wrapper protocol and a protected protocol. A protected protocol is seen as embedded in the wrapper protocol, which then provides a protecting shield for the protected protocol. The wrapper protocol can protect any protocol that has been designed for protection by the wrapper protocol (see clauses 7.4 and 8.10).

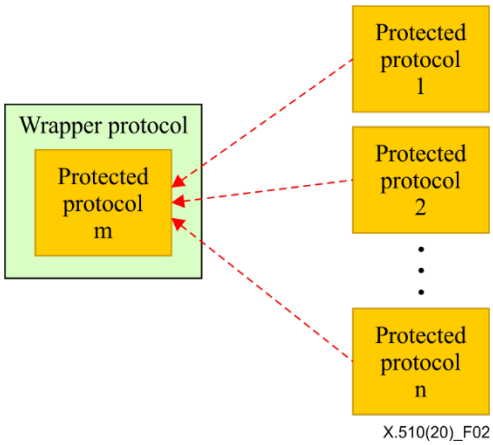

Figure 2 – Protected protocol plug-in

7.3 Communications structure

Figure 3 depicts the communications between two entities that are each running the wrapper protocol protecting an instance of the same protocol. The data exchange by the protected protocol is then protected by the two instances of the wrapper protocol. The two instances of the wrapper protocol may also need to communicate without involving the instances of the protected protocol.

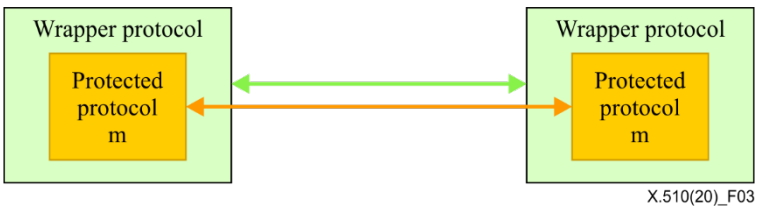

Figure 3 – Embedded communication

7.4 Another view of the relationship between the wrapper protocol and the protected protocol

Figure 4 illustrates how the wrapper protocol embeds a protected protocol. At different places within the wrapper protocol, it interacts with the protected protocol. As the protected protocol specification includes several protected protocol data unit (PrPDU) types, it is required that the top statement of the protected protocol is a choice among the defined PrPDU types.

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

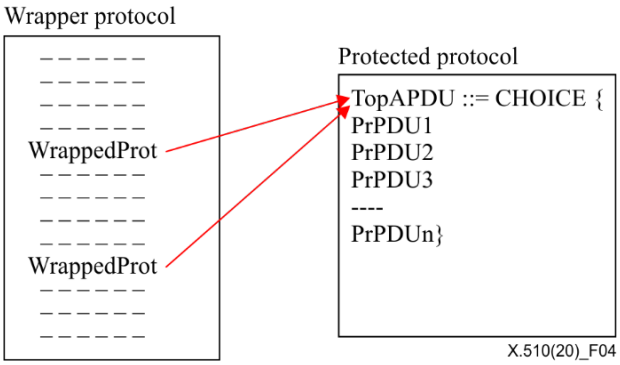

Figure 4 – Inclusion of the protected protocol

7.5 Structure of application protocol data unit

Figure 5 depicts the structure of the transmitted application protocol data unit (APDU), called the wrapper protocol data unit (WrPDU). When actual data is to be transmitted by the two instances of the protected protocol, this data it is embedded in the WrPDU in the form of a protected protocol data unit (PrPDU). When the two instances of the wrapper protocol need to interact without involving the protected protocol, there will be no embedded PrPDU.

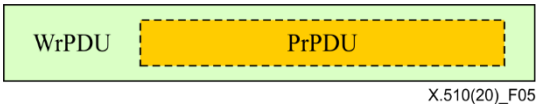

Figure 5 – APDU structure

7.6 Exception conditions

This Specification requires checks of received APDUs (WrPDUs and PrPDUs). The objective is to force compliant implementations to perform these checks to ensure resilient implementations.

This Specification defines two classes of exception.

a) Errors that could potentially be caused by an attack by an adversary. When such an error occurs, no diagnostic information is returned, as such information might be useful for the adversary. Instead, it is assumed that an implementation logs the incident for later analysis.

b) Errors that may safely be assumed not to be caused by an adversary. In this case, a diagnostic message may be returned to the sender. It is recommended that implementations also log such incidents.

IETF RFC 5424 specifies classes of incidents. This Specification refers to two of those classes as follows.

a) Alert: Action shall be taken immediately. Errors that could potentially have been caused by an adversary.

b) Error: Exceptions assumed not to have been caused by an adversary.

In both cases, the association is either not established or it is aborted.

This Specification does not mandate any specific logging mechanism.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

SECTION 2 – THE WRAPPER PROTOCOL

8 Wrapper protocol general concepts

8.1 Introduction

Section 2 specifies the general wrapper protocol providing security to other protocols. This clause considers the wrapper protocol in general. Clause 9 specifies the WrPDUs used to establish, terminate and abort associations. Clause 10 specifies the WrPDUs used during the data transfer phase. Clause 11 analyses the information flow. Clause 12 specifies a procedure for error checking.

The wrapper protocol requires that an association be established between two application entities before they go into the data transfer phase.

NOTE – The terms "association" and "application entity" are taken from the OSI area. However, these terms are here extended also to be applied to a non-OSI area.

The wrapper protocol makes use of parameterized data types to avoid specifications that relate to specific cryptographic algorithms keeping the protocol open to employ different cryptographic algorithms without changes to the basic protocol.

8.2 UTC time specification

The security of the wrapper protocol's shared key generation is based, among other things, on the presence of a trusted source of time on the network.

WrPDUs include information about the time of their creation. The time is given as a coordinated universal time (UTC)time.

This specification makes use of UTC time according to the following.

a) The year, months and date shall be written as YYYYMMDD, where YYYY is the year, MM is the number of the month and DD is the number of the day of the month.

b) The hour, minute and second and fraction of second shall be written as hhmmss,sss, where hh is the hour in the 24 h clock system, mm is the minute and the ss is the seconds. A comma (',') or full stop ('.') is used as the decimal separator.

c) The accuracy shall be either 1 h, 1 min, 1 s or a fraction of a second to any degree of accuracy.

d) The time shall be specified as a pure UTC time and a 'Z' shall be appended to the time. Other forms shall not be used.

e) There shall not be a 'T' between the data and the time specification.

The time value is given as a value of the **`GeneralizedTime`** ASN.1 data type.

8.3 Use of alternative cryptographic algorithms

Some WrPDUs allow for inclusion of alternative cryptographic algorithms. Such alternative cryptographic algorithms are intended for migrating cryptographic algorithms to more safe cryptographic algorithms.

The requirements for cryptographic mechanisms and algorithms acceptable for use in the wrapper protocol will be further defined in future versions of this Specification.

The cryptographic mechanisms and algorithms used by the end-users of the wrapper protocol shall comply with national regulations.

8.4 Establishment of shared keys

To encrypt a wrapped PrPDU or generate an ICV require the establishment of shared symmetric keys. Two types of symmetric keys are defined:

– content encryption keys, one for each direction if encryption of wrapped PrPDUs is required;

– ICV encryption keys, one for each direction, except if AEAD is used.

Different symmetric keys shall be used for the two directions of communication.

The classical way to establish shared symmetric encryption keys is to use a DH key agreement algorithm.

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

8.5 Sequence numbers

The wrapper protocol uses sequence number for WrPDUs transmitted during the data transfer phase, one sequence number per direction of communication. The sequence number shall take the value 0 for the first WrPDU sent after the establishment of an association and be incremented by one for each WrPDU sent. The purpose is:

– to allow detection of replay of WrPDUs caused by an error or by an adversary;

– to detect missing WrPDUs.

The sequence number is a signed integer with a value range from 0 to $2^{31} - 1$ (0 ... 2147483647) (max. 4 octets).

8.6 Use of invocation identification in the wrapper protocol

While protected protocols may have a requirement for pairing data requests and data responses, the same requirement may not be relevant for the wrapper protocol. However, there might be cases where the wrapper protocol needs to pair data requests and data responses, e.g., where non-repudiation is required for the protected protocol.

The invocation identification is held by an **invokeID** component having a syntax of a 6-octet character string where the first three characters are either REQ or RSP, depending on whether it a component of a data request or data response. The last three characters are numerals taking a value in the range 000 to 127, as given by the protected protocol.

8.7 Mapping to underlying services

The wrapper protocol maps directly on to the transmission control protocol (TCP) layer as specified by IETF RFC 793. The TCP port number 9877 as assigned by Internet Assigned Numbers Authority (IANA) to this Specification shall be used.

The service number assigned by IANA is x510.

8.8 Definition of protected protocols

A wrapped protocol is identified by an information object being an instance of the **WRAPPED-PROT** information object class. The **WRAPPED-PROT** information object class is equivalent to the **TYPE-IDENTIFIER** information object class defined by Rec. ITU-T X.681 | ISO/IEC 8824-2.

```
WRAPPED-PROT ::= TYPE-IDENTIFIER
```

This information object is used to bind the type of protected protocol identified by an object identifier to the abstract syntax of that protocol.

A protected protocol reference shall have the following syntax.

```
WrappedProt {WRAPPED-PROT:SupportedProtSet} ::= SEQUENCE {
  id    WRAPPED-PROT.&id({SupportedProtSet}),
  prot  WRAPPED-PROT.&Type({SupportedProtSet}{@id}),
  ... }
```

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

a) the **id** component shall hold the object identifier assigned to the protected protocol in question;

b) The **prot** component shall hold an instance of the top level APDU being a choice of all the PrPDUs defined by the protected protocol (see Figure 4).

How the protected protocol is wrapped by the wrapper is further specified in Annex C.

8.9 Overview of wrapper protocol data units

The **WrapperPDU** data type represents the top APDU of the set of WrPDUs supported by the wrapper protocol.

```
WrapperPDU ::= CHOICE {
  handshakeReq       [0]  HandshakeReq,
  handshakeAcc       [1]  HandshakeAcc,
  handshakeWrpRej    [2]  HandshakeWrpRej,
  handshakeProRej    [3]  HandshakeProRej,
  handshakeSecAbort  [4]  HandshakeSecAbort,
  handshakeProAbort  [5]  HandshakeProAbort,
  dtSecAbort         [6]  DtSecAbort,
  applAbort          [7]  ApplAbort,
  releaseReq         [8]  ReleaseReq,
```

© ISO/IEC 2020 – All rights reserved

Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

```
  releaseRsp         [9]  ReleaseRsp,
  dataTransferClient [10] DataTransferClient,
  dataTransferServer [11] DataTransferServer,
  ... }
```

9 Association management

9.1 Introduction to association management

An association is defined as 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.

The initiator of an association is called the client and the target of an association request is called the server.

9.2 Association handshake request

The **HandshakeReq** WrPDU is used by the client to initiate an association. It has the following syntax.

```
HandshakeReq ::= Signed{TbsHandshakeReq}

TbsHandshakeReq ::= SEQUENCE {
  version        Version DEFAULT {v1},
  prProt         WRAPPED-PROT.&id ({SupportedProtSet}),
  sigAlg         AlgorithmIdentifier {{SupportedSignatureAlgorithms}},
  altSigAlg  [0] AlgorithmIdentifier {{SupportedAltSignatureAlgorithms}} OPTIONAL,
  pkiPath        DER-PkiPath,
  assoID         AssoID,
  time           TimeStamp,
  keyEst         AlgorithmWithInvoke{{SupportedKeyEstablishmentAlgos}},
  altKeyEst  [1] AlgorithmWithInvoke{{SupportedAltKeyEstablishmentAlgos}} OPTIONAL,
  encr-mode      CHOICE {
    aead       [2] SEQUENCE SIZE (1..MAX) OF
      algo           AlgorithmIdentifier{{SupportedAeadAlgorithms}},
    non-aead   [3] SEQUENCE {
      encr       [0] SEQUENCE SIZE (1..MAX) OF
        algo           AlgorithmIdentifier{{SupportedSymmetricKeyAlgorithms}}
                         OPTIONAL,
      icvAlgID   [1] SEQUENCE SIZE (1..MAX) OF
        algo           AlgorithmIdentifier{{SupportedIcvAlgorithms}} },
    ... },
  attCert        DER-AttributeCertificate OPTIONAL,l
  applData   [4] WrappedProt{{SupportedProtSet}} OPTIONAL,
  ... }

Version ::= BIT STRING {
  v1 (0)  -- version 1
  }

DER-PkiPath ::= OCTET STRING
  (CONTAINING PkiPath ENCODED BY der)

DER-AttributeCertificate ::= OCTET STRING
  (CONTAINING AttributeCertificate ENCODED BY der)

der OBJECT IDENTIFIER ::=
  {joint-iso-itu-t asn1(1) ber-derived(2) distinguished-encoding(1)}

AssoID ::= INTEGER (0.. 32767)

TimeStamp ::= GeneralizedTime

SupportedSignatureAlgorithms ALGORITHM ::= {...}

SupportedAltSignatureAlgorithms ALGORITHM ::= {...}

SupportedKeyEstablishmentAlgos ALGORITHM ::= {...}

SupportedAltKeyEstablishmentAlgos ALGORITHM ::= {...}
```


© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

```
SupportedAeadAlgorithms ALGORITHM ::= {...}

SupportedSymmetricKeyAlgorithms ALGORITHM ::= {...}

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

The **Signed** data type is defined in clause 6.5.2. As detailed in clause 8.3, a **Signed** data value may contain two digital signatures during a migration period. When a second digital signature is included, then the **altSigAlg** component shall also be present.

The **TbsHandshakeReq** data type has the following components.

a) The **version** component shall specify the version(s) of the wrapper protocol supported by the client. The **version** component is a bit-string that allows the client to set multiple bits if it supports multiple versions.

NOTE 1 – At the time of publication, only version 1 is defined.

b) The **prProt** component specifies the type of protocol to be protected. It shall hold the object identifier identifying that protocol.

c) The **sigAlg** component shall hold the digital signature algorithm to be used for the digital signature. If the **Signed** data value includes two digital signatures, then this component shall be used to generate the digital signature to be placed in the **signature** component of the **Signed** data type.

The inclusion of the digital signature algorithm within the integrity-protected area serves two purposes. It allows the digital signature algorithm to be protected by the digital signature and by placing it near the start of the WrPDU makes it possible to generate the hash for digital signature verification during the first pass of the WrPDU.

NOTE 2 – This Specification does not mandate a specific set of digital signature algorithms. Reference specifications or implementers' agreements may replace the dots with a set of digital signature algorithms to be supported for a specific environment.

d) The **altSigAlg** component, when present, may be used for digital signature algorithm migration purposes (see clause 8.3). It shall be present if the **Signed** data value includes two digital signatures. Otherwise, it shall be absent. If present, it shall be used to generate the digital signature to be placed in the **altSignature** component of the **Signed** data type.

e) The **pkiPath** component shall hold the certification path necessary to verify the digital signature as specified in Rec. ITU-T X.509 | ISO/IEC 9594-8. The first element shall be a CA certificate issued by a trust anchor trusted by the relying party. The last element shall be the end-entity public-key certificate used to validate the digital signature. As a special case, the end-entity public-key certificate may be issued directly by the trust anchor. This component is encapsulated in an octet string being distinguished encoding rules (DER) encoded allowing other than basic encoding rules (BER) to be used for general encoding.

f) The **assoID** component shall uniquely identify the identity of an association within the context a client and server pair.

g) The **time** component shall hold the UTC generalized time at which this component was created (see clause 8.2 for details).

h) The **keyEst** component shall hold the key establishment algorithm used to establish shared symmetric keys to be used during data transfer. Two or four keys shall be generated as specified in item j).

i) The **altKeyEst** component, when present, may be used for key establishment algorithm migration.

j) The **encr-mode** component is a choice between two alternatives as follows.

– The **aead** alternative is used when the client wants to take advantage of the performance benefits of using an AEAD algorithm for both encryption and integrity protection using a single operation. When this alternative is taken, two symmetric keys shall be generated, one for each direction. The client may specify several AEAD algorithms. The client shall list the algorithms according to its preference by having the most preferred algorithm as the first one in the sequence-of. If a **HandshakeAcc** WrPDU is to be returned, the server shall take the same alternative and specify the first AEAD algorithm it supports of those suggested.

– The **non-aead** alternative is taken when the client does not suggest that encryption be used or the client for some reason wants added flexibility by using separate encryption and ICV specifications.

This alternative requires two symmetric keys to be generated for ICV purposes and an additional two symmetric keys to be generated if encryption is required.

This alternative has two components as follows.

© ISO/IEC 2020 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

i) In the **encr** component, when present, the client shall suggest a sequence-of one or more symmetric key algorithms listed according to its preference by having the most preferred algorithm as the first one in the sequence-of. If a **HandshakeAcc** WrPDU is to be returned, the server shall, if relevant, specify the first symmetric key algorithm it supports of those suggested.

ii) In the **icvAlgID** component, the client shall suggest a sequence-of one or more ICV algorithms listed according to its preference by having the most preferred algorithm as the first one in the sequence-of. If a **HandshakeAcc** WrPDU is to be returned, the server shall specify the first ICV algorithm it supports of those suggested.

k) The **attCert** component, when present, shall hold an attribute certificate providing access control information. This component is encapsulated in an octet string being DER encoded allowing other than BER to be used for general encoding.

l) The **applData** component, when required by the protected protocol, shall hold an instance of a PrPDU specifying some initialization information required by the protected protocol.

9.3 Association accept

The **HandshakeAcc** WrPDU shall be issued by the server, when accepting an association request.

The **HandshakeAcc** WrPDU has the following syntax.

```
HandshakeAcc ::= Signed{TbsHandshakeAcc}

TbsHandshakeAcc ::= SEQUENCE {
  version        Version DEFAULT {v1},
  sigSel         CHOICE {
    sigAlg         AlgorithmIdentifier{{SupportedSignatureAlgorithms}},
    altSigAlg  [0] AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}} },
  pkiPath        DER-PkiPath,
  assoID         AssoID,
  time           TimeStamp,
  keyEstSel      CHOICE {
    keyEst         AlgorithmWithInvoke{{SupportedKeyEstablishmentAlgos}},
    altKeyEst  [1] AlgorithmWithInvoke{{SupportedAltKeyEstablishmentAlgos}} },
  encr-mode      CHOICE {
    aead       [2] AlgorithmIdentifier{{SupportedAeaAlgorithms}},
    non-aead   [3] SEQUENCE {
      encr       [0] AlgorithmIdentifier{{SupportedSymmetricKeyAlgorithms}} OPTIONAL,
      icvAlgID   [1] AlgorithmIdentifier{{SupportedIcvAlgorithms}} },
    ... },
  attCert        DER-AttributeCertificate OPTIONAL,
  applData   [4] WrappedProt{{SupportedProtSet}} OPTIONAL,
  ... }
```

The **TbsHandshakeAcc** data type has the following components.

a) The **version** component shall specify exactly one version that is supported by the server. It shall be selected among those suggested in the corresponding **HandshakeReq**. The highest supported version of those suggested by the client should be selected.

b) The **sigSel** component has two alternatives:

– the **sigAlg** alternative shall be selected if the client did not include an alternative digital signature algorithm or if the client included an alternative algorithm, but the server does not support that alternative algorithm;

– the **altSigAlg** alternative shall be selected if the client has included an alternative digital signature algorithm and the server supports that algorithm.

The alternative taken determines which digital signature algorithm shall be used in future communications within the association.

The selected digital signature algorithm shall be used to generate the digital signature to be placed in the **signature** component of the **Signed** data type. The **altSignature** component of the Signed data type shall be absent.

NOTE – If the **altSigAlg** alternative is taken, the parties in the communication now know that they both support the more secure digital signature algorithm, which may be utilized in future communications.

© ISO/IEC 2020 – All rights reserved
Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020

c) The **pkiPath** component shall hold the certification path used to verify the digital signature. This component shall be encapsulated in an octet string being DER encoded allowing other than BER to be used for general encoding.

d) The **assoID** component shall have the same value as specified for the corresponding component of the associated **HandshakeReq** WrPDU.

e) The **time** component shall hold the UTC generalized time at which of this component was created (see clause 8.2 for details).

f) The **keyEstSel** component has two alternatives:

– the **keyEst** alternative shall be selected if the client did not include an alternative key establishment or if the client included an alternative algorithm, but the server does not support that alternative algorithm;

– the **altKeyEst** alternative shall be selected if the client has included an alternative key establishment algorithm and the server supports that algorithm.

g) The **encr-mode** component is a choice between two alternatives. The server shall take the same alternative as the client did in the **HandshakeReq** WrPDU:

– if the client selected the **aead** alternative, the server shall select the first supported AE-algorithm of the sequence-of of the AEAD-algorithms suggested by the client;

– if the client selected the **non-aead** alternative, the server shall include the same components as included in the **HandshakeReq** WrPDU:

i) if the client included the **encr** component, the server shall select the first supported symmetric key algorithm of the sequence-of of those suggested by the client,

ii) the **icvAlgID** component shall select the first supported ICV algorithm of the sequence-of of those suggested by the client.

h) The **attCert** component, when present, shall hold an attribute certificate. This component shall be encapsulated in an octet string being DER encoded allowing other than BER to be used for general encoding.

i) The **applData** component, when required by the protected protocol, shall hold an instance of a PrPDU responding to initialization information provided by the client.

9.4 Association reject due to security issues

When the server detects an error or an unsupported component in the wrapper protocol part of a **HandshakeReq** WrPDU, a **HandshakeWrpRej** WrPDU shall be returned.

The **HandshakeWrpRej** WrPDU has the following syntax.

```
HandshakeWrpRej ::= Signed{TbsHandshakeWrpRej}

TbsHandshakeWrpRej ::= SEQUENCE {
  version        Version DEFAULT {v1},
  sigSel         CHOICE {
    sigAlg         AlgorithmIdentifier{{SupportedSignatureAlgorithms}},
    altSigAlg  [0] AlgorithmIdentifier{{SupportedAltSignatureAlgorithms}} },
  assoID         AssoID,
  time           TimeStamp,
  pkiPath        DER-PkiPath,
  diag           WrpError OPTIONAL,
  ... }
```

The **altAlgorithmIdentifier** and the **altSignature** components of the **Signed** data value shall be absent.

The **TbsHandshakeWrpRej** data type has the following components.

a) The **version** component shall specify exactly one version that is supported by the server. It should be selected among those suggested in the corresponding **HandshakeReq** WrPDU. The highest supported version of those suggested should be selected. However, if the server does not support any of the suggested versions, it shall return one alternative value it does support.

b) The **sigSel** component has two alternatives:

© ISO/IEC 2020 – All rights reserved

Documents Document Sharing Hub
Full document details: BS ISO/IEC 9594-11:2020