

BS ISO/IEC 9594-8:2014

Incorporating corrigendum October 2015



BSI Standards Publication

Information technology — Open Systems Interconnection — The Directory

Part 8: Public-key and attribute certificate
frameworks

bsi.

...making excellence a habit.
AVA **Documents**
Document Sharing Hub

National foreword

This British Standard is the UK implementation of ISO/IEC 9594-8:2014, incorporating corrigendum October 2015.

The start and finish of text introduced or altered by corrigendum is indicated in the text by tags. Text altered by ISO/IEC corrigendum October 2015 is indicated in the text by **AC1** ~~AC1~~.

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 secretary.

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 2015.

Published by BSI Standards Limited 2015

ISBN 978 0 580 92217 6

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 March 2014.

Amendments/corrigenda issued since publication

Date	Text affected
31 October 2015	Implementation of ISO/IEC corrigendum October 2015

**Information technology — Open Systems
Interconnection — The Directory —**

**Part 8:
Public-key and attribute certificate
frameworks**

*Technologies de l'information — Interconnexion de systèmes ouverts
(OSI) — L'annuaire*

Partie 8: Cadre général des certificats de clé publique et d'attribut





COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2014

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

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 9594-8 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*, in collaboration with ITU-T. The identical text is published as Rec. ITU-T X.509 (10/2012).

This seventh edition cancels and replaces the sixth edition (ISO/IEC 9594-8:2008), which has been technically revised. It also incorporates the Technical Corrigenda ISO/IEC 9594-8:2008/Cor.1:2011, ISO/IEC 9594-8:2008/Cor.2:2012 and ISO/IEC 9594-8:2008/Cor.3:2013.

ISO/IEC 9594 consists of the following parts, under the general title *Information technology — Open Systems Interconnection — The Directory*:

- *Part 1: Overview of concepts, models and services*
- *Part 2: Models*
- *Part 3: Abstract service definition*
- *Part 4: Procedures for distributed operation*
- *Part 5: Protocol specifications*
- *Part 6: Selected attribute types*
- *Part 7: Selected object classes*
- *Part 8: Public-key and attribute certificate frameworks*
- *Part 9: Replication*

CONTENTS

	<i>Page</i>
1 Scope	1
2 Normative references	2
2.1 Identical Recommendations International Standards	2
2.2 Paired Recommendations International Standards equivalent in technical content.....	3
2.3 Recommendations	3
2.4 Other references	3
3 Definitions	3
3.1 OSI Reference Model security architecture definitions	3
3.2 Baseline identity management terms and definitions	3
3.3 Directory model definitions	4
3.4 Access control framework definitions.....	4
3.5 Public-key and attribute certificate definitions.....	4
4 Abbreviations	7
5 Conventions.....	8
6 Frameworks overview	8
6.1 Digital signatures	9
6.2 Formal definitions for public-key cryptography	10
6.3 Distinguished encoding of Basic Encoding Rules.....	10
6.4 Applying distinguished encoding.....	11
7 Public-keys and public-key certificates.....	11
7.1 Introduction.....	11
7.2 Public-key certificate	12
7.3 Public-key certificate extensions.....	14
7.4 Types of public-key certificates	15
7.5 Trust anchor	15
7.6 Entity relationship	16
7.7 Certification path.....	16
7.8 Generation of key pairs	18
7.9 Public-key certificate creation.....	18
7.10 Certificate revocation list	19
7.11 Repudiation of a digital signing	21
8 Public-key certificate and CRL extensions.....	22
8.1 Policy handling.....	23
8.2 Key and policy information extensions.....	25
8.3 Subject and issuer information extensions	31
8.4 Certification path constraint extensions	33
8.5 Basic CRL extensions	37
8.6 CRL distribution points and delta-CRL extensions.....	46
9 Delta CRL relationship to base.....	52
10 Certification path processing procedure	53
10.1 Path processing inputs.....	53
10.2 Path processing outputs.....	54
10.3 Path processing variables	54
10.4 Initialization step.....	55
10.5 Certificate processing.....	55
11 PKI directory schema	57
11.1 PKI directory object classes and name forms.....	57
11.2 PKI directory attributes	59
11.3 PKI directory matching rules	61
11.4 PKI directory syntax definitions	66

	<i>Page</i>
12 Attribute Certificates	68
12.1 Attribute certificate structure	69
12.2 Attribute certification paths.....	71
13 Attribute Authority, SOA and Certification Authority relationship	71
13.1 Privilege in attribute certificates	73
13.2 Privilege in public-key certificates.....	73
14 PMI models	73
14.1 General model	73
14.2 Control model	75
14.3 Delegation model	76
14.4 Group assignment model.....	76
14.5 Roles model.....	77
14.6 Recognition of Authority Model	78
14.7 XML privilege information attribute.....	82
14.8 Permission attribute and matching rule	83
15 Privilege management certificate extensions.....	83
15.1 Basic privilege management extensions.....	84
15.2 Privilege revocation extensions.....	87
15.3 Source of Authority extensions	87
15.4 Role extensions	90
15.5 Delegation extensions	91
15.6 Recognition of Authority Extensions	95
16 Privilege path processing procedure.....	98
16.1 Basic processing procedure.....	98
16.2 Role processing procedure	99
16.3 Delegation processing procedure	99
17 PMI directory schema	102
17.1 PMI directory object classes	102
17.2 PMI Directory attributes.....	103
17.3 PMI general directory matching rules	105
18 Directory authentication	107
18.1 Simple authentication procedure	107
18.2 Password policy	109
18.3 Strong Authentication	119
19 Access control	122
20 Protection of Directory operations	122
Annex A – Public-Key and Attribute Certificate Frameworks.....	123
Annex B – Reference definition of algorithm object identifiers.....	153
Annex C – CRL generation and processing rules	154
C.1 Introduction.....	154
C.2 Determine parameters for CRLs	155
C.3 Determine CRLs required	156
C.4 Obtain CRLs	157
C.5 Process CRLs	157
Annex D – Examples of delta CRL issuance.....	161
Annex E – Privilege policy and privilege attribute definition examples	163
E.1 Introduction.....	163
E.2 Sample syntaxes	163
E.3 Privilege attribute example.....	167
Annex F – An introduction to public key cryptography ²⁾	168
Annex G – Examples of use of certification path constraints	170

	<i>Page</i>
G.1 Example 1: Use of basic constraints.....	170
G.2 Example 2: Use of policy mapping and policy constraints	170
G.3 Use of Name Constraints Extension.....	170
Annex H – Guidance on determining for which policies a certification path is valid	179
H.1 Certification path valid for a user-specified policy required	179
H.2 Certification path valid for any policy required	180
H.3 Certification path valid regardless of policy	180
H.4 Certification path valid for a user-specific policy desired, but not required	180
Annex I – Key usage certificate extension issues	181
Annex J – External ASN.1 modules	182
Annex K – Use of Protected Passwords for Bind operations.....	190
Annex L – Examples of password hashing algorithms.....	191
L.1 Null Hashing method	191
L.2 MD5 method	191
L.3 SHA-1 method	191
Annex M – Alphabetical list of information item definitions	192
Annex N – Amendments and corrigenda.....	195

Introduction

This Recommendation | International Standard, together with other Recommendations | International Standards, has been produced to facilitate the interconnection of information processing systems to provide directory services. A set of such systems, together with the directory information which they hold, can be viewed as an integrated whole, called the *Directory*. The information held by the Directory, collectively known as the Directory Information Base (DIB), is typically used to facilitate communication between, with or about objects such as application-entities, people, terminals and distribution lists.

The Directory plays a significant role in Open Systems Interconnection, whose aim is to allow, with a minimum of technical agreement outside of the interconnection standards themselves, the interconnection of information processing systems:

- from different manufacturers;
- under different managements;
- of different levels of complexity; and
- of different ages.

Many applications have requirements for security to protect against threats to the communication of information. Virtually all security services are dependent upon the identities of the communicating parties being reliably known, i.e., authentication.

This Recommendation | International Standard defines a framework for public-key certificates. This framework includes the specification of data objects used to represent the certificates themselves, as well as revocation notices for issued certificates that should no longer be trusted. The public-key certificate framework defined in this Recommendation | International Standard, while it defines some critical components of a public-key infrastructure (PKI), it does not define a PKI in its entirety. However, this Recommendation | International Standard provides the foundation upon which full PKIs and their specifications would be built.

Similarly, this Recommendation | International Standard defines a framework for attribute certificates. That framework includes the specification of data objects used to represent the certificates themselves, as well as revocation notices for issued certificates that should no longer be trusted. The attribute certificate framework defined in this Recommendation | International Standard, while it defines some critical components of a Privilege Management Infrastructure (PMI), it does not define a PMI in its entirety. However, this Recommendation | International Standard provides the foundation upon which full PMIs and their specifications would be built.

Information objects for holding PKI and PMI objects in the Directory and for comparing presented values with stored values are also defined.

This Recommendation | International Standard also defines a framework for the provision of authentication services by the Directory to its users.

This Recommendation | International Standard provides the foundation frameworks upon which industry profiles can be defined by other standards groups and industry forums. Many of the features defined as optional in these frameworks may be mandated for use in certain environments through profiles. This seventh edition technically revises and enhances the sixth edition of this Recommendation | International Standard.

This seventh edition specifies versions 1, 2 and 3 of public-key certificates and versions 1 and 2 of certificate revocation lists. This edition also specifies version 2 of attribute certificates.

The extensibility function was added in an earlier edition with version 3 of the public-key certificate and with version 2 of the certificate revocation list and was incorporated into the attribute certificate from its initial inception. This function is specified in clause 7. It is anticipated that any enhancements to this edition can be accommodated using this function and avoid the need to create new versions.

Annex A, which is an integral part of this Recommendation | International Standard, provides the ASN.1 modules which contain all of the definitions associated with the frameworks.

Annex B, which is an integral part of this Recommendation | International Standard, defines object identifiers assigned to authentication and encryption algorithms, in the absence of a formal register.

Annex C, which is an integral part of this Recommendation | International Standard, provides rules for generating and processing Certificate Revocation Lists.

Annex D, which is not an integral part of this Recommendation | International Standard, provides examples of delta-CRL issuance.

Annex E, which is not an integral part of this Recommendation | International Standard, provides examples of privilege policy syntaxes and privilege attributes.

Annex F, which is not an integral part of this Recommendation | International Standard, is an introduction to public-key cryptography.

Annex G, which is not an integral part of this Recommendation | International Standard, contains examples of the use of certification path constraints.

Annex H, which is not an integral part of this Recommendation | International Standard, provides guidance for PKI enabled applications on the processing of certificate policy while in the certification path validation process.

Annex I, which is not an integral part of this Recommendation | International Standard, provides guidance on the use of the **contentCommitment** bit in the **keyUsage** certificate extension.

Annex J, which is not an integral part of this Recommendation | International Standard, includes extracts of external ASN.1 modules referenced by this Recommendation | International Standard.

Annex K, which is not an integral part of this Recommendation | International Standard, provides a suggested technique for a Bind protected password.

Annex L, which is not an integral part of this Recommendation | International Standard, gives some examples of password hashing algorithms.

Annex M, which is not an integral part of this Recommendation | International Standard, contains an alphabetical list of information item definitions in this Recommendation | International Standard.

Annex N, which is not an integral part of this Recommendation | International Standard, lists the amendments and defect reports that have been incorporated to form this edition of this Recommendation | International Standard.

INTERNATIONAL STANDARD
RECOMMENDATION ITU-T

**Information technology – Open Systems Interconnection –
The Directory: Public-key and attribute certificate frameworks**

SECTION 1 – GENERAL

1 Scope

This Recommendation | International Standard addresses some of the security requirements in the areas of authentication and other security services through the provision of a set of frameworks upon which full services can be based. Specifically, this Recommendation | International Standard defines frameworks for:

- public-key certificates;
- attribute certificates; and
- authentication services.

The public-key certificate framework defined in this Recommendation | International Standard includes a definition of the information objects for a public-key infrastructure (PKI), including public-key certificates and Certificate Revocation Lists (CRLs). The attribute certificate framework includes a definition of the information objects for a Privilege Management Infrastructure (PMI), including attribute certificates, and Attribute Certificate Revocation Lists (ACRLs). This Recommendation | International Standard also provides the framework for issuing, managing, using and revoking certificates. An extensibility mechanism is included in the defined formats for both certificate types and for all revocation list schemes. This Recommendation | International Standard also includes a set of standard extensions for each, which is expected to be generally useful across a number of applications of PKI and PMI. The schema components (including object classes, attribute types and matching rules) for storing PKI and PMI objects in the Directory, are included in this Recommendation | International Standard. Other elements of PKI and PMI, beyond these frameworks, such as key and certificate management protocols, operational protocols, additional certificate and CRL extensions are expected to be defined by other standards bodies (e.g., ISO TC 68, IETF, etc.).

The authentication scheme defined in this Recommendation | International Standard is generic and may be applied to a variety of applications and environments.

The Directory makes use of public-key certificates and attribute certificates, and the framework for the Directory's use of these facilities is also defined in this Recommendation | International Standard. Public-key technology, including certificates, is used by the Directory to enable strong authentication and signed operations, and for storage of signed data in the Directory. Attribute certificates can be used by the Directory to enable rule-based access control. Although the framework for these is provided in this Recommendation | International Standard, the full definition of the Directory's use of these frameworks, and the associated services provided by the Directory and its components is supplied in the complete set of ITU-T X.500 series of Recommendations | ISO/IEC 9594 (all parts).

This Recommendation | International Standard, in the Authentication services framework, also:

- specifies the form of authentication information held by the Directory;
- describes how authentication information may be obtained from the Directory;
- states the assumptions made about how authentication information is formed and placed in the Directory;
- defines three ways in which applications may use this authentication information to perform authentication and describes how other security services may be supported by authentication.

This Recommendation | International Standard describes two levels of authentication: simple authentication, using a password as a verification of claimed identity; and strong authentication, involving credentials formed using cryptographic techniques. While simple authentication offers some limited protection against unauthorized access, only strong authentication should be used as the basis for providing secure services. It is not intended to establish this as a general framework for authentication, but it can be of general use for applications which consider these techniques adequate.

Authentication (and other security services) can only be provided within the context of a defined security policy. It is a matter for users of an application to define their own security policy which may be constrained by the services provided by a standard.

It is a matter for standards-defining applications which use the authentication framework to specify the protocol exchanges which need to be performed in order to achieve authentication based upon the authentication information obtained from the Directory. The protocol used by applications to obtain credentials from the Directory is the Directory Access Protocol (DAP), specified in Rec. ITU-T X.519 | ISO/IEC 9594-5.

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.411 (1999) | ISO/IEC 10021-4:2003, *Information technology – Message Handling Systems (MHS) – Message Transfer System: Abstract Service Definition and Procedures.*
- Recommendation ITU-T X.500 (2012) | ISO/IEC 9594-1:2014, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services.*
- Recommendation ITU-T X.501 (2012) | ISO/IEC 9594-2:2014, *Information technology – Open Systems Interconnection – The Directory: Models.*
- Recommendation ITU-T X.511 (2012) | ISO/IEC 9594-3:2014, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition.*
- Recommendation ITU-T X.518 (2012) | ISO/IEC 9594-4:2014, *Information technology – Open Systems Interconnection – The Directory: Procedures for distributed operation.*
- Recommendation ITU-T X.519 (2012) | ISO/IEC 9594-5:2014, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications.*
- Recommendation ITU-T X.520 (2012) | ISO/IEC 9594-6:2014, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types.*
- Recommendation ITU-T X.521 (2012) | ISO/IEC 9594-7:2014, *Information technology – Open Systems Interconnection – The Directory: Selected object classes.*
- Recommendation ITU-T X.525 (2012) | ISO/IEC 9594-9:2014, *Information technology – Open Systems Interconnection – The Directory: Replication.*
- Recommendation ITU-T X.660 (2008) | ISO/IEC 9834-1:2008, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the International Object Identifier tree.*
- Recommendation ITU-T X.680 (2008) | ISO/IEC 8824-1:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- Recommendation ITU-T X.681 (2008) | ISO/IEC 8824-2:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- Recommendation ITU-T X.682 (2008) | ISO/IEC 8824-3:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- Recommendation ITU-T X.683 (2008) | ISO/IEC 8824-4:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*
- Recommendation ITU-T X.690 (2008) | ISO/IEC 8825-1:2008, *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 (2008) | ISO/IEC 8825-2:2008, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER).*
- Recommendation ITU-T X.812 (1995) | ISO/IEC 10181-3:1996, *Information technology – Open Systems Interconnection – Security frameworks for open systems: Access control framework.*
- Recommendation ITU-T X.813 (1996) | ISO/IEC 10181-4:1997, *Information technology – Open Systems Interconnection – Security frameworks for open systems: Non-repudiation framework.*
- Recommendation ITU-T X.841 (2000) | ISO/IEC 15816:2002, *Information technology – Security techniques – Security information objects for access control.*

2.2 Paired Recommendations | International Standards equivalent in technical content

- Recommendation CCITT 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 Recommendations

- Recommendation ITU-T X.1252 (2010), *Baseline identity management terms and definitions*.

2.4 Other references

- IETF RFC 791 (1981), *Internet Protocol*.
- IETF RFC 822 (1982), *STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES*.
- IETF RFC 1035 (1987), *Domain names – implementation and specification*.
- IETF RFC 1630 (1994), *Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web*.
- IETF RFC 4523 (2006), *Lightweight Directory Access Protocol (LDAP) Schema Definitions for X.509 Certificates*.
- IETF RFC 5280 (2008), *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*.
- AC1 – IETF RFC 5914 (2010), *Trust Anchor Format*.
- IETF RFC 3492 (2003), *Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)*.
- IETF RFC 5890 (2010), *Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework*. AC1

3 Definitions

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

3.1 OSI Reference Model security architecture definitions

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

- asymmetric (encipherment);
- authentication exchange;
- authentication information;
- confidentiality;
- credentials;
- cryptography;
- data origin authentication;
- decipherment;
- digital signature;
- encipherment;
- key;
- password;
- peer-entity authentication;
- symmetric (encipherment).

3.2 Baseline identity management terms and definitions

The following term is defined in Rec. ITU-T X.1252:

- trust: The firm belief in the reliability and truth of information or in the ability and disposition of an entity to act appropriately, within a specified context.

3.3 Directory model definitions

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

- a) attribute;
- b) Directory Information Base;
- c) Directory Information Tree;
- d) Directory System Agent;
- e) Directory User Agent;
- f) distinguished name;
- g) entry;
- h) object;
- i) root.

3.4 Access control framework definitions

The following terms are defined in Rec. ITU-T X.812 | ISO/IEC 10181-3:

- a) Access control Decision Function (ADF);
- b) Access control Enforcement Function (AEF).

3.5 Public-key and attribute certificate definitions

The following terms are defined in this Recommendation | International Standard:

3.5.1 attribute certificate (AC): A data structure, digitally signed by an Attribute Authority, that binds some attribute values with identification information about its holder.

3.5.2 Attribute Authority (AA): An authority which assigns privileges by issuing attribute certificates.

3.5.3 attribute authority revocation list (AARL): A revocation list containing a list of references to attribute certificates issued to AAs that are no longer considered valid by the issuing authority.

3.5.4 attribute certificate revocation list (ACRL): A revocation list containing a list of references to attribute certificates that are no longer considered valid by the issuing authority.

3.5.5 authentication token; (token): Information conveyed during a strong authentication exchange, which can be used to authenticate its sender.

3.5.6 authority: An entity, responsible for the issuance of certificates. Two types are defined in this Recommendation | International Standard; a certification authority which issues public-key certificates and an attribute authority which issues attribute certificates.

3.5.7 authority certificate: A certificate issued to an authority (e.g., either to a certification authority or to an attribute authority).

3.5.8 base CRL: A CRL that is used as the foundation in the generation of a dCRL.

3.5.9 CA-certificate: A public-key certificate for one CA issued by either another CA or by the same CA.

3.5.10 certificate policy: A named set of rules that indicate the applicability of a certificate to a particular community and/or class of application with common security requirements. For example, a particular certificate policy might indicate the applicability of a type of certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.

3.5.11 certification practice statement (CPS): A statement of the practices that a CA employs in issuing certificates.

3.5.12 certificate revocation list (CRL): A signed list indicating a set of certificates that are no longer considered valid by the certificate issuer. In addition to the generic term CRL, some specific CRL types are defined for CRLs that cover particular scopes.

3.5.13 certificate serial number: An integer value, unique within the issuing authority, which is unambiguously associated with a certificate issued by that authority.

3.5.14 certificate-using system: An implementation of those functions defined in this Recommendation | International Standard that are used by a relying party.

- 3.5.15 certificate validation:** The process of ensuring that a certificate was valid at a given time, including possibly the construction and processing of a certification path, and ensuring that all certificates in that path were valid (i.e., were not expired or revoked) at that given time.
- 3.5.16 certification authority (CA):** An authority trusted by one or more users to create and assign public-key certificates. Optionally the certification authority may create the subjects' keys.
- 3.5.17 certification authority revocation list (CARL):** A revocation list containing a list of CA-certificates issued to certification authorities that are no longer considered valid by the certificate issuer.
- 3.5.18 certification path:** An ordered list of one or more public-key certificates, starting with a public-key certificate signed by the trust anchor, and ending with the public key certificate to be validated. All intermediate public-key certificates, if any, are CA-certificates in which the subject of the preceding certificate is the issuer of the following certificate.
- 3.5.19 CRL distribution point:** A directory entry or other distribution source for CRLs; a CRL distributed through a CRL distribution point may contain revocation entries for only a subset of the full set of certificates issued by one CA or may contain revocation entries for multiple CAs.
- 3.5.20 cross-certificate:** A public-key certificate where the issuer and the subject are different CAs. CAs issue cross-certificates to other CAs as a mechanism to authorize the subject CA's existence.
- 3.5.21 cryptographic system, cryptosystem:** A collection of transformations from plain text into cipher text and vice versa, the particular transformation(s) to be used being selected by keys. The transformations are normally defined by a mathematical algorithm.
- 3.5.22 data confidentiality:** This service can be used to provide the protection of data from unauthorized disclosure. The data confidentiality service is supported by the authentication framework. It can be used to protect against data interception.
- 3.5.23 delegation:** Conveyance of privilege from one entity that holds such privilege, to another entity.
- 3.5.24 delegation path:** An ordered sequence of certificates which together with the authentication of a privilege asserter's identity, can be processed to verify the authenticity of an asserter's privilege.
- 3.5.25 delta-CRL (dCRL):** A partial revocation list that only contains entries for certificates that have had their revocation status changed since the issuance of the referenced base CRL.
- 3.5.26 end-entity:** Either a public-key certificate subject that uses its private key for purposes other than signing certificates, or an attribute certificate holder that uses its attributes to gain access to a resource.
- 3.5.27 end-entity attribute certificate:** An attribute certificate issued to an end-entity.
- 3.5.28 end-entity attribute certificate revocation list (EARL):** A revocation list containing a list of end-entity attribute certificates that are no longer considered valid by the issuing attribute authority.
- 3.5.29 end-entity certificate:** An attribute or public-key certificate issued to an end-entity.
- 3.5.30 end-entity public-key certificate:** A public-key certificate issued to an end-entity.
- 3.5.31 end-entity public-key certificate revocation list (EPRL):** A revocation list containing a list of end-entity public-key certificates that are no longer considered valid by the issuing certification authority.
- 3.5.32 environmental variables:** Those aspects of policy required for an authorization decision, that are not contained within static structures, but are available through some local means to a privilege verifier (e.g., time of day or current account balance).
- 3.5.33 full CRL:** A complete revocation list that contains entries for all certificates that have been revoked for the given scope.
- 3.5.34 hash function:** A (mathematical) function which maps values from a large (possibly very large) domain into a smaller range. A "good" hash function is such that the results of applying the function to a (large) set of values in the domain will be evenly distributed (and apparently at random) over the range.
- 3.5.35 holder:** An entity to whom some privilege has been delegated either directly from the Source of Authority or indirectly through another Attribute Authority.
- 3.5.36 indirect CRL (iCRL):** A revocation list that contains at least revocation information about certificates issued by authorities other than that which issued this CRL.

3.5.37 key agreement: A method for negotiating a key value online without transferring the key, even in an encrypted form, e.g., the Diffie-Hellman technique (see ISO/IEC 11770-1 for more information on key agreement mechanisms).

3.5.38 password expiration: A situation where a user password has reached the end of its validity period; the account is locked and the user has to change the password before doing any other directory operation.

3.5.39 password quality attributes: Attributes that specify how a password shall be constructed. Password quality attributes include things like minimum length, mixture of characters (uppercase, lowercase, figures, punctuation, etc.), and avoidance of trivial passwords.

3.5.40 password history: A list of old passwords and the times they were inserted in the history.

3.5.41 object method: An action that can be invoked on a resource (e.g., a file system may have read, write and execute object methods).

3.5.42 one-way function: A (mathematical) function f which is easy to compute, but which for a general value y in the range, it is computationally difficult to find a value x in the domain such that $f(x) = y$. There may be a few values of y for which finding x is not computationally difficult.

3.5.43 policy decision point (PDP): The point where policy decisions are made (synonymous with ADF).

3.5.44 policy enforcement point (PEP): The point where the policy decisions are actually enforced (synonymous with AEF).

3.5.45 policy mapping: Recognizing that, when a CA in one domain certifies a CA in another domain, a particular certificate policy in the second domain may be considered by the authority of the first domain to be equivalent (but not necessarily identical in all respects) to a particular certificate policy in the first domain.

3.5.46 private key: (In a public key cryptosystem) that key of an entity's key pair which is known only by that entity.

3.5.47 privilege: An attribute or property assigned to an entity by an authority.

3.5.48 privilege assenter: A privilege holder using their attribute certificate or public-key certificate to assert privilege.

3.5.49 privilege management infrastructure (PMI): The infrastructure able to support the management of privileges in support of a comprehensive authorization service and in relationship with a public-key infrastructure.

3.5.50 privilege policy: The policy that outlines conditions for privilege verifiers to provide/perform sensitive services to/for qualified privilege asserters. Privilege policy relates attributes associated with the service, as well as attributes associated with privilege asserters.

3.5.51 privilege verifier: An entity verifying certificates against a privilege policy.

3.5.52 public-key: (In a public key cryptosystem) that key of a user's key pair which is publicly known.

3.5.53 public-key certificate (PKC): The public key of a user, together with some other information, rendered unforgeable by digital signature with the private key of the CA which issued it.

3.5.54 public-key infrastructure (PKI): The infrastructure able to support the management of public keys able to support authentication, encryption, integrity or non-repudiation services.

[AC₁] 3.5.55 registration authority: Those aspects of the responsibilities of a certification authority that are related to identification and authentication of the subject of a public-key certificate to be issued by that certification authority. A registration authority may either be a separate entity or be an integrated part of the certification authority.

NOTE – This definition is different in scope from the one defined in Rec. ITU-T X.660 | ISO/IEC 9834-1. **[AC₁]**

3.5.56 relying party: A user or agent that relies on the data in a certificate in making decisions.

3.5.57 role assignment certificate: A certificate that contains the role attribute, assigning one or more roles to the certificate subject/holder.

3.5.58 role specification certificate: A certificate that contains the assignment of privileges to a role.

3.5.59 sensitivity: Characteristic of a resource that implies its value or importance.

3.5.60 simple authentication: Authentication by means of simple password arrangements.

3.5.61 security policy: The set of rules laid down by the security authority governing the use and provision of security services and facilities.

- 3.5.62 self-issued attribute certificate:** An attribute certificate where the issuer and the holder are the same attribute authority. An attribute authority might use a self-issued attribute certificate, for example, to publish policy information. ^(AC1)
- 3.5.63 self-issued certificate:** A public-key certificate where the issuer and the subject are the same CA. A CA might use self-issued certificates, for example, during a key rollover operation to provide trust from the old key to the new key.
- 3.5.64 self-signed certificate:** A special case of self-issued certificates where the private key used by the CA to sign the certificate corresponds to the public key that is certified within the certificate. A CA might use a self-signed certificate, for example, to advertise their public key or other information about their operations.
- NOTE – Use of self-issued certificates and self-signed certificates issued by other than CAs are outside the scope of this Recommendation | International Standard.
- 3.5.65 source of authority (SOA):** An Attribute Authority that a privilege verifier for a particular resource trusts as the ultimate authority to assign a set of privileges.
- 3.5.66 strong authentication:** Authentication by means of cryptographically derived credentials.
- 3.5.67 trust anchor:** A trust anchor is an entity that is trusted by a relying party and used for validating certificates in certification paths.
- 3.5.68 trust anchor information:** Trust anchor information is at least the: distinguished name of the trust anchor, associated public key, algorithm identifier, public key parameters (if applicable), and any constraints on its use including a validity period. The trust anchor information may be provided as a self-signed CA-certificate or as a normal CA-certificate (i.e., cross-certificate).
- 3.5.69 trust anchor store:** A trust anchor information collection at a relying party for one or more trust anchors. ^(AC1)

4 Abbreviations

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

AA	Attribute Authority
AARL	Attribute Authority Revocation List
AC	Attribute Certificate
ACRL	Attribute Certificate Revocation List
ADF	Access control Decision Function
AEF	Access control Enforcement Function
AIA	Authority Information Access
CA	Certification Authority
CARL	Certification Authority Revocation List
CRL	Certificate Revocation List
DAP	Directory Access Protocol
dCRL	Delta Certificate Revocation List
DIB	Directory Information Base
DIT	Directory Information Tree
DS	Delegation Service
DSA	Directory System Agent
DUA	Directory User Agent
EARL	End-entity Attribute certificate Revocation List
EPRL	End-entity Public-key certificate Revocation List
^(AC1) FQDN	Fully-Qualified Domain Name ^(AC1)
IAI	Issuer's ACs Identifiers
iCRL	Indirect Certificate Revocation List
^(AC1) IDN	Internationalized Domain Name
LDH	Letters, Digits, Hyphen ^(AC1)
OCSF	Online Certificate Status Protocol
PDP	Policy Decision Point

PEP	Policy Enforcement Point
PKC	Public-Key Certificate
PKCS	Public-Key Cryptosystem
PKI	Public-Key Infrastructure
PMI	Privilege Management Infrastructure
RoA	Recognition of Authority
SOA	Source of Authority

5 Conventions

The term "Directory Specification" (as in "this Directory Specification") shall be taken to mean Rec. ITU-T X.509 | ISO/IEC 9594-8. The term "Directory Specifications" shall be taken to mean the ITU-T X.500-series Recommendations and all parts of ISO/IEC 9594.

This Directory Specification uses the term *first edition systems* to refer to systems conforming to the first edition of these Directory Specifications, i.e., the 1988 edition of the series of CCITT X.500 Recommendations and the ISO/IEC 9594:1990 edition.

This Directory Specification uses the term *second edition systems* to refer to systems conforming to the second edition of these Directory Specifications, i.e., the 1993 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:1995 edition.

This Directory Specification uses the term *third edition systems* to refer to systems conforming to the third edition of these Directory Specifications, i.e., the 1997 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:1998 edition.

This Directory Specification uses the term *fourth edition systems* to refer to systems conforming to the fourth edition of the Directory Specifications, i.e., the 2001 editions of Recs ITU-T X.500, X.501, X.511, X.518, X.519, X.520, X.521, X.525, and X.530, the 2000 edition of Rec. ITU-T X.509, and parts 1-10 of the ISO/IEC 9594:2001 edition.

This Directory Specification uses the term *fifth edition systems* to refer to systems conforming to the fifth edition of these Directory Specifications, i.e., the 2005 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:2005 edition.

This Directory Specification uses the term *sixth edition systems* to refer to systems conforming to the sixth edition of these Directory Specifications, i.e., the 2008 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:2008 edition.

This Directory Specification uses the term *seventh edition systems* to refer to systems conforming to the seventh edition of these Directory Specifications, i.e., the 2012 edition of the series of ITU-T X.500 Recommendations and the ISO/IEC 9594:2014 edition.

This Directory 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. The names of procedures, typically referenced when specifying the semantics of processing, are differentiated from normal text by displaying them in bold Times New Roman. Access control permissions are presented in italicized Times New Roman. When a definition is referenced for the first time in normal text it is also presented in italicized Times New Roman.

If the items in a list are numbered (as opposed to using "-" or letters), then the items shall be considered steps in a procedure.

6 Frameworks overview

This Directory Specification defines a framework for obtaining and trusting a public key of an entity in order to encrypt information to be decrypted by that entity, or in order to verify the digital signature of that entity. The framework includes the issuance of a public-key certificate by a Certification Authority (CA) and the validation of that public-key certificate by the relying party, i.e., the entity relying on the content of the public-key certificate. The validation includes:

- establishing a trusted path of public-key certificates between a trusted entity called a trust anchor (see clause 7.5) and the certificate subject, i.e., the entity for which the public-key certificate has been issued;

- verifying the digital signatures on each public-key certificate in the path; and
- validating all the public-key certificates along that path (i.e., that they were not expired or not revoked at a given time).

This Directory Specification defines a framework for obtaining and trusting privilege attributes of an entity in order to determine whether they are authorized to access a particular resource. The framework includes the issuance of an attribute certificate by an Attribute Authority (AA) and the validation of that attribute certificate by a privilege verifier. The validation includes:

- ensuring that the privileges in the certificate are sufficient when compared against the privilege policy;
- establishing a trusted delegation path of certificates if necessary;
- verifying the digital signature on each certificate in the path;
- ensuring that each issuer was authorized to delegate privileges; and
- validating that the certificates have not expired or been revoked by their issuers.

Although PKI and PMI are separate infrastructures and may be established independently from one another, they are related. This Directory Specification recommends that holders and issuers of attribute certificates be identified within attribute certificates by pointers to their appropriate public-key certificates. Authentication of the attribute certificate issuers and holders, to ensure that entities claiming privilege and issuing privilege are who they claim to be, is done using the normal processes of the PKI to authenticate identities. This authentication process is not duplicated within the attribute certificate framework.

This Directory Specification makes extensive use of public-key cryptography. Annex F introduces this technology.

6.1 Digital signatures

Digital signatures are used in both PKI and PMI as the mechanism by which the authority that issues a certificate certifies the binding in the certificate. In PKI, the digital signature of the issuing CA on a public-key certificate certifies the binding between the public-key material and the subject of the public-key certificate. In PMI, the digital signature of the issuing AA certifies the binding between the attributes (privileges) and the holder of the certificate. This subclause describes digital signatures in general. Sections 2 and 3 of this Directory Specification discuss the use of digital signatures within PKI and PMI specifically.

This subclause is not intended to specify a standard for digital signatures in general, but to specify the means by which instances of the PKI and PMI specific data types are signed.

Information (info) is signed by appending to it an enciphered summary of the information. The summary is produced by means of a one-way hash function, while the enciphering is carried out using the private key of the signer (see Figure 1). Thus:

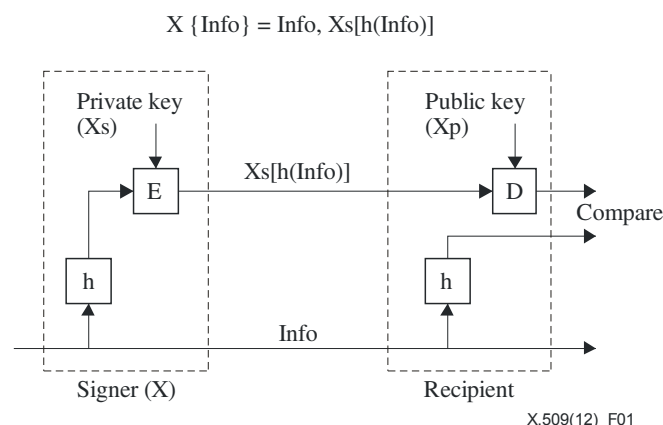


Figure 1 – Digital signatures

NOTE – The encipherment using the private key ensures that the signature cannot be forged. The one-way nature of the hash function ensures that false information, generated to have the same hash result (and thus signature) cannot be substituted.

The recipient of signed information verifies the signature by:

- applying the one-way hash function to the information;
- comparing the result with that obtained by deciphering the signature using the public key of the signer.

This Directory Specification does not mandate a single one-way hash function for use in signing. It is intended that the framework shall be applicable to any suitable hash function, and shall thus support changes to the methods used because of future advances in cryptography, mathematical techniques or computational capabilities. However, two users wishing to authenticate shall support the same hash function for authentication to be performed correctly. Thus, within the context of a set of related applications, the choice of a single function shall serve to maximize the community of users able to authenticate and communicate securely.

6.2 Formal definitions for public-key cryptography

The encipherment of a data item may be described using the following ASN.1:

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

The value of the bit string is generated by taking the octets which form the complete encoding (using the ASN.1 Basic Encoding Rules – Rec. ITU-T X.690 | ISO/IEC 8825-1) of the value of the **ToBeEnciphered** type and applying an encipherment procedure to those octets.

NOTE 1 – The encryption procedure requires agreement on the algorithm to be applied, including any parameters of the algorithm such as any necessary keys, initialization values, and padding instructions. It is the responsibility of the encryption procedures to specify the means by which synchronization of the sender and receiver of data is achieved, which may include information in the bits to be transmitted.

NOTE 2 – The encryption procedure is required to take as input a string of octets and to generate a single string of bits as its result.

NOTE 3 – Mechanisms for secure agreement on the encryption algorithm and its parameters by the sender and receiver of data are outside the scope of this Directory Specification.

The signature of a data item is formed by encrypting a shortened or "hashed" transformation of the item, and may be described by the following ASN.1:

```
HASH{ToBeHashed} ::= SEQUENCE {  
  algorithmIdentifier AlgorithmIdentifier{{SupportedAlgorithms}},  
  hashValue            BIT STRING (CONSTRAINED BY {  
    -- shall be the result of applying a hashing procedure to the DER-encoded  
    -- octets of a value of -- ToBeHashed } ),  
  ... }
```

```
ENCRYPTED-HASH{ToBeSigned} ::= BIT STRING (CONSTRAINED BY {  
  -- shall be the result of applying a hashing procedure to the DER-encoded (see 6.2)  
  -- octets of a value of -- ToBeSigned -- and then applying an encipherment procedure  
  -- to those octets -- } )
```

```
SIGNATURE{ToBeSigned} ::= SEQUENCE {  
  algorithmIdentifier AlgorithmIdentifier{{SupportedAlgorithms}},  
  encrypted            ENCRYPTED-HASH{ToBeSigned},  
  ... }
```

NOTE 4 – The encryption procedure requires the agreements listed in Note 1, and agreement as to whether the hashed octets are encrypted directly, or only after further encoding them as a BIT STRING using the ASN.1 Basic Encoding Rules.

In the case where a signature is appended to a data type, the following ASN.1 may be used to define the data type resulting from applying a signature to the given data type.

```
SIGNED{ToBeSigned} ::= SEQUENCE {  
  toBeSigned          ToBeSigned,  
  COMPONENTS OF SIGNATURE{ToBeSigned},  
  ... }
```

6.3 Distinguished encoding of Basic Encoding Rules

In order to enable the validation of **SIGNED** and **SIGNATURE** types in a distributed environment, a distinguished encoding is required. A distinguished encoding of a **SIGNED** or **SIGNATURE** data value shall be obtained by applying the Basic Encoding Rules defined in Rec. ITU-T X.690 | ISO/IEC 8825-1, with the following restrictions:

- the definite form of length encoding shall be used, encoded in the minimum number of octets;
- for string types, the constructed form of encoding shall not be used;
- if the value of a type is its default value, it shall be absent;
- the components of a Set type shall be encoded in ascending order of their tag value;

- e) the components of a Set-of type shall be encoded in ascending order of their octet value;
- f) if the value of a Boolean type is **TRUE**, the encoding shall have its contents octet set to "FF";
- g) each unused bit in the final octet of the encoding of a Bit String value, if there are any, shall be set to zero;
- h) the encoding of a Real type shall be such that bases 8, 10 and 16 shall not be used, and the binary scaling factor shall be zero;
- i) the encoding of a UTC time shall be as specified in Rec. ITU-T X.690 | ISO/IEC 8825-1;
- j) the encoding of a Generalized time shall be as specified in Rec. ITU-T X.690 | ISO/IEC 8825-1.

6.4 Applying distinguished encoding

Generating distinguished encoding requires the abstract syntax of the data to be encoded to be fully understood. An entity may be required to sign data or check the signature of data that contains unknown protocol extensions or unknown attribute syntaxes. The entity shall follow these rules:

- It shall preserve the encoding of received information whose abstract syntax it does not fully know and which it expects to subsequently sign.
- When signing data for sending, it shall send data whose syntax it fully knows with a distinguished encoding and any other data with its preserved encoding, and shall sign the actual encoding it sends.
- When checking signatures in received data, it shall check the signature against the actual data received rather than its conversion of the received data to a distinguished encoding.

SECTION 2 – PUBLIC-KEY CERTIFICATE FRAMEWORK

The public-key certificate framework defined here is for use by applications with requirements for authentication, integrity, confidentiality and non-repudiation.

The binding of a public-key to an entity is provided by an authority through a digitally signed data structure called a public-key certificate. The format of public-key certificates is defined here, including an extensibility mechanism and a set of specific certificate extensions. If, for any reason, an authority revokes a previously issued public-key certificate, users need to be able to learn that revocation has occurred so they do not use an untrustworthy certificate. Revocation lists are one scheme that can be used to notify users of revocations. The format of revocation lists is defined here, including an extensibility mechanism and a set of revocation list extensions. In both the certificate and revocation list case, other bodies may also define additional extensions that are useful to their specific environments.

A relying party needs to validate a public-key certificate prior to using that public-key certificate for an application. Procedures for performing that validation are also defined here, including verifying the integrity of the public-key certificate itself, its revocation status, and its validity with respect to the intended use.

The Directory uses public-key certificates in its provision of security services including:

- strong authentication between and among directory components;
- authentication and integrity of directory operations; as well as
- integrity and authentication of stored data.

7 Public-keys and public-key certificates

7.1 Introduction

In order for a user to be able to trust a public-key for another user, for instance to authenticate the identity of that user, the public-key shall be obtained from a trusted source. Such a source, called a Certification Authority (CA), certifies a public key by issuing a public-key certificate which binds the public-key to the entity which holds the corresponding private-key. The procedures used by a CA to ensure that an entity is in fact in possession of the private key and other procedures related to the issuance of public-key certificates are outside the scope of this Directory Specification. The certificate, the form of which is specified later in this clause, has the following properties:

- any user with access to the public key of the CA can recover the public key which was certified;
- no party other than the CA can modify the certificate without this being detected (certificates are unforgeable).

Because certificates are unforgeable, they can be published by being placed in the Directory, without the need for the latter to make special efforts to protect them.

NOTE – Although the CAs are unambiguously defined by a distinguished name in the DIT, this does not imply that there is any relationship between the organization of the CAs and the DIT.

7.2 Public-key certificate

A CA issues a public-key certificate of an entity by signing (see clause 6.1) a collection of information, including its distinguished name, the user's distinguished name, a validity period, the value of a public key algorithm and public key, as well as an optional additional information like for the permitted usage of the user's public key. The following ASN.1 data type can be used to represent public-key certificates:

```
Certificate ::= SIGNED{TBSCertificate}

TBSCertificate ::= SEQUENCE {
    version                [0] Version DEFAULT v1,
    serialNumber            CertificateSerialNumber,
    signature               AlgorithmIdentifier{{SupportedAlgorithms}},
    issuer                  Name,
    validity                Validity,
    subject                 Name,
    subjectPublicKeyInfo    SubjectPublicKeyInfo,
    issuerUniqueIdentifier  [1] IMPLICIT UniqueIdentifier OPTIONAL,
    ...,
    [[2: -- if present, version shall be v2 or v3
    subjectUniqueIdentifier [2] IMPLICIT UniqueIdentifier OPTIONAL]],
    [[3: -- if present, version shall be v2 or v3
    extensions             [3] Extensions OPTIONAL]]
    -- If present, version shall be v3]]
}

Version ::= INTEGER {v1(0), v2(1), v3(2)}

CertificateSerialNumber ::= INTEGER

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

-- Definition of the following information object set is deferred, perhaps to
-- standardized profiles or to protocol implementation conformance statements. The
-- set is required to specify a table constraint on the parameters component of
-- AlgorithmIdentifier.

SupportedAlgorithms ALGORITHM ::= {...}
```

The following information object class is used to define specific algorithms.

```
[AC1] ALGORITHM ::= CLASS {
    &Type          OPTIONAL,
    &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
    [PARMS        &Type]
    IDENTIFIED BY &id } [AC1]

Validity ::= SEQUENCE {
    notBefore Time,
    notAfter  Time,
    ... }

SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm      AlgorithmIdentifier{{SupportedAlgorithms}},
    subjectPublicKey BIT STRING,
    ... }

Time ::= CHOICE {
    utcTime        UTCTime,
    generalizedTime GeneralizedTime }

Extensions ::= SEQUENCE OF Extension
```

```

Extension ::= SEQUENCE {
    extnId      EXTENSION.&id({ExtensionSet}),
    critical    BOOLEAN DEFAULT FALSE,
    extnValue   OCTET STRING
        (CONTAINING EXTENSION.&ExtnType({ExtensionSet}){@extnId})
        ENCODED BY der),
    ... }

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

ExtensionSet EXTENSION ::= {...}

```

Before a value of **Time** is used in any comparison operation, e.g., as part of a matching rule in a search, and if the syntax of **Time** has been chosen as the **UTCTime** type, the value of the two digit year field shall be rationalized into a four digit year value as follows:

- If the 2-digit value is 00 through to 49 inclusive, the value shall have 2000 added to it.
- If the 2-digit value is 50 through to 99 inclusive, the value shall have 1900 added to it.

NOTE 1 – The use of GeneralizedTime may prevent interworking with implementations unaware of the possibility of choosing either UTCTime or GeneralizedTime. It is the responsibility of those specifying the domains in which certificates defined in this Directory Specification will be used, e.g., profiling groups, as to when the GeneralizedTime may be used. In no case shall UTCTime be used for representing dates beyond 2049.

The **TBSCertificate** data type is the unsigned public-key certificate and is referred to as a to-be-signed public-key certificate.  It shall be encoded using the DER. 

The **version** field shall hold the version of the encoded public-key certificate. If the **extensions** component is present in the public-key certificate, **version** shall be **v3**. If the **issuerUniqueIdentifier** or **subjectUniqueIdentifier** component is present **version** shall be **v2** or **v3**.

The **serialNumber** field shall hold an integer assigned by the CA to the public-key certificate. The value of **serialNumber** shall be unique for each public-key certificate issued by a given CA (i.e., the issuer name and serial number identify a unique public-key certificate).

The **signature** field contains the algorithm identifier for the algorithm and hash function used by the CA in signing the certificate (e.g., md5WithRSAEncryption, sha-1WithRSAEncryption, id-dsa-with-sha1, etc.). It shall be the same value as used in the **algorithmIdentifier** component of the **SIGNATURE** data type when signing the public-key certificate.

NOTE 2 – This field is redundant except possibly for its participation in matching certificates (see clause 11.3.2).

The **issuer** field shall hold the distinguished name of the CA that issued the public-key certificate. It shall hold a non-empty distinguished name.

The **validity** field shall hold the time interval during which the CA warrants that it will maintain information about the status of the public-key certificate.

The **subject** field shall identify the entity associated with the public-key found in the **subjectPublicKey** component of the **subjectPublicKeyInfo** field. If the public-key certificate is for an end-entity, then the distinguished name may be an empty sequence providing that the **subjectAltName** extension is present and is flagged as critical. Otherwise, it shall be a non-empty distinguished name (see clause 8.3.2.1).

The **subjectPublicKeyInfo** field consists of two components:

- the **algorithm** component shall hold the algorithm which this public key is an instance of (e.g., rsaEncryption, dhpublicnumber, id-dsa, etc.); and
- the **subjectPublicKey** shall hold the public key being certified.

The **issuerUniqueIdentifier** field is used uniquely to identify an issuer in case of name reuse.

The **subjectUniqueIdentifier** field is used uniquely to identify a subject in case of name reuse.

NOTE 3 – The use of issuerUniqueIdentifier and the subjectUniqueIdentifier is deprecated. These fields were added because at one time there was some fear of the reuse of distinguished names.

A user may obtain one or more public-key certificates from one or more CAs. Each certificate bears the name of the CA which issued it.

7.3 Public-key certificate extensions

The **extensions** field allows for the addition of new fields to the structure without modification to the ASN.1 definition. An extension field consists of an extension identifier, a criticality flag, and an encoding of a data value of an ASN.1 type associated with the identified extension. For those extensions where ordering of individual extensions within the **SEQUENCE** is significant, the specification of those individual extensions shall include the rules for the significance of the order therein. When a relying party processing a certificate does not recognize an extension and the criticality flag is **FALSE**, it may ignore that extension. If the criticality flag is **TRUE**, unrecognized extensions shall cause the structure to be considered invalid, i.e., in a certificate, an unrecognized critical extension would cause validation of a signature using that public-key certificate to fail. When a relying party recognizes and is able to fully process an extension, then the relying party shall process the extension regardless of the value of the criticality flag. When a relying party recognizes and is able to partially process an extension for which the criticality flag is **TRUE**, then its behaviour in the presence of unrecognized elements is extension specific and may be documented in each extension. However, the default behaviour, when not specified specifically for an extension, is to treat the entire extension as unrecognized. If unrecognized elements appear within the extension, and the extension is not marked critical, those unrecognized elements shall be ignored according to the rules of extensibility documented in clause 12.2.2 in Rec. ITU-T X.519 | ISO/IEC 9594-5.

Note that any extension that is flagged non-critical will cause inconsistent behaviour among relying parties that will process the extension and relying parties that do not recognize the extension and will ignore it. The same may be true for extensions that are flagged critical, between relying parties that can fully process the extension and those that can partially process the extension, depending upon the extension.

A CA issuing a public-key certificate has three options with respect to an extension:

- i) it can exclude the extension from the certificate;
- ii) it can include the extension and flag it non-critical;
- iii) it can include the extension and flag it critical.

A relying party has three possible actions to take with respect to an extension:

- i) if the extension is unrecognized and is marked non-critical, the relying party shall ignore the extension and accept the certificate (all other things being equal);
- ii) if the extension is unrecognized and marked critical, the relying party shall reject the certificate;
- iii) if the extension is recognized, the relying party shall process the extension and accept or reject the certificate depending on the content of the extension and the conditions under which processing is occurring (e.g., the current values of the path processing variables).

Some extensions can only be marked critical. In these cases, a relying party that understands the extension processes it; the acceptance/rejection of the certificate is dependent (at least in part) on the content of the extension. A relying party that does not understand the extension shall reject the certificate.

Some extensions can only be marked non-critical. In these cases, a relying party that understands the extension shall process it and acceptance/rejection of the certificate is dependent (at least in part) on the content of the extension. A relying party that does not understand the extension accepts the certificate (unless factors other than this extension cause it to be rejected).

Some extensions may be marked critical or non-critical. In these cases, a relying party that understands the extension processes it: the acceptance/rejection of the certificate is dependent (at least in part) on the content of the extension, regardless of the criticality flag. A relying party that does not understand the extension accepts the certificate if the extension is marked non-critical (unless factors other than this extension cause it to be rejected) and rejects the certificate if the extension is marked critical.

When a CA considers including an extension in a certificate it does so with the expectation that its intent will be adhered to wherever possible. If it is necessary that the content of the extension be considered prior to any reliance on the public-key certificate, a CA shall flag the extension critical. This is done with the realization that any relying party that does not process the extension will reject the certificate (probably limiting the set of applications that can verify the certificate). The CA may mark certain extensions non-critical to achieve backward compatibility with validation applications that cannot process the extensions. Where the need for backward compatibility and interoperability with validation applications incapable of processing the extensions is more vital than the ability of the CA to reinforce the extensions, then these optionally critical extensions would be marked non-critical. It is most likely that CAs would set optionally critical extensions as non-critical during a transition period while the verifiers' certificate processing applications are upgraded to ones that can process the extensions.

Specific extensions may be defined in ITU-T Recommendations | International Standards or by any organization which has a need. The object identifier which identifies an extension shall be defined in accordance with Rec. ITU-T X.660 | ISO/IEC 9834-1. Standard extensions for public-key certificates are defined in clause 8 of this Directory Specification.

The following information object class is used to define specific extensions.

```
EXTENSION ::= CLASS {
    &id                OBJECT IDENTIFIER UNIQUE,
    &ExtnType }
WITH SYNTAX {
    SYNTAX             &ExtnType
    IDENTIFIED BY &id }
```

7.4 Types of public-key certificates

There are two primary types of public-key certificates, end-entity public-key certificates and CA-certificates.

An end-entity public-key certificate is a public-key certificate issued by a CA to a subject that is not an issuer of other public-key certificates.

A CA-certificate is a public-key certificate issued by a CA to a subject that is also a CA and therefore is capable of issuing public-key certificates. A CA-certificate shall include the **basicConstraints** extension with the **ca** components set to **TRUE** (see clause 8.4.2.1).

CA-certificates can themselves be categorized by the following types:

- Self-issued certificate – This is a CA-certificate where the issuer and the subject are the same CA. A CA might use self-issued certificates, for example, during a key rollover operation to provide trust from the old key to the new key.
- Self-signed certificate – This is a special case of self-issued certificates where the private key used by the CA to sign the certificate corresponds to the public key that is certified within the certificate. A CA might use a self-signed certificate, for example, to advertise their public key or other information about their operations.
- Cross-certificate – This is a CA-certificate where the issuer and the subject are different CAs. CAs issue certificates to other CAs either as a mechanism to authorize the subject CA's existence (e.g., in a strict hierarchy) or to recognize the existence of the subject CA (e.g., in a distributed trust model). The cross-certificate structure is used for both of these.

AC₁ 7.5 Trust anchor

An entity is a trust anchor for a particular relying party for one or more purposes, typically including certificate validation. A trust anchor is identified by trust anchor information. Trust anchor information includes a public key and some associated data. This trust anchor information is configured into the relying party in a trust anchor store. A relying party may have configured information about multiple trust anchors into one or more trust anchor stores.

A trust anchor may be a CA that issues public-key certificates and certificate revocation lists (CRLs) (see clause 7.10). The relying party may then use the trust anchor information for public-key certificate and CRL validation.

A trust anchor may also function as an end entity by signing other types of information such as software packages, time stamps, responses to online certificate status protocol (OCSP) requests (see IETF RFC 6960), etc.

A CA may be a trust anchor for some entities with respect to particular public-key certificates, but may otherwise be an ordinary CA.

NOTE 1 – As an example, entities within a company may trust all the public-key certificates issued by the company CA. This CA is then the trust anchor for these local relying parties with respect to locally issued public-key certificates. However, by use of name constraints, it might not be a trust anchor with respect to public-key certificates issued outside the company. Likewise, relying parties outside the company may not consider the company CA as the trust anchor for any public-key certificates.

NOTE 2 – The term trust anchor is seen as synonymous with the term root-CA. In a strict hierarchy, the CA at the top of the hierarchy may be the root CA and it may also be a trust anchor. However, in more complex environments, it may not be possible to identify a root CA. Even when it is possible to identify a root CA, a relying party may not necessarily consider it a trust anchor. An intermediate CA may instead take that role.

IETF RFC 5914 defines trust anchor information as a choice between three alternatives:

```
TrustAnchorChoice ::= CHOICE {
    certificate Certificate,
    tbsCert [1] EXPLICIT TBSCertificate,
    taInfo[2] EXPLICIT TrustAnchorInfo } AC1
```

[AC₁] The **certificate** alternative specifies a public-key certificate that can be either a self-signed certificate or a public-key certificate.

The **tbsCert** alternative specifies an unsigned public-key certificate as defined in clause 7.2.

NOTE 3 – This alternative is deprecated by this Specification and therefore not considered further.

The **taInfo** alternative specifies a special trust anchor information format defined by IETF RFC 5914.

In case the trust anchor information is not used for signing public-key certificates, it shall be an end-entity public-key certificate. **[AC₁]**

7.6 Entity relationship

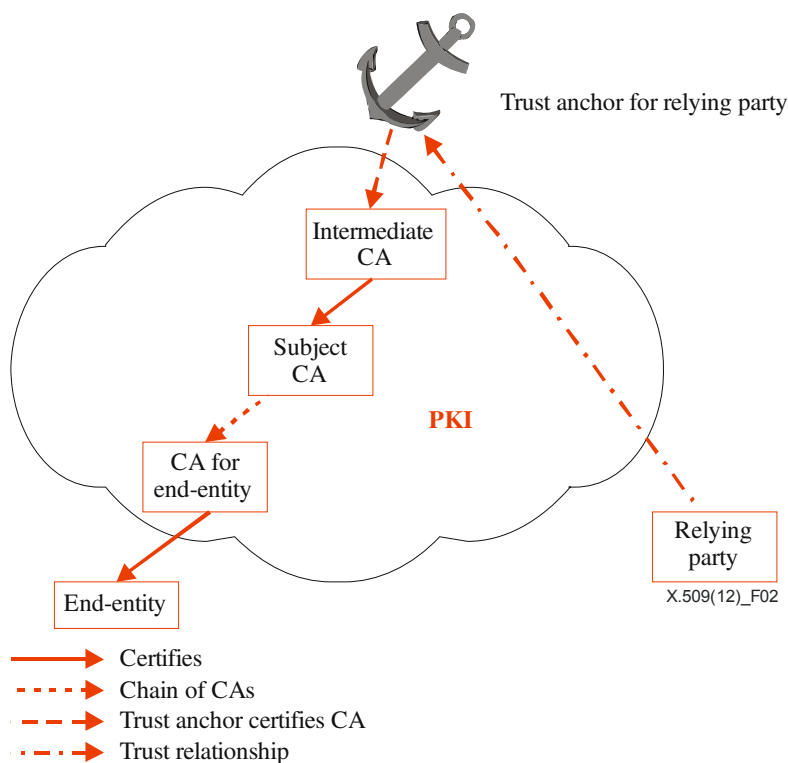


Figure 2 – Entity relationships

There may be several CAs between the trust anchor recognized by the relying party and an end-entity for which a public-key certificate is to be validated. Each CA has issued one or more cross-certificates for the next CA on the path between the trust anchor and the end-entity. The CA that issues a cross-certificate to another CA takes the role of intermediate CA. The CA that is the subject for a cross-certificate takes the role of subject CA. This is illustrated in Figure 2. The same CA may take both the roles of an intermediate CA and a subject CA.

In some situations, conflicting or overlapping requirements for constraints, such as name constraints, may require a CA to issue more than one cross-certificate to another CA. In this case, multiple, different paths of certificates are established between the end-entity and the trust anchor.

7.7 Certification path

Before a public-key certificate can be securely used by a relying party, it shall be validated. In order to validate such a public-key certificate, a chain of public-key certificates, called a certification path, shall be established between the public-key certificate signed by a trust anchor recognized by the relying party and the public-key certificate to be validated. Every public-key certificate within that path shall be checked. A certification path is thus an ordered list of public-key certificates starting with a public-key certificate signed by the trust anchor, and ending with the public key certificate to be validated. All intermediate public-key certificates, if any, are CA-certificates in which the subject of the preceding certificate is the issuer of the following certificate.

Each public-key certificate in a certification path is unique. A path that contains the same certificate multiple times is not a valid certification path.

The **issuer** and **subject** fields of each certificate are used, in part, to identify a valid path. For each pair of adjacent public-key certificates in a valid certification path, the value of the **subject** field in one certificate shall match the value of the **issuer** field in the subsequent certificate. In addition, the value of the **issuer** field in the public-key certificate issued by the trust anchor shall match the distinguished name of the trust anchor. Only the names in these fields are used when checking the validity of a certification path. Names in certificate extensions are not used for this purpose. The **distinguishedNameMatch** matching rule, defined in clause 13.5.2 of Rec. ITU-T X.501 | ISO/IEC 9594-2, shall be used to compare the distinguished name in the **issuer** field of one certificate with the distinguished name in the **subject** field of another.

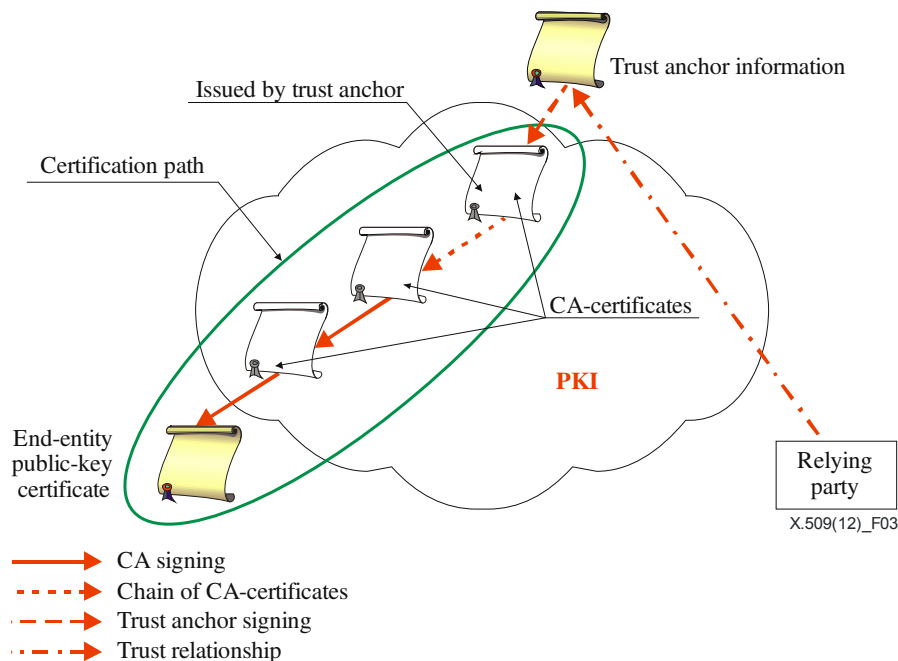


Figure 3 – Certification path

Figure 3 illustrates the situation where a relying party needs to check the validity of an end-entity public-key certificate and the relying party is able to construct a certification path between the end-entity and a trust anchor recognized by the relying party.

Trust suffers dilution as certification paths grow in length. The **basicConstraints** extension (see clause 8.4.2.1) allows restrictions to be put onto the length of the path. The validation of a public-key certificate may be affected by extensions in the chain of a public-key certificate, such as the **certificatePolicies** extension (see clause 8.2.2.6) and **nameConstraints** (see clause 8.4.2.2). It is the responsibility of the relying party to check that the restrictions are observed.

A user may obtain one or more certificates from one or more Certification Authorities. Each certificate bears the name of the CA which issued it. The following ASN.1 data types can be used to represent certificates and a certification path:

```
Certificates ::= SEQUENCE {
    userCertificate      Certificate,
    certificationPath    ForwardCertificationPath OPTIONAL,
    ... }

CertificationPath ::= SEQUENCE {
    userCertificate      Certificate,
    theCACertificates    SEQUENCE OF CertificatePair OPTIONAL}
```

The **userCertificate** component shall hold the end-entity public-key certificate.

The **CACertificates** component may hold an element for each CA from the end-entity up to and including the CA which has been certified by the trust anchor. If the end-entity public-key certificate has been issued directly by the trust anchor, this component shall be absent.

The **CertificatePair** data type is defined in clause 11.2.3. The **issuedToThisCA** component of the **CertificatePair** data type shall be present to ensure an unbroken certification path.

NOTE – The `CertificationPath` data type had already been defined by the first edition of this Directory Specification before the concept of certification path was fully developed. The order of elements in a `CertificationPath` instance appears to be the opposite of that of a certification path. This data type is used, as an example, by the directory protocols for the support of strong authentication and electronic signature. It is recommended that new applications use the `PkiPath` data type.

In addition, the following ASN.1 data type can be used to represent the forward certification path. This component contains the certification path which can point back to the originator.

ForwardCertificationPath ::= SEQUENCE OF CrossCertificates

CrossCertificates ::= SET OF Certificate

PkiPath ::= SEQUENCE OF Certificate

PkiPath is used to represent a certification path. Within the sequence, the order of public-key certificates is such that the subject of the first certificate is the issuer of the second certificate, etc.

Each public-key certificate in a certification path shall be unique. No public-key certificate may appear more than once in a value of the **theCACertificates** component of **CertificationPath** or in a value of **Certificate** in the **CrossCertificates** component of **ForwardCertificationPath** or a value of **Certificate** in **PkiPath**.

7.8 Generation of key pairs

The overall security management policy of an implementation shall define the lifecycle of key pairs, and is, thus, outside the scope of this framework. However, it is vital to the overall security that all private keys remain known only to the entity (subject) to whom they belong.

Key data is not easy for a human user to remember, so a suitable method for storing it in a convenient transportable manner shall be employed. One possible mechanism would be to use a "Smart Card". This would hold the private and (optionally) public keys of the user, the user's certificate, and a copy of the CA's public key. The use of this card shall additionally be secured by, e.g., at least the use of a Personal Identification Number (PIN), increasing the security of the system by requiring the user to possess the card and to know how to access it. The exact method chosen for storing such data, however, is beyond the scope of this Directory Specification.

Three ways in which a user's key pair may be produced are:

- a) The user generates its own key pair. This method has the advantage that a user's private key is never released to another entity, but requires a certain level of competence by the user.
- b) The key pair is generated by a third party. The third party shall release the private key to the user in a physically secure manner, and then actively destroy all information relating to the creation of the key pair plus the keys themselves. Suitable physical security measures shall be employed to ensure that the third party and the data operations are free from tampering.
- c) The key pair is generated by the CA. This is a special case of b), and the considerations there apply.

NOTE – The CA already exhibits trusted functionality with respect to the user, and shall be subject to the necessary physical security measures. This method has the advantage of not requiring secure data transfer to the CA for certification.

The cryptosystem in use imposes particular (technical) constraints on key generation.

7.9 Public-key certificate creation

A public-key certificate associates the public key and unique distinguished name of the subject it describes. Thus:

- a) a CA shall be satisfied of the identity of a subject before creating a certificate for it;
- b) a CA shall not issue certificates for two different subjects with the same name.

It is important that the transfer of information to the CA is not compromised, and suitable physical security measures shall be taken. In this regard:

- a) It would be a serious breach of security if the CA issued a public-key certificate for a subject with a public key that had been tampered with.
- b) If the means of generation of key pairs of 7.8 b) or of 7.8 c) is employed, the subject's private key shall be transferred to the user in a secure manner.
- c) If the means of generation of key pairs of 7.8 a) or of 7.8 b) is employed, the subject may use different methods (online or offline) to communicate its public key to the CA in a secure manner. Online methods may provide some additional flexibility for remote operations performed between the user and the CA.

A public-key certificate is a publicly available piece of information, and no specific security measures need to be employed with respect to its transmission e.g., to a DSA or LDAP server. As it is produced by an offline CA on behalf of a subject who shall be given a copy of it, the subject needs only store this information in its directory entry on a subsequent directory access. Alternatively, the CA could lodge the public-key certificate for the subject, in which case the CA shall be given suitable access rights to entity's entry.

7.10 Certificate revocation list

The authority that issues certificates (public-key or attribute) also has the responsibility to indicate the validity of the certificates that it issues. Generally, certificates are subject to possible subsequent revocation. This revocation and a notification of the revocation may be done directly by the same authority that issued the certificate, or indirectly by another authority duly authorized by the authority that issued the certificate. An authority that issues certificates is required to state, possibly through a published statement of their practices, through the certificates themselves, or through some other identified means, whether:

- the certificates cannot be revoked; or
- the certificates may be revoked by the same certificate-issuing authority directly; or
- the certificate-issuing authority authorizes a different entity to perform revocation.

Authorities that do revoke certificates are required to state, through similar means, what mechanism(s) can be used by relying parties to obtain revocation status information about certificates issued by that authority. This Directory Specification defines a Certificate Revocation List (CRL) mechanism but does not preclude the use of alternative mechanisms. One such alternative mechanism is the Online Certificate Status Protocol (OCSP) specified in IETF RFC 2560¹⁾. Using this protocol, a relying party (client) requests the revocation status of a certificate from an OCSP server. The server may use CRLs, or other mechanisms to check the status of the certificate and respond to the client accordingly. If OCSP can be used by relying parties to check the status of a certificate, IETF RFC 5280 contains a certificate extension (Authority Info Access) that would be included in such certificates and would provide sufficient information to access an appropriate OCSP server. Relying parties check revocation status information, as appropriate, for all certificates considered during the path processing procedure described in clause 10 and the delegation path processing procedure described in clause 16 to validate a certificate.

Only a CA that is authorized to issue CRLs may choose to delegate that authority to another entity. If this delegation is done, it shall be verifiable at the time of certificate/CRL verification. The **cRLDistributionPoints** extension can be used for this purpose. The **cRLIssuer** field of this extension would be populated with the name(s) of any entities, other than the certificate issuer itself, that have been authorized to issue CRLs concerning the revocation status of the certificate in question.

Certificates, including public-key certificates, as well as attribute certificates, shall have a lifetime associated with them, at the end of which they expire. In order to provide continuity of service, the authority shall ensure timely availability of replacement certificates to supersede expired/expiring certificates. Revocation notice date is the date/time that a revocation notice for a certificate first appears on a CRL, regardless of whether it is a base or dCRL. In the CRL, revocation notice date is the value contained in the **thisUpdate** field. Revocation date is the date/time the CA actually revoked the certificate, which could be different from the first time it appears on a CRL. In the CRL, revocation date is the value contained in the **revocationDate** component. Invalidity date is the date/time at which it is known or suspected that the private key was compromised or that the certificate should otherwise be considered invalid. This date may be earlier than the revocation date. In the CRL, invalidity date is the value contained in the **invalidityDate** entry extension.

Two related points are:

- Validity of certificates may be designed so that each becomes valid at the time of expiry of its predecessor, or an overlap may be allowed. The latter prevents the authority from having to install and distribute a large number of certificates that may run out at the same expiration date.
- Expired certificates will normally be removed from the Directory. It is a matter for the security policy and responsibility of the authority to keep old certificates for a period if a non-repudiation of data service is provided.

Certificates may be revoked prior to their expiration time, e.g., if the user's private key is assumed to be compromised, the user is no longer to be certified by the CA, or if the CA's certificate is assumed to be compromised. The revocation of an end-entity public-key certificate or a CA-certificate shall be made known by the CA, and a new certificate shall be made available, if appropriate. The CA may then inform the holder of the certificate about its revocation by an offline procedure.

¹⁾ IETF RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol (OCSP)*, June 1999

An authority that issues and subsequently revokes certificates:

- a) may be required to maintain an audit record of its revocation events for all certificate types issued by that authority (e.g., public-key certificates, attribute certificates issued to end-entities, as well as other authorities);
- b) shall provide revocation status information to relying parties using CRLs, Online Certificate Status Protocol (OCSP) or another mechanism for the publication of revocation status information;
- c) if using CRLs, it shall maintain and publish CRLs even if the lists of revoked certificates are empty;
- d) if using only partitioned CRLs, it shall issue a full set of partitioned CRLs covering the complete set of certificates whose revocation status will be reported using the CRL mechanism. Thus, the complete set of partitioned CRLs shall be equivalent to a full CRL for the same set of certificates, if the CRL issuer was not using partitioned CRLs.

Relying parties may use a number of mechanisms to locate revocation status information provided by an authority. For example, there may be a pointer in the certificate itself that directs the relying party to a location where revocation information is provided. There may be a pointer in a revocation list that redirects the relying party to a different location. The relying party may locate revocation information in a repository (e.g., a directory) or through other means outside the scope of this Directory Specification (e.g., locally configured).

The maintenance of Directory entries affected by the authority's revocation lists is the responsibility of the Directory and its users, acting in accordance with the security policy. For example, the user may modify its object entry by replacing the old certificate with a new one. The latter shall then be used to authenticate the user to the Directory.

If revocation lists are published in the Directory, they are held within entries as attributes of the following types:

- `certificateRevocationList`;
- `authorityRevocationList`;
- `deltaRevocationList`;
- `attributeCertificateRevocationList`; and
- `attributeAuthorityRevocationList`.

```
CertificateList ::= SIGNED{CertificateListContent}
```

```
CertificateListContent ::= SEQUENCE {  
  version          Version OPTIONAL,  
  -- if present, version shall be v2  
  signature        AlgorithmIdentifier{{SupportedAlgorithms}},  
  issuer           Name,  
  thisUpdate       Time,  
  nextUpdate       Time OPTIONAL,  
  revokedCertificates SEQUENCE OF SEQUENCE {  
    serialNumber    CertificateSerialNumber,  
    revocationDate  Time,  
    crlEntryExtensions Extensions OPTIONAL,  
    ...} OPTIONAL,  
  ...,  
  ...,  
  crlExtensions    [0] Extensions OPTIONAL }
```

[AC₁] The **version** field shall indicate the version of the encoded revocation list. If the **extensions** component is present in the revocation list, the version shall be **v2**. If the extensions component is not present, the version shall either be absent or present as **v2**.

NOTE 1 – In the first and the second editions of this specification, the version component was always absent. In the third, fourth, fifth and sixth editions of this specification, the version shall be v2, if the extensions component flagged as critical is present in the revocation list. Or the version may either be absent or present as v2, if no extensions component flagged as critical is present in the revocation list. **[AC₁]**

The **signature** field shall contain the algorithm identifier for the algorithm used by the authority to sign the revocation list. It shall be the same value as used in the **algorithmIdentifier** component of the **SIGNATURE** data type when signing the revocation list.

NOTE 2 – This field is redundant.

The **issuer** field shall identify the entity that has signed and issued the revocation list.

The **thisUpdate** field shall indicate the date/time at which this revocation list was issued.

The **nextUpdate** field, if present, shall indicate the date/time by which the next revocation list in this series will be issued. The next revocation list could be issued before the indicated date, but it shall not be issued any later than the indicated time.

The **revokedCertificates** field shall identify certificates that have been revoked. The revoked certificates are identified by their serial numbers. If none of the certificates covered by this CRL has been revoked, it is strongly recommended that the **revokedCertificates** parameter be omitted from the CRL, rather than being included with an empty **SEQUENCE**.

The **crlExtensions** field, if present, shall contain one or more CRL extensions.

NOTE 3 – The checking of the entire list of certificates is a local matter. The list shall not be assumed to be in any particular order unless specific ordering rules have been specified by the issuing authority, e.g., in that authority's policy.

NOTE 4 – If a non-repudiation of data service is dependent on keys provided by the authority, the service should ensure that all relevant keys of the authority (revoked or expired) and the time stamped revocation lists are archived and certified by a current authority.

[AC₁] Text deleted **[AC₁]**

When an implementation processing a CRL encounters the serial number of the certificate of interest in a CRL entry, but does not recognize a critical extension in the **crlEntryExtensions** field from that CRL entry, that CRL cannot be used to determine the status of the certificate. When an implementation does not recognize a critical extension in the **crlExtensions** field, that CRL cannot be used to determine the status of the certificate, regardless of whether the serial number of the certificate of interest appears in that CRL or not.

NOTE 5 – In these cases, local policy may dictate actions in addition to and/or stronger than those stated in this Directory Specification, such as seeking revocation status information from other sources.

Certificates for which revocation status cannot be determined should not be considered valid certificates.

If an extension affects the treatment of the list (e.g., multiple CRLs need to be scanned to examine the entire list of revoked certificates, or an entry may represent a range of certificates), then either that extension or a related extension shall be indicated as critical in the **crlExtensions** field. Therefore, a critical extension in the **crlEntryExtensions** field of an entry shall affect only the certificate specified in that entry, unless there is a related critical extension in the **crlExtensions** field that advertises a special treatment for it. The only example of this situation defined in this Directory Specification is the **certificateIssuer** CRL entry extension and the related **issuingDistributionPoint** CRL extension when the **indirectCRL** Boolean from that extension is set to **TRUE**.

NOTE 6 – Standard extensions for CRLs are defined in clause 8 of this Directory Specification.

If unknown elements appear within the extension, and the extension is not marked critical, those unknown elements shall be ignored according to the rules of extensibility documented in clause 12.2.2 of Rec. ITU-T X.519 | ISO/IEC 9594-5.

7.11 Repudiation of a digital signing

Participants in an event may subsequently decide to repudiate anything that they digitally signed in that event. For example, one can dispute one's participation in a key establishment or being the originator of a signed email message as easily as one can dispute one's signing of a document with the intent to be bound to the content of that document. The repudiation may not be successful. The Non-repudiation Framework, Rec. ITU-T X.813 | ISO/IEC 10181-4, describes a dispute resolution process as follows:

- 1) evidence generation;
- 2) evidence transfer, storage and retrieval;
- 3) evidence verification; and
- 4) dispute resolution.

The generated evidence may include, but is not limited to:

- audit records pertinent to the event and an assertion of intent;
- statements made by third party notaries;
- policy statements;
- digitally signed information, including audit records and notary statements;
- timestamps of the digitally signed information;
- the certificates supporting the digital signature;
- the appropriate revocation information published and available at the time of the disputed event; and,
- any certificate revocations subsequent to the time of the event which indicate a key compromise occurred before the time of the event.

The integrity of stored data that might be presented as evidence may be maintained in a variety of ways, e.g., access control, storage of hashes by a trusted third party, digital signature. It may also be necessary periodically to strengthen the protection of that stored data to counteract improvements in computer processing and/or crypto-analysis.

NOTE – Neither the type and amount of evidence generated nor the level of integrity is specified by this Directory Specification. However, it is expected that the level of effort will be commensurate with the risk involved.

Evidence verification may require the revalidation of the digital signatures of data, e.g., messages, documents, certificates, CRLs, and timestamps that were used in the initial validation process. The fact that a certificate has expired shall not preclude its use for revalidating signatures created during the validity period of that certificate. A certificate that has been revoked may be used if it can be determined that the certificate was valid at the time of the disputed event.

Even if all the digital evidence described above is considered technically valid, other conditions, e.g., the intent, understanding or competence of the signer, may allow the signer successfully to repudiate it.

8 Public-key certificate and CRL extensions

The certificate extensions defined in this clause are for use with public-key certificates, unless otherwise stated. Extensions for use with attribute certificates are defined in clause 15. CRL extensions defined in this clause may be used in CRLs, CARLs and also for ACRLs and AARLs defined in clause 17.

This clause specifies extensions in the following areas:

- a) Key and policy information: These certificate and CRL extensions convey additional information about the keys involved, including key identifiers for subject and issuer keys, indicators of intended or restricted key usage, and indicators of certificate policy.
- b) Subject and issuer attributes: These certificate and CRL extensions support alternative names, of various name forms, for a certificate subject, a certificate issuer, or a CRL issuer. These extensions can also convey additional attribute information about the certificate subject, to assist a relying party in being confident that the certificate subject is a particular person or entity.
- c) Certification path constraints: These certificate extensions allow constraint specifications to be included in CA-certificates, i.e., certificates for CAs issued by other CAs, to facilitate the automated processing of certification paths when multiple certificate policies are involved. Multiple certificate policies arise when policies vary for different applications in an environment or when interoperability with external environments occurs. The constraints may restrict the types of certificates that can be issued by the subject CA or that may occur subsequently in a certification path.
- d) Basic CRL extensions: These CRL extensions allow a CRL to include indications of revocation reason, to provide for temporary suspension of a certificate, and to include CRL-issue sequence numbers to allow relying parties to detect missing CRLs in a sequence from one CRL issuer.
- e) CRL distribution points and delta-CRLs: These certificate and CRL extensions allow the complete set of revocation information from one CA to be partitioned into separate CRLs and allow revocation information from multiple CAs to be combined in one CRL. These extensions also support the use of partial CRLs indicating only changes since an earlier CRL issue.

Inclusion of any extension in a certificate or CRL is at the option of the authority issuing that certificate or CRL.

In a certificate or CRL, an extension is flagged as being either critical or non-critical. If an extension is flagged critical and a relying party does not recognize the extension field type or does not implement the semantics of the extension, then that relying party shall consider the certificate invalid. If an extension is flagged non-critical, a relying party that does not recognize or implement that extension type may process the remainder of the certificate ignoring the extension. If an extension is flagged non-critical, a relying party that does recognize the extension, shall process the extension. Extension type definitions in this Directory Specification indicate if the extension is always critical, always non-critical, or if criticality can be decided by the certificate or CRL issuer. The reason for requiring some extensions to be always non-critical is to allow relying parties which do not need to use such extensions to omit support for them without jeopardizing the ability to interoperate with all CAs.

NOTE – A relying party may require certain non-critical extensions to be present in a certificate in order for that certificate to be considered acceptable. The need for inclusion of such extensions may be implied by local policy rules of the relying party or may be a CA policy rule indicated to the relying party by inclusion of a particular certificate policy identifier in the certificate policies extension with that extension being flagged critical.

For all certificate extensions, CRL extensions, and CRL entry extensions defined in this Directory Specification, there shall be no more than one instance of each extension type in any certificate, CRL, or CRL entry, respectively.

8.1 Policy handling

8.1.1 Certificate policy

This framework contains three types of entity: the relying party, the CA and the certificate subject (or end-entity). Each entity operates under obligations to the other two entities and, in return, enjoys limited warranties offered by them. These obligations and warranties are defined in a certificate policy. A certificate policy is a document (usually in plain-language). It may be referenced by an object identifier, which may be included in the certificate policies extension of the certificate issued by the CA, to the end-entity and upon which the relying party relies. A certificate may be issued in accordance with one or more than one policy. The definition of the policy and assignment of the identifier is performed by a policy authority. The set of policies administered by a policy authority is called a policy domain. All certificates are issued in accordance with a policy, even if the policy is neither recorded anywhere nor referenced in the certificate. This Directory Specification does not prescribe the style or contents of the certificate policy.

The relying party may be bound to its obligations under the certificate policy by the act of importing an authority public key and using it as trust anchor information, or by relying on a certificate that includes the associated policy identifier. The CA may be bound to its obligations under the policy by the act of issuing a certificate that includes the associated policy identifier. The end-entity may be bound to its obligations under the policy by the act of requesting and accepting a certificate that includes the associated policy identifier and by using the corresponding private key. Implementations that do not use the certificate policies extension should achieve the required binding by other means.

For an entity simply to declare conformance to a policy does not generally satisfy the assurance requirements of the other entities in the framework. They require some reason to believe that the other parties operate a reliable implementation of the policy. However, if explicitly stated in the policy, relying parties may accept the CA's assurances that its end-entities agree to be bound by their obligations under the policy, without having to confirm this directly with them. This aspect of certificate policy is outside the scope of this Directory Specification.

A CA may place limitations on the use of its certificates, in order to control the risk that it assumes as a result of issuing certificates. For instance, it may restrict the community of relying parties, the purposes for which they may use its certificates and/or the type and extent of damages that it is prepared to make good in the event of a failure on its part, or that of its end-entities. These matters should be defined in the certificate policy.

Additional information, to help affected entities understand the provisions of the policy, may be included in the certificate policies extension in the form of policy qualifiers.

8.1.2 Cross-certificates and policy handling

The warranties and obligations shared by the subject CA, the intermediate CA and the relying party are defined by the certificate policy identified in the cross-certificate, in accordance with which the subject CA may act as, or on behalf of, an end-entity. The warranties and obligations shared by the certificate subject, the subject CA and the intermediate CA are defined by the certificate policy identified in the end-entity's certificate, in accordance with which the intermediate CA may act as, or on behalf of, a relying party.

A certification path is said to be valid under the set of policies that are common to all public-key certificates in the path.

In addition to the situation described above, there are two special cases to be considered:

- a) the CA does not use the certificate policies extension to convey its policy requirements to relying parties; and
- b) the relying party or intermediate CA delegates the job of controlling policy to the next CA in the path.

In the first case, the public-key certificate should not contain a certificate policies extension at all. As a result, the set of policies under which the path is valid will be null. But, the path may be valid nonetheless. Relying parties shall still ensure that they are using the public-key certificate in conformance with the policies of the CAs in the path.

In the second case, the relying party or intermediate CA should include the special value *any-policy* in the *initial-policy-set* or cross-certificate. Where a public-key certificate includes the special value *any-policy*, it should not include any other certificate policy identifiers. The identifier *any-policy* should not have any associated policy qualifiers.

The relying party can ensure that all its obligations are conveyed by setting the *initial-explicit-policy* indicator. In this way, only authorities that use the certificate policies extension as their way of achieving binding are accepted in the path, and relying parties have no additional obligations. Because CAs also attract obligations when they act as, or on behalf of, a relying party, they can ensure that all their obligations are conveyed by setting **requireExplicitPolicy** component of the **policyConstraints** extension in the cross-certificate.

8.1.3 Policy mapping

Some certification paths may cross boundaries between policy domains. The warranties and obligations according to which the cross-certificate is issued may be materially equivalent to some or all of the warranties and obligations

according to which the subject CA issues certificates to end-entities, even though the policy authorities under which the two CAs operate may have selected different object identifiers for these materially equivalent policies. In this case, the intermediate CA may include a policy mappings extension in the cross-certificate. In the policy mappings extension, the intermediate CA assures the relying party that it will continue to enjoy the familiar warranties, and that it should continue to fulfil its familiar obligations, even though subsequent entities in the certification path operate in a different policy domain. The intermediate CA should include one or more mappings for each of a subset of the policies under which it issued the cross-certificate, and it should not include mappings for any other policies. If one or more of the certificate policies according to which the subject CA operates is identical to those according to which the intermediate CA operates (i.e., it has the same unique identifier), then these identifiers should be excluded from the policy mapping extension, but included in the certificate policies extension.

Policy mapping has the effect of converting all policy identifiers in certificates further down the certification path to the identifier of the equivalent policy, as recognized by the relying party.

Policies shall not be mapped either to or from the special value *any-policy*.

Relying parties may determine that public-key certificates issued in a policy domain other than its own should not be relied upon, even though a trusted intermediate CA may determine its policy to be materially equivalent to its own. It can do this by setting the *initial-policy-mapping-inhibit* input to the path validation procedure. Additionally, an intermediate CA may make a similar determination on behalf of its relying parties. In order to ensure that relying parties correctly enforce this requirement, it can set `inhibitPolicyMapping` in a `policyConstraints` extension.

8.1.4 Certification path processing

The relying party faces a choice between two strategies:

- a) it can require that the certification path be valid under at least one of a set of policies pre-determined by the user; or
- b) it can ask the path validation module to report the set of policies for which the certification path is valid.

The first strategy may be most appropriate when the relying party knows, *a priori*, the set of policies that are acceptable for its intended use.

The second strategy may be the most appropriate when the relying party does not know, *a priori*, the set of policies that are acceptable for its intended use.

In the first instance, the certification path validation procedure will indicate the path to be valid only if it is valid under one or more of the policies specified in the *initial-policy-set*, and it will return the sub-set of the *initial-policy-set* under which the path is valid. In the second instance, the certification path validation procedure may indicate that the path is invalid under the *initial-policy-set*, but valid under a disjoint set: the *authorities-constrained-policy-set*. Then the relying party shall determine whether its intended use of the certificate is consistent with one or more of the certificate policies under which the path is valid. By setting the *initial-policy-set* to *any-policy*, the relying party can cause the procedure to return a valid result if the path is valid under any (unspecified) policy.

8.1.5 Self-issued certificates

A CA may issue a certificate to itself under three circumstances:

- a) as a convenient way of encoding the public key associated with the private key used to sign the certificate, so that it can be communicated to, and stored as trust anchor information by, its certificate-using systems;
- b) for certifying additional public keys of the CA used for purposes other than those covered by category a) (such as OCSP and possibly CRL signing); and
- c) for replacing its own expired CA-certificates.

These types of CA-certificates are called self-issued certificates, and they can be recognized by the fact that the issuer and subject names present in them are identical. For the purposes of path validation, self-issued certificates of category a) are self-signed certificates and are therefore verified with the public key contained in them, and if they are encountered in the path, they shall be ignored.

Self-issued certificates of type b) may only appear as end certificates in a path, and shall be processed as end certificates.

Self-issued certificates of type c) (also known as self-issued intermediate certificates) may appear as intermediate certificates in a path. As a matter of good practice, when replacing a key that is on the point of expiration, a CA should request the issuance of any in-bound cross-certificates that it requires for its replacement public key before using the key. Nevertheless, if self-issued certificates of this category are encountered in the path, they shall be processed as intermediate certificates, with the following exception: they do not contribute to the path length for the purposes of

processing the `pathLenConstraint` component of the `basicConstraints` extension and the *skip-certificates* values associated with the *policy-mapping-inhibit-pending* and *explicit-policy-pending* indicators.

If an authority uses the same key to sign certificates and CRLs, a single self-issued certificate of category a) shall be used. If an authority uses a different key to sign CRLs than that used to sign certificates, the authority may choose to issue two self-issued certificates of category a), one for each of the keys. In this situation, relying parties would need access to both self-issued certificates to establish separate trust anchors for certificates and CRLs signed by that authority. Alternatively, an authority may issue one self-issued certificate of category a) for certificate signing and one self-issued certificate of category b) for CRL signing. In this situation, relying parties use the key certified in the certificate of category a) as their single trust anchor for both certificates and CRLs signed by that authority. In this case, if the self-issued certificate of category b) were to be used to verify signatures on CRLs, there is no means defined in this standard to check the validity of that certificate.

If self-issued certificates of category b) are encountered within a path, they shall be ignored.

NOTE – Other mechanisms for distributing CA public keys are outside the scope of this Directory Specification.

8.2 Key and policy information extensions

8.2.1 Requirements

The following requirements relate to key and policy information:

- a) CA key pair updating can occur at regular intervals or in special circumstances. There is a need for a certificate field to convey an identifier of the public key to be used to verify the certificate signature. A relying party can use such identifiers in finding the correct CA-certificate for validating the certificate issuer's public key.
- b) In general, a certificate subject has different public keys and, correspondingly, different certificates for different purposes, e.g., digital signature and encipherment key agreement. A certificate field is needed to assist a relying party in selecting the correct certificate for a given subject for a particular purpose or to allow a CA to stipulate that a certified key may only be used for a particular purpose.
- c) Subject key pair updating can occur at regular intervals or in special circumstances. There is a need for a certificate field to convey an identifier to distinguish between different public keys for the same subject used at different points in time. A relying party can use such identifiers in finding the correct certificate.
- d) The private key corresponding to a certified public key is typically used over a different period from the validity of the public key. With digital signature keys, the usage period for the signing private key is typically shorter than that for the verifying public key. The validity period of the certificate indicates a period for which the public key may be used, which is not necessarily the same as the usage period of the private key. In the event of a private key compromise, the period of exposure can be limited if the relying party knows the legitimate use period for the private key. There is therefore a requirement to be able to indicate the usage period of the private key in a public-key certificate.
- e) Because certificates may be used in environments where multiple certificate policies apply, provision needs to be made for including certificate policy information in certificates.
- f) When cross-certifying from one organization to another, it can sometimes be agreed that certain of the two organizations' policies can be considered equivalent. A CA-certificate needs to allow the certificate issuer to indicate that one of its own certificate policies is equivalent to another certificate policy in the subject CA's domain. This is known as policy mapping.
- g) A user of an encipherment or digital signature system which uses certificates defined in this Directory Specification needs to be able to determine in advance the algorithms supported by other users.

8.2.2 Public-key certificate and CRL extension fields

The following extension fields are defined:

- a) Authority key identifier;
- b) Subject key identifier;
- c) Key usage;
- d) Extended key usage;
- e) Private key usage period;
- f) Certificate policies;
- g) Policy mappings.

These extension fields shall be used only as certificate extensions, except for authority key identifier which may also be used as a CRL extension. Unless otherwise noted, these extensions may be used in both CA-certificates and end-entity certificates.

8.2.2.1 Authority key identifier extension

[AC₁] This extension, which may be used as either a public-key certificate extension or CRL extension, identifies the public key to be used to verify the signature on this public-key certificate or CRL. It enables distinct keys used by the same CA used for signing public-key certificates to be distinguished (e.g., as key updating occurs) and it enables distinct keys used by the same CRL issuer to be distinguished. This extension is defined as follows: **[AC₁]**

```
authorityKeyIdentifier EXTENSION ::= {  
    SYNTAX          AuthorityKeyIdentifier  
    IDENTIFIED BY   id-ce-authorityKeyIdentifier }  
  
AuthorityKeyIdentifier ::= SEQUENCE {  
    keyIdentifier          [0] KeyIdentifier OPTIONAL,  
    authorityCertIssuer    [1] GeneralNames OPTIONAL,  
    authorityCertSerialNumber [2] CertificateSerialNumber OPTIONAL,  
    ... }  
(WITH COMPONENTS { ..., authorityCertIssuer      PRESENT,  
                        authorityCertSerialNumber PRESENT } |  
  WITH COMPONENTS { ..., authorityCertIssuer      ABSENT,  
                        authorityCertSerialNumber ABSENT } )  
  
KeyIdentifier ::= OCTET STRING
```

The key may be identified by an explicit key identifier in the **keyIdentifier** component, by the identification of a certificate for the key (giving certificate issuer in the **authorityCertIssuer** component and certificate serial number in the **authorityCertSerialNumber** component), or by both explicit key identifier and identification of a certificate for the key. If both forms of identification are used then the certificate or CRL issuer shall ensure they are consistent. A key identifier shall be unique with respect to all key identifiers for the issuing authority for the certificate or CRL containing the extension. An implementation which supports this extension is not required to be able to process all name forms in the **authorityCertIssuer** component. (See clause 8.3.2.1 for details of the **GeneralNames** type.)

Certification authorities shall assign certificate serial numbers such that every (issuer, certificate serial number) pair uniquely identifies a single certificate. The **keyIdentifier** form can be used to select CA-certificates during path construction. The **authorityCertIssuer**, **authoritySerialNumber** pair can only be used to provide preference to one certificate over others during path construction.

This extension is always non-critical.

8.2.2.2 Subject key identifier extension

This field identifies the public key being certified. It enables distinct keys used by the same subject to be differentiated (e.g., as key updating occurs). This field is defined as follows:

```
subjectKeyIdentifier EXTENSION ::= {  
    SYNTAX          SubjectKeyIdentifier  
    IDENTIFIED BY   id-ce-subjectKeyIdentifier }  
  
SubjectKeyIdentifier ::= KeyIdentifier
```

A key identifier shall be unique with respect to all key identifiers for the subject with which it is used. This extension is always non-critical.

8.2.2.3 Key usage extension

This field identifies the intended usage for which the certificate has been issued. The intended usage may be further constrained by policy. This policy may be stated in a certificate policy definition, a contract, or other specification. However, a policy shall not override the constraint indicated by a **KeyUsage** bit, e.g., a certificate policy could not allow a certificate to be used for digital signature if **KeyUsage** indicated that it could only be used for key agreement.

Setting a specific value of **KeyUsage** in a certificate does not in itself signal for an instance of communication that the communicating parties are acting in accordance with this setting, e.g., when signing a document. The definition of methods by which parties may signal their intent for a specific instance of communication (e.g., commitment to content for that specific instance) is outside the scope of this Directory Specification, but it is anticipated that multiple methods will exist. Although not recommended, it is possible to use the content of the certificate, e.g., certificate policy, to signal the intent of the signing. However, since that signal was made when the certificate was issued by the CA, such use may not meet the requirement that declaring the intent is made at the time of signing by the signer.

More than one bit may be set in an instance of the **keyUsage** extension. The setting of multiple bits shall not change the meaning of each individual bit but shall indicate that the certificate may be used for all of the purposes indicated by the set bits. There may be risks incurred when setting multiple bits. A review of those risks is documented in Annex I.

This field is defined as follows:

AC1 *Text deleted* AC1

```
keyUsage EXTENSION ::= {
  SYNTAX          KeyUsage
  IDENTIFIED BY   id-ce-keyUsage }
```

```
KeyUsage ::= BIT STRING {
  digitalSignature (0),
  contentCommitment (1),
  keyEncipherment (2),
  dataEncipherment (3),
  keyAgreement (4),
  keyCertSign (5),
  cRLSign (6),
  encipherOnly (7),
  decipherOnly (8) }
```

Bits in the **KeyUsage** type are as follows:

- a) **digitalSignature**: for verifying digital signatures that are used with an entity authentication service, a data origin authentication service and/or an integrity service;
- b) **contentCommitment**: for verifying digital signatures which are intended to signal that the signer is committing to the content being signed. The type of commitment the certificate can be used to support may be further constrained by the CA, e.g., through a certificate policy. The precise type of commitment of the signer e.g., "reviewed and approved" or "with the intent to be bound", may be signalled by the content being signed, e.g., the signed document itself or some additional signed information.

Since a content commitment signing is considered a digitally signed transaction, the **digitalSignature** bit need not be set in the certificate. If it is set, it does not affect the level of commitment the signer has endowed in the signed content.

Note that it is not incorrect to refer to this **keyUsage** bit using the identifier **nonRepudiation**. However, the use of this identifier has been deprecated. Regardless of the identifier used, the semantics of this bit are as specified in this Directory Specification;

- c) **keyEncipherment**: for enciphering keys or other security information, e.g., for key transport;
- d) **dataEncipherment**: for enciphering user data, but not keys or other security information as in c) above;
- e) **keyAgreement**: for use as a public key agreement key;
- f) **keyCertSign**: for verifying a CA's signature on certificates.

Since certificate signing is considered a commitment to the content of the certificate by the CA, neither the **digitalSignature** bit nor the **contentCommitment** bit need be set in the certificate. If either (or both) is set, it does not affect the level of commitment the signer has endowed in the signed certificate;

- g) **cRLSign**: for verifying an authority's signature on CRLs.

Since CRL signing is considered to be commitment to the content of the CRL by the CRL issuer, neither the **digitalSignature** bit nor the **contentCommitment** bit need be set in the certificate. If either (or both) is set, it does not affect the level of commitment the signer has endowed in the signed CRL;

- h) **encipherOnly**: public key agreement key for use only in enciphering data when used with **keyAgreement** bit also set (meaning with other key usage bit set is undefined);
- i) **decipherOnly**: public key agreement key for use only in deciphering data when used with **keyAgreement** bit also set (meaning with other key usage bit set is undefined).

Application specifications should indicate which of the **digitalSignature** or **contentCommitment** bits are appropriate for their use. If a signing application has no knowledge of the signer's intent regarding commitment to content, the application shall sign and support that signing with a certificate that has the **digitalSignature** bit set in that certificate's **keyUsage** extension.

Even though a digital signature was verified using a certificate that has only the **digitalSignature** bit set, other factors external to the verification of the digital signature may also play a role in determining the intent of the signing.

Conversely, even though a digital signature was verified using a certificate that has only the **contentCommitment** bit set, external factors may be used by the signer to disclaim commitment to the signed content.

The bit **keyCertSign** is for use in CA-certificates only. If **KeyUsage** is set to **keyCertSign**, the value of the **cA** component of the **basicConstraints** extension shall be set to **TRUE**. CAs may also use other defined key usage bits in **KeyUsage**, e.g., **digitalSignature** for providing authentication and integrity of online administration transactions.

This extension may, at the option of the certificate issuer, be either critical or non-critical.

If the extension is flagged critical or if the extension is flagged non-critical but the relying party recognizes it, then the certificate shall be used only for a purpose for which the corresponding key usage bit is set to one. If the extension is flagged non-critical and the relying party does not recognize it, then this extension shall be ignored.

A bit set to zero indicates that the key is not intended for that purpose. If the extension is present with all bits set to zero, the key is intended for a purpose other than those listed above.

8.2.2.4 Extended key usage extension

This field indicates one or more purposes for which the certified public key may be used, in addition to, or in place of the basic purposes indicated in the key usage extension field. This field is defined as follows:

```
extKeyUsage EXTENSION ::= {  
    SYNTAX          SEQUENCE SIZE (1..MAX) OF KeyPurposeId  
    IDENTIFIED BY   id-ce-extKeyUsage }
```

KeyPurposeId ::= OBJECT IDENTIFIER

A CA may assert any-extended-key-usage by using the **anyExtendedKeyUsage** object identifier. This enables a CA to issue a certificate that contains **KeyPurposeId** object identifiers for extended key usages that may be required by certificate-using applications, without restricting the certificate to only those key usages. If extended key usage would restrict key usage, then the inclusion of this object identifier removes that restriction.

anyExtendedKeyUsage OBJECT IDENTIFIER ::= { id-ce-extKeyUsage 0 }

Key purposes may be defined by any organization with a need. Object identifiers used to identify key purposes shall be assigned in accordance with Rec. ITU-T X.660 | ISO/IEC 9834-1.

This extension may, at the option of the issuing CA, be either critical or non-critical.

If the extension is flagged critical, then the certificate shall be used only for one of the purposes indicated.

If the extension is flagged non-critical, then it indicates the intended purpose or purposes of the key, and may be used in finding the correct key/certificate of an entity that has multiple keys/certificates. If this extension is present, and the relying party recognizes and processes the **extendedKeyUsage** extension type, then the relying party shall ensure that the certificate shall be used only for one of the purposes indicated. (Using applications may nevertheless require that a particular purpose be indicated in order for the certificate to be acceptable to that application.)

If a certificate contains both a critical key usage field and a critical extended key usage field, then both fields shall be processed independently and the certificate shall only be used for a purpose consistent with both fields. If there is no purpose consistent with both fields, then the certificate shall not be used for any purpose.

This Directory Specification defines the following key purpose that can be included in the extended key usage extension. Other purposes that can also be included are defined in other specifications, such as IETF RFC 5280.

keyPurposes OBJECT IDENTIFIER ::= {id-kp 1}

8.2.2.5 Private key usage period extension

This field indicates the period of use of the private key corresponding to the certified public key. It is applicable only for digital signature keys. This field is defined as follows:

```
privateKeyUsagePeriod EXTENSION ::= {  
    SYNTAX          PrivateKeyUsagePeriod  
    IDENTIFIED BY   id-ce-privateKeyUsagePeriod }
```

```
PrivateKeyUsagePeriod ::= SEQUENCE {  
    notBefore [0] GeneralizedTime OPTIONAL,  
    notAfter [1] GeneralizedTime OPTIONAL,  
    ... }  
(WITH COMPONENTS {..., notBefore PRESENT } |
```

WITH COMPONENTS { ..., notAfter PRESENT })

The **notBefore** component indicates the earliest date and time at which the private key could be used for signing. If the **notBefore** component is not present, then no information is provided as to when the period of valid use of the private key commences. The **notAfter** component indicates the latest date and time at which the private key could be used for signing. If the **notAfter** component is not present then no information is provided as to when the period of valid use of the private key concludes.

This extension is always non-critical.

NOTE 1 – The period of valid use of the private key may be different from the certified validity of the public key as indicated by the certificate validity period. With digital signature keys, the usage period for the signing private key is typically shorter than that for the verifying public key.

NOTE 2 – If the verifier of a digital signature wants to check that the certificate has not been revoked, for example, due to key compromise, up to the time of verification, then a valid certificate will still exist for the public key at verification time. After the certificate(s) for a public key have expired, a signature verifier cannot rely on compromises being notified via CRLs.

8.2.2.6 Certificate policies extension

This field lists certificate policies, recognized by the issuing CA, that apply to the certificate, together with optional qualifier information pertaining to these certificate policies. The list of certificate policies is used in determining the validity of a certification path, as described in clause 10. The optional qualifiers are not used in the certification path processing procedure, but relevant qualifiers are provided as an output of that process to the certificate using application to assist in determining whether a valid path is appropriate for the particular transaction. Typically, different certificate policies will relate to different applications which may use the certified key. The presence of this extension in an end-entity certificate indicates the certificate policies for which this certificate is valid. The presence of this extension in a certificate issued by one CA to another CA indicates the certificate policies for which certification paths containing this certificate may be valid. This field is defined as follows:

```
certificatePolicies EXTENSION ::= {
  SYNTAX          CertificatePoliciesSyntax
  IDENTIFIED BY   id-ce-certificatePolicies }

CertificatePoliciesSyntax ::= SEQUENCE SIZE (1..MAX) OF PolicyInformation

PolicyInformation ::= SEQUENCE {
  policyIdentifier  CertPolicyId,
  policyQualifiers SEQUENCE SIZE (1..MAX) OF PolicyQualifierInfo OPTIONAL,
  ... }

CertPolicyId ::= OBJECT IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {
  policyQualifierId  CERT-POLICY-QUALIFIER.&id({SupportedPolicyQualifiers}),
  qualifier          CERT-POLICY-QUALIFIER.&Qualifier
                    ({SupportedPolicyQualifiers}{@policyQualifierId}) OPTIONAL,
  ... }

SupportedPolicyQualifiers CERT-POLICY-QUALIFIER ::= {...}
```

A value of the **PolicyInformation** type identifies and conveys qualifier information for one certificate policy. The component **policyIdentifier** contains an identifier of a certificate policy and the component **policyQualifiers** contains policy qualifier values for that element.

This extension may, at the option of the certificate issuer, be either critical or non-critical.

If the extension is flagged critical, it indicates that the certificate shall only be used for the purpose, and in accordance with the rules implied by one of the indicated certificate policies. The rules of a particular policy may require the relying party to process the qualifier value in a particular way.

If the extension is flagged non-critical, use of this extension does not necessarily constrain use of the certificate to the policies listed. However, a relying party may require a particular policy to be present in order to use the certificate (see clause 10). Policy qualifiers may, at the option of the relying party, be processed or ignored.

Certificate policies and certificate policy qualifier types may be defined by any organization with a need. Object identifiers used to identify certificate policies and certificate policy qualifier types shall be assigned in accordance with Rec. ITU-T X.660 | ISO/IEC 9834-1. A CA may assert any-policy by using the **anyPolicy** object identifier in order to trust a certificate for all possible policies. Because of the need for identification of this special value to apply regardless of the application or environment, that object identifier is assigned in this Directory Specification. No object identifiers will be assigned in this Directory Specification for specific certificate policies. That assignment is the responsibility of the entity that defines the certificate policy.

anyPolicy OBJECT IDENTIFIER ::= {id-ce-certificatePolicies 0}

The identifier **anyPolicy** should not have any associated policy qualifiers.

The following ASN.1 object class is used in defining certificate policy qualifier types:

```
CERT-POLICY-QUALIFIER ::= CLASS {  
    &id                OBJECT IDENTIFIER UNIQUE,  
    &Qualifier          OPTIONAL }  
WITH SYNTAX {  
    POLICY-QUALIFIER-ID &id  
    [QUALIFIER-TYPE     &Qualifier] }
```

A definition of a policy qualifier type shall include:

- a statement of the semantics of the possible values; and
- an indication of whether the qualifier identifier may appear in a certificate policies extension without an accompanying value and, if so, the implied semantics in such a case.

NOTE – A qualifier may be specified as having any ASN.1 type. When the qualifier is anticipated to be used primarily with applications that do not have ASN.1 decoding functions, it is recommended that the type OCTET STRING be specified. The ASN.1 OCTET STRING value can then convey a qualifier value encoded according to any convention specified by the policy element defining organization.

8.2.2.7 Policy mappings extension

This field, which shall be used in CA-certificates only, allows a certificate issuer to indicate that, for the purposes of the user of a certification path containing this certificate, one of the issuer's certificate policies can be considered equivalent to a different certificate policy used in the subject CA's domain. This field is defined as follows:

```
policyMappings EXTENSION ::= {  
    SYNTAX          PolicyMappingsSyntax  
    IDENTIFIED BY   id-ce-policyMappings }  
  
PolicyMappingsSyntax ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {  
    issuerDomainPolicy CertPolicyId,  
    subjectDomainPolicy CertPolicyId,  
    ... }
```

The **issuerDomainPolicy** component indicates a certificate policy that is recognized in the issuing CA's domain and that can be considered equivalent to the certificate policy indicated in the **subjectDomainPolicy** component that is recognized in the subject CA's domain.

Policies shall not be mapped to or from the special value **anyPolicy**.

This extension may, at the option of the certificate issuer, be either critical or non-critical. It is recommended that it be critical, otherwise a relying party may not correctly interpret the stipulation of the issuing CA.

NOTE 1 – An example of policy mapping is as follows. The U.S. government domain may have a policy called Canadian Trade and the Canadian government may have a policy called U.S. Trade. While the two policies are distinctly identified and defined, there may be an agreement between the two governments to accept certification paths extending cross-border within the rules implied by these policies for relevant purposes.

NOTE 2 – Policy mapping implies significant administrative overheads and the involvement of suitably diligent and authorized personnel in related decision-making. In general, it is preferable to agree upon a more global use of common policies than it is to apply policy mapping. In the above example, it would be preferable for the U.S., Canada and Mexico to agree upon a common policy for North American Trade.

NOTE 3 – It is anticipated that policy mapping will be practical only in limited environments in which policy statements are very simple.

8.3 Subject and issuer information extensions

8.3.1 Requirements

The following requirements relate to certificate subject and certificate issuer attributes:

- a) Certificates need to be usable by applications that employ a variety of name forms, including Internet electronic mail names, Internet domain names, ITU-T X.400 originator/recipient addresses, and EDI party names. It is therefore necessary to be able securely to associate multiple names of a variety of name forms with a certificate subject or a certificate or CRL issuer.
- b) A relying party may need securely to know certain identifying information about a subject in order to have confidence that the subject is indeed the person or thing intended. For example, information such as postal address, position in a corporation, or a picture image may be required. Such information may be conveniently represented as directory attributes, but these attributes are not necessarily part of the distinguished name. A certificate field is therefore needed for conveying additional directory attributes beyond those in the distinguished name.

8.3.2 Certificate and CRL extension fields

The following extension fields are defined:

- a) Subject alternative name;
- b) Issuer alternative name;
- c) Subject directory attributes.

These fields shall be used only as public-key certificate extensions, except for issuer alternative name which may also be used as a CRL extension. As certificate extensions, they may be present in CA-certificates or end-entity public-key certificates.

8.3.2.1 Subject alternative name extension

This field contains one or more alternative names, using any of a variety of name forms, for the entity that is bound by the CA to the certified public key. This field is defined as follows:

```
subjectAltName EXTENSION ::= {
    SYNTAX          GeneralNames
    IDENTIFIED BY   id-ce-subjectAltName }

GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName ::= CHOICE {
    otherName          [0]  INSTANCE OF OTHER-NAME,
    rfc822Name         [1]  IA5String,
    dNSName            [2]  IA5String,
    x400Address        [3]  ORAddress,
    directoryName      [4]  Name,
    ediPartyName       [5]  EDIPartyName,
    uniformResourceIdentifier [6] IA5String,
    iPAddress          [7]  OCTET STRING,
    registeredID       [8]  OBJECT IDENTIFIER,
    ... }

OTHER-NAME ::= TYPE-IDENTIFIER

EDIPartyName ::= SEQUENCE {
    nameAssigner [0]  UnboundedDirectoryString OPTIONAL,
    partyName    [1]  UnboundedDirectoryString,
    ... }
```

The values in the alternatives of the **GeneralName** type are names of various forms as follows:

- the **otherName** alternative is a name of any form defined as an instance of the **OTHER-NAME** information object class;
- the **rfc822Name** alternative is an Internet electronic mail address defined in accordance with IETF RFC 822;
- AC1 – the **dNSName** alternative shall be a fully-qualified domain name (FQDN). The domain name shall be in the syntax as specified by section 2.3.1 of IETF RFC 5890 meaning that a domain name is a sequence of labels in the letters, digits, hyphen (LDH) format separated by dots. AC1

[AC₁] A label may be in one of two formats:

- a) All characters in the label are from the Basic Latin collection as defined by ISO/IEC 10646 (i.e., having code points in the ranges 002D, 0030-0039, 0041-005A and 0061-007A) and it does not start with "xn--". The maximum length is 63 octets.
- b) It is an A-label as defined in IETF RFC 5890, i.e., it starts with the "xn--" and is a U-label converted to valid ASCII characters as in item a) using the Punycode algorithm defined by IETF RFC 3492. The converted string shall be maximum 59 octets. To be valid, it shall be possible for an A-label to be converted to a valid U-label. The U-label is as also defined in IETF RFC 5890.

NOTE 1 – An A-label is normally not human-readable. **[AC₁]**

- the **x400Address** alternative is an O/R address defined in accordance with Rec. ITU-T X.411 | ISO/IEC 10021-4;
- the **directoryName** alternative is a distinguished name defined in accordance with Rec. ITU-T X.501 | ISO/IEC 9594-2;
- the **ediPartyName** alternative is a name of a form agreed between communicating Electronic Data Interchange partners; the **nameAssigner** component identifies an authority that assigns unique values of names in the **partyName** component;
- the **uniformResourceIdentifier** alternative is a Uniform Resource Identifier for the worldwide web defined in accordance with IETF RFC 1630;
- [AC₁]** – the **ipAddress** alternative is an Internet Protocol address defined in accordance with IETF RFC 791 for IPv4 (four octets) or in accordance with IETF 2460 for IPv6 (16 octets). **[AC₁]**
- the **registeredID** alternative is an object identifier of any registered object assigned in accordance with Rec. ITU-T X.660 | ISO/IEC 9834-1.

For every name form used in the **GeneralName** type, there shall be a name registration system that ensures that any name used unambiguously identifies one entity to both the issuing CA and relying parties.

This extension may, at the option of the certificate issuer, be either critical or non-critical. An implementation which supports this extension is not required to be able to process all name forms. If the extension is flagged critical, at least one of the name forms that is present shall be recognized and processed, otherwise the certificate shall be considered invalid. Apart from the preceding restriction, a relying party is permitted to ignore any name with an unrecognized or unsupported name form. It is recommended that, provided the subject field of the public-key certificate contains a distinguished name that unambiguously identifies the subject, this field be flagged non-critical.

NOTE 1 – Use of the TYPE-IDENTIFIER class is described in annexes A and C of Rec. ITU-T X.681 | ISO/IEC 8824-2.

NOTE 2 – If this extension field is present and is flagged critical, the **subject** field of an end-entity public-key certificate may contain a null name (e.g., a sequence of zero relative distinguished names) in which case the subject is identified only by the name or names in this extension (see clause 7.2).

8.3.2.2 Issuer alternative name extension

This field contains one or more alternative names, using any of a variety of name forms, for the certificate or CRL issuer. This field is defined as follows:

```
issuerAltName EXTENSION ::= {  
  SYNTAX          GeneralNames  
  IDENTIFIED BY   id-ce-issuerAltName }  
}
```

This extension may, at the option of the certificate or CRL issuer, be either critical or non-critical. An implementation which supports this extension is not required to be able to process all name forms. If the extension is flagged critical, at least one of the name forms that are present shall be recognized and processed, otherwise the certificate or CRL shall be considered invalid. Apart from the preceding restriction, a relying party is permitted to ignore any name with an unrecognized or unsupported name form. It is recommended that, provided the issuer field of the certificate or CRL contains a distinguished name that unambiguously identifies the issuing authority, this field be flagged non-critical.

8.3.2.3 Subject directory attributes extension

This field conveys any desired Directory attributes associated with the subject of the certificate. This field is defined as follows:

```
subjectDirectoryAttributes EXTENSION ::= {  
  SYNTAX          AttributesSyntax  
  IDENTIFIED BY   id-ce-subjectDirectoryAttributes }  
}
```

AttributesSyntax ::= SEQUENCE SIZE (1..MAX) OF Attribute{{SupportedAttributes}}

This extension may, at the option of the certificate issuer, be either critical or non-critical. A certificate-using system processing this extension is not required to understand all attribute types included in the extension. If the extension is flagged critical, at least one of the attribute types contained in the extension shall be understood for the certificate to be accepted. If the extension is flagged critical and none of the contained attribute types is understood, the certificate shall be rejected.

If this extension is present in a public-key certificate, some of the extensions defined in clause 15 may also be present.

8.4 Certification path constraint extensions

8.4.1 Requirements

For certification path processing:

- a) End-entity public-key certificates need to be distinguishable from CA-certificates to protect against end-entities establishing themselves as CAs without authorization. It also needs to be possible for a CA to limit the length of a subsequent chain resulting from a certified subject CA, e.g., to no more than one more certificate or no more than two more certificates.
- b) A CA needs to be able to specify constraints which allow a relying party to check that less-trusted CAs in a certification path (i.e., CAs further down the certification path from the CA with whose public key the relying party starts) are not violating their trust by issuing certificates to subjects in an inappropriate name space. Adherence to these constraints needs to be automatically checkable by the relying party.
- c) Certification path processing needs to be implementable in an automated, self-contained module. This is necessary to permit trusted hardware or software modules to be implemented which perform the certification path processing functions.
- d) It should be possible to implement certification path processing without depending upon real-time interactions with the local user.
- e) It should be possible to implement certification path processing without depending upon the use of trusted local databases of policy-description information. (Some trusted local information – an initial public key, at least – is needed for certification path processing but the amount of such information should be minimized.)
- f) Certification paths need to operate in environments in which multiple certificate policies are recognized. A CA needs to be able to stipulate which CAs in other domains it trusts and for which purposes. Chaining through multiple policy domains needs to be supported.
- g) Complete flexibility in trust models is required. A strict hierarchical model which is adequate for a single organization is not adequate when considering the needs of multiple interconnected enterprises. Flexibility is required in the selection of the first trusted CA in a certification path. In particular, it should be possible to require that the certification path start in the local security domain of the public-key user system.
- h) Naming structures should not be constrained by the need to use names in certificates, i.e., distinguished name structures considered natural for organizations or geographical areas shall not need adjustment in order to accommodate CA requirements.
- i) Certificate extension fields need to be backward-compatible with the unconstrained certification path approach system as specified in earlier editions of this Directory Specification.
- j) A CA needs to be able to inhibit the use of policy mapping and to require explicit certificate policy identifiers to be present in subsequent certificates in a certification path.

NOTE – In any relying party, the processing of a certification path requires an appropriate level of assurance. This Directory Specification defines functions that may be used in implementations that are required to conform to specific assurance statements. For example, an assurance requirement could state that certification path processing shall be protected from subversion of the process (such as software-tampering or data modification). The level of assurance should be commensurate with business risk. For example:

- processing internal to an appropriate cryptographic module may be required for public keys used to validate high value funds transfer; whereas
- processing in software may be appropriate for home banking balance inquiries.

Consequently, certification path processing functions should be suitable for implementation in hardware cryptographic modules or cryptographic tokens as one option.

- k) A CA needs to be able to prevent the special value any-policy from being considered a valid policy in subsequent certificates in a certification path.

8.4.2 Certificate extension fields

The following extension fields are defined:

- a) **basicConstraints**;
- b) **nameConstraints**;
- c) **policyConstraints**; and
- d) **inhibitAnyPolicy**.

These extension fields shall be used only as certificate extensions. Name constraints and policy constraints shall be used only in CA-certificates; basic constraints may also be used in end-entity public-key certificates. Examples of the use of these extensions are given in Annex G.

8.4.2.1 Basic constraints extension

This field indicates if the subject may act as a CA, with the certified public key being used to verify certificate signatures. If so, a certification path length constraint may also be specified. This field is defined as follows:

```
basicConstraints EXTENSION ::= {  
  SYNTAX          BasicConstraintsSyntax  
  IDENTIFIED BY   id-ce-basicConstraints }  
  
BasicConstraintsSyntax ::= SEQUENCE {  
  cA                BOOLEAN DEFAULT FALSE,  
  pathLenConstraint INTEGER(0..MAX) OPTIONAL,  
  ... }
```

The **cA** component with the value **TRUE** indicates that the certified public key may be used to verify public-key certificate signatures.

The **pathLenConstraint** component may be present if **cA** is set to **TRUE**. Otherwise, it shall be absent. It gives the maximum number of CA-certificates that may follow this CA-certificate in a certification path. Value 0 indicates that the subject of this CA-certificate may issue public-key certificates only to end-entities and not to further CAs. If no **pathLenConstraint** field appears in any CA-certificate of a certification path, there is no limit to the allowed length of the certification path. The constraint takes effect beginning with the next CA-certificate in the path. The constraint restricts the length of the segment of the certification path between the CA-certificate containing this extension and the end-entity public-key certificate. It has no impact on the number of CA-certificates in the certification path between the trust anchor and the CA-certificate containing this extension. Therefore, the length of a complete certification path may exceed the maximum length of the segment constrained by this extension. The constraint controls the number of non self-issued CA-certificates between the CA-certificate containing the constraint and the end-entity certificate. Therefore, the total length of this segment of the path, excluding self-issued certificates, may exceed the value of the constraint by as many as two certificates. (This includes the certificates at the two endpoints of the segment plus the CA-certificates between the two endpoints that are constrained by the value of this extension.)

This extension may, at the option of the issuing CA, be either critical or non-critical. It is recommended that it be flagged critical, otherwise, an entity which is not authorized to be a CA may issue certificates and a relying party may unwittingly use such a certificate.

If this extension is present and is flagged critical, or is flagged non-critical but is recognized by the relying party, then:

- if the value of **cA** is not set to **TRUE** then the certified public key shall not be used to verify a certificate signature;
- if the value of **cA** is set to **TRUE** and **pathLenConstraint** is present then the relying party shall check that the certification path being processed is consistent with the value of **pathLenConstraint**.

If this extension is not present, or is flagged non-critical and is not recognized by a relying party, then the public-key certificate is to be considered an end-entity public-key certificate and cannot be used to verify certificate signatures.

NOTE – To constrain a public-key certificate subject to being only an end-entity, i.e., not a CA, the issuer may include this extension field containing only an empty SEQUENCE value.

8.4.2.2 Name constraints extension

This field, which shall be used only in a CA-certificate, indicates one or more name forms which have constraints placed upon their name spaces, and in which all subject names in the same name form in subsequent certificates in a certification path must be located. If this extension is absent, then no constraints are placed on any name form. If this extension is present but a name form is not included in the extension, then no constraints are imposed on that name form.

NOTE – Because there can be an unbounded set of registeredID name forms, then in general it is not possible to constrain every possible name form of subject names with this extension.

This field is defined as follows:

```
nameConstraints EXTENSION ::= {
    SYNTAX          NameConstraintsSyntax
    IDENTIFIED BY   id-ce-nameConstraints }

NameConstraintsSyntax ::= SEQUENCE {
    permittedSubtrees  [0]  GeneralSubtrees OPTIONAL,
    excludedSubtrees  [1]  GeneralSubtrees OPTIONAL,
    ... }
(WITH COMPONENTS { ..., permittedSubtrees  PRESENT } |
 WITH COMPONENTS { ..., excludedSubtrees  PRESENT } )

GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

GeneralSubtree ::= SEQUENCE {
    base          GeneralName,
    minimum  [0]  BaseDistance DEFAULT 0,
    maximum  [1]  BaseDistance OPTIONAL,
    ... }

BaseDistance ::= INTEGER(0..MAX)
```

At least one of **permittedSubtrees** and **excludedSubtrees** components shall be present.

If present, the **permittedSubtrees** component specifies one or more subtrees, for one or more name forms, within which subject names in acceptable certificates shall be contained. If present, the **excludedSubtrees** component specifies one or more subtrees for one or more name forms within which subject names in acceptable certificates shall not be contained. Subject names that are compared against specified subtrees include those present in both the **subject** field and the **subjectAltNames** extension of a certificate. Each subtree is defined by the name of the root of the subtree, the **base** component, and, optionally, within that subtree, an area that is bounded by upper and/or lower levels.

The **minimum** field specifies the upper bound of the area within the subtree. All names whose final name component is above the level specified are not contained within the area. A value of **minimum** equal to zero (the default) corresponds to the base, i.e., the top node of the subtree. For example, if **minimum** is set to one, then the subtree excludes the base node but includes subordinate nodes.

The **maximum** field specifies the lower bound of the area within the subtree. All names whose last component is below the level specified are not contained within the area. A value of **maximum** of zero corresponds to the base, i.e., the top of the subtree. An absent **maximum** component indicates that no lower limit should be imposed on the area within the subtree. For example, if **maximum** is set to one, then the subtree excludes all nodes except the subtree base and its immediate subordinates.

The set of all **permittedSubtrees** and **excludedSubtrees** for a name form together comprise the constrained name space for the name form. All subject names, in certificates issued by the subject CA and subsequent CAs in a certification path, which are of a constrained name form, shall be located in the constrained name space for the certificate to be acceptable.

permittedSubtrees, if present, specifies the subtrees within which all the subject names that are of a constrained name form shall lie, for the certificate to be acceptable. If **excludedSubtrees** is present, any certificate issued by the subject CA or subsequent CAs in the certification path that has a subject name within these subtrees is unacceptable. If both **permittedSubtrees** and **excludedSubtrees** are present for a name form and the name spaces overlap, the exclusion statement takes precedence.

If none of the name forms of the subject name in the certificate is constrained by this extension, the certificate is acceptable.

In some situations, more than one certificate may need to be issued to satisfy the name constraints requirements. Annex G illustrates two of these situations. For example, if name constraints are defined for multiple name forms, but a certificate needs to meet the name constraints for only one of the name forms (logical OR on constraints), then multiple certificates should be issued, each constraining a single name form.

Of the name forms available through the **GeneralName** type, only those name forms that have a well-defined hierarchical structure may be used in these fields.

The **directoryName** name form satisfies this requirement; when using this name form a naming subtree corresponds to a DIT subtree. A **certificate** is considered subordinate to the **base** (and therefore a candidate to be within the subtree) if the **SEQUENCE** of **RDNs**, which forms the full DN in **base**, is identical to the initial **SEQUENCE** of the same number of **RDNs** which forms the first part of the DN of the subject (in the **subject** field or **directoryName** of **subjectAltNames** extension) of the **certificate**. The DN of the subject of the **certificate** may have additional trailing **RDNs** in its sequence that do not appear in the DN in **base**. The **distinguishedNameMatch** matching rule is used to compare the value of **base** with the initial sequence of **RDNs** in the DN of the subject of the certificate.

Conformant implementations are not required to recognize all possible name forms. If the extension is flagged as being critical and a certificate-using implementation does not recognize a name form used in any **base** component, the certificate shall be handled as if an unrecognized critical extension had been encountered. If the extension is flagged as being non-critical and a certificate-using implementation does not recognize a name form used in any **base** component, then that subtree may be ignored.

When testing certificate subject names for consistency with a name constraint, names in non-critical subject alternative name extensions shall be processed, not ignored.

This extension may, at the option of the certificate issuer, be either critical or non-critical. It is recommended that it be flagged as critical; otherwise, a relying party may not check that subsequent certificates in a certification path are located in the constrained name spaces intended by the issuing CA.

If this extension is present and is flagged as being critical, then a relying party shall check that the certification path being processed is consistent with the value in this extension.

Annex G contains examples of use of the name constraints extension.

8.4.2.3 Policy constraints extension

This field specifies constraints which may require explicit certificate policy identification or inhibit policy mapping for the remainder of the certification path. This field is defined as follows:

```
policyConstraints EXTENSION ::= {  
    SYNTAX          PolicyConstraintsSyntax  
    IDENTIFIED BY   id-ce-policyConstraints }  
  
PolicyConstraintsSyntax ::= SEQUENCE {  
    requireExplicitPolicy [0] SkipCerts OPTIONAL,  
    inhibitPolicyMapping  [1] SkipCerts OPTIONAL,  
    ... }  
(WITH COMPONENTS {..., requireExplicitPolicy PRESENT } |  
 WITH COMPONENTS {..., inhibitPolicyMapping PRESENT } )  
  
SkipCerts ::= INTEGER(0..MAX)
```

At least one of the **requireExplicitPolicy** and **inhibitPolicyMapping** components shall be present.

If the **requireExplicitPolicy** component is present, and the certification path includes a certificate issued by a nominated CA, it is necessary for all certificates in the path to contain, in the certificate policies extension, an acceptable policy identifier. An acceptable policy identifier is the identifier of a certificate policy required by the user of the certification path, the identifier of a policy which has been declared equivalent to one of these policies through policy mapping, or *any-policy*. The nominated CA is either the issuer CA of the certificate containing this extension (if the value of **requireExplicitPolicy** is 0) or a CA which is the issuer of a subsequent certificate in the certification path (as indicated by a non-zero value).

If the **inhibitPolicyMapping** component is present, it indicates that, in all certificates starting from a nominated CA in the certification path until the end of the certification path, policy mapping is not permitted. The nominated CA is either the subject CA of the certificate containing this extension (if the value of **inhibitPolicyMapping** is 0) or a CA which is the subject of a subsequent certificate in the certification path (as indicated by a non-zero value).

A value of type **SkipCerts** indicates the number of certificates in the certification path to skip before a constraint becomes effective.

This extension may, at the option of the certificate issuer, be either critical or non-critical. It is recommended that it be flagged critical; otherwise, a relying party may not correctly interpret the stipulation of the issuing CA.

8.4.2.4 Inhibit any policy extension

This field specifies a constraint that indicates *any-policy* is not considered an explicit match for other certificate policies for all non-self-issued certificates in the certification path starting with a nominated CA. The nominated CA is either the

subject CA of the certificate containing this extension (if the value of `inhibitAnyPolicy` is 0) or a CA which is the subject of a subsequent certificate in the certification path (as indicated by a non-zero value).

```
inhibitAnyPolicy EXTENSION ::= {
  SYNTAX          SkipCerts
  IDENTIFIED BY   id-ce-inhibitAnyPolicy }
```

This extension may, at the option of the certificate issuer, be either critical or non-critical. It is recommended that it be critical, otherwise a relying party may not correctly interpret the stipulation of the issuing CA.

8.5 Basic CRL extensions

8.5.1 Requirements

The following requirements relate to CRLs:

- a) Relying parties need to be able to track all CRLs issued from a CRL issuer or CRL distribution point (see clause 8.6) and be able to detect a missing CRL in the sequence. CRL sequence numbers are therefore required.
- b) Some CRL users may wish to respond differently to a revocation, depending upon the reason for the revocation. There is therefore a requirement for a CRL entry to indicate the reason for revocation.
- c) There is a requirement for an authority to be able to temporarily suspend validity of a certificate and subsequently either revoke or reinstate it. Possible reasons for such an action include:
 - desire to reduce liability for erroneous revocation when a revocation request is unauthenticated and there is inadequate information to determine whether it is valid;
 - other business needs, such as temporarily disabling the certificate of an entity pending an audit or investigation.
- d) A CRL contains, for each revoked certificate, the date when the authority posted the revocation. Further information may be known as to when an actual or suspected key compromise occurred, and this information may be valuable to a relying party. The revocation date is insufficient to solve some disputes because, assuming the worst, all signatures issued during the validity period of the certificate have to be considered invalid. However, it may be important for a user that a signed document be recognized as valid even though the key used to sign the message was compromised after the signature was produced. To assist in solving this problem, a CRL entry can include a second date which indicates when it was known or suspected that the private key was compromised.
- b) Relying parties need to be able to determine, from the CRL itself, additional information including the scope of certificates covered by this list, the ordering of revocation notices, and which stream of CRLs the CRL number is unique within.
- e) Issuers need the ability dynamically to change the partitioning of CRLs and to refer relying parties to the new location for relevant CRLs if the partitioning changes.
- f) Delta CRLs may also be available that update a given base CRL. Relying parties need to be able to determine, from a given CRL, whether delta CRLs are available, where they are located and when the next delta CRL will be issued.
- g) In addition to CRLs publishing a notification that certificates have been revoked, there is a requirement to publish a notification that certificates will be revoked as of a specified date and time in the future.
- h) There is a requirement to provide more efficient ways to indicate in a CRL that a set of certificates has been revoked.

8.5.2 CRL extension fields

The following extension fields are defined:

- a) CRL number;
- b) CRL scope;
- c) Status referral;
- d) CRL stream identifier;
- e) Ordered list;
- f) Delta information;
- g) To be revoked;
- h) Revoked Group of certificates; and
- i) Expired certificates on CRL.

8.5.2.1 CRL number extension

This CRL extension field conveys a monotonically increasing sequence number for each CRL issued by a given CRL issuer through a given authority directory attribute or CRL distribution point. It allows a CRL user to detect whether CRLs issued prior to the one being processed were also seen and processed. This field is defined as follows:

```
cRLNumber EXTENSION ::= {  
    SYNTAX          CRLNumber  
    IDENTIFIED BY   id-ce-cRLNumber }
```

```
CRLNumber ::= INTEGER(0..MAX)
```

This extension is always non-critical.

8.5.2.2 CRL scope extension

NOTE – Use of the CRL scope extension is deprecated.

The scope of a CRL is indicated within that CRL using the following CRL extension. In order to prevent a CRL substitution attack against an application that does not support the scope extension, the scope extension, if present, shall be marked critical.

This extension may be used to provide scope statements of various CRL types including:

- simple CRLs that provide revocation information about certificates issued by a single authority;
- indirect CRLs that provide revocation information about certificates issued by multiple authorities;
- delta-CRLs that update previously issued revocation information;
- indirect delta-CRLs that provide revocation information that updates multiple base CRLs issued by a single authority or by multiple authorities.

```
crlScope EXTENSION ::= {  
    SYNTAX          CRLScopeSyntax  
    IDENTIFIED BY   id-ce-cRLScope }
```

```
CRLScopeSyntax ::= SEQUENCE SIZE (1..MAX) OF PerAuthorityScope
```

```
PerAuthorityScope ::= SEQUENCE {  
    authorityName      [0] GeneralName OPTIONAL,  
    distributionPoint  [1] DistributionPointName OPTIONAL,  
    onlyContains       [2] OnlyCertificateTypes OPTIONAL,  
    onlySomeReasons    [4] ReasonFlags OPTIONAL,  
    serialNumberRange  [5] NumberRange OPTIONAL,  
    subjectKeyIdRange  [6] NumberRange OPTIONAL,  
    nameSubtrees       [7] GeneralNames OPTIONAL,  
    baseRevocationInfo [9] BaseRevocationInfo OPTIONAL,  
    ... }
```

```
OnlyCertificateTypes ::= BIT STRING {  
    user      (0),  
    authority (1),  
    attribute (2)}
```

```
NumberRange ::= SEQUENCE {  
    startingNumber [0] INTEGER OPTIONAL,  
    endingNumber   [1] INTEGER OPTIONAL,  
    modulus        INTEGER OPTIONAL,  
    ... }
```

```
BaseRevocationInfo ::= SEQUENCE {  
    cRLStreamIdentifier [0] CRLStreamIdentifier OPTIONAL,  
    cRLNumber           [1] CRLNumber,  
    baseThisUpdate      [2] GeneralizedTime,  
    ... }
```

If the CRL is an indirect CRL that provides revocation status information for multiple authorities, the extension will include multiple **PerAuthorityScope** constructs, one or more for each of the authorities for which revocation information is included. Each instance of **PerAuthorityScope** that relates to an authority other than that issuing this CRL shall contain the **authorityName** component. If the CRL is a dCRL that provides delta revocation status information for multiple base CRLs issued by a single authority, the extension will include multiple **PerAuthorityScope** constructs, one for each of the base CRLs for which this dCRL provides updates. Even though