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

*Technologies de l'information — Interconnexion de systèmes ouverts
(OSI) — L'annuaire: Cadres de clé publique et de certificat d'attribut*

Reference number
ISO/IEC 9594-8:2001(E)



PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2001

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either 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.ch
Web www.iso.ch

Printed in Switzerland

CONTENTS

	<i>Page</i>
Introduction.....	viii
SECTION 1 – GENERAL	1
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
3 Definitions	3
3.1 OSI Reference Model security architecture definitions.....	3
3.2 Directory model definitions.....	3
3.3 Definitions.....	4
4 Abbreviations.....	6
5 Conventions.....	7
6 Frameworks overview	7
6.1 Digital signatures.....	8
SECTION 2 – PUBLIC-KEY CERTIFICATE FRAMEWORK.....	10
7 Public-keys and public-key certificates	11
7.1 Generation of key pairs	15
7.2 Public-key certificate creation	15
7.3 Certificate validity.....	15
8 Public-key certificate and CRL extensions.....	17
8.1 Policy handling.....	18
8.1.1 Certificate policy	18
8.1.2 Cross-certification	19
8.1.3 Policy mapping.....	20
8.1.4 Certification path processing.....	20
8.1.5 Self-issued certificates.....	20
8.2 Key and policy information extensions	21
8.2.1 Requirements.....	21
8.2.2 Public-key certificate and CRL extension fields	21
8.2.2.1 Authority key identifier extension.....	22
8.2.2.2 Subject key identifier extension	22
8.2.2.3 Key usage extension.....	22
8.2.2.4 Extended key usage extension.....	23
8.2.2.5 Private key usage period extension	24
8.2.2.6 Certificate policies extension.....	24
8.2.2.7 Policy mappings extension	25
8.3 Subject and issuer information extensions	26
8.3.1 Requirements.....	26
8.3.2 Certificate and CRL extension fields.....	26
8.3.2.1 Subject alternative name extension	26
8.3.2.2 Issuer alternative name extension.....	26
8.3.2.3 Subject directory attributes extension.....	27
8.4 Certification path constraint extensions.....	27
8.4.1 Requirements.....	27
8.4.2 Certificate extension fields.....	28
8.4.2.1 Basic constraints extension.....	28
8.4.2.2 Name constraints extension	29
8.4.2.3 Policy constraints extension	30
8.4.2.4 Inhibit any policy extension	30

8.5	Basic CRL extensions.....	30
8.5.1	Requirements.....	30
8.5.2	CRL and CRL entry extension fields.....	31
8.5.2.1	CRL number extension.....	31
8.5.2.2	Reason code extension	32
8.5.2.3	Hold instruction code extension	32
8.5.2.4	Invalidity date extension.....	33
8.5.2.5	CRL scope extension.....	33
8.5.2.6	Status referral extension	35
8.5.2.7	CRL stream identifier extension.....	36
8.5.2.8	Ordered list extension.....	36
8.5.2.9	Delta information extension	37
8.6	CRL distribution points and delta-CRL extensions.....	37
8.6.1	Requirements.....	37
8.6.2	CRL distribution point and delta-CRL extension fields	38
8.6.2.1	CRL distribution points extension	38
8.6.2.2	Issuing distribution point extension.....	39
8.6.2.3	Certificate issuer extension.....	40
8.6.2.4	Delta CRL indicator extension	40
8.6.2.5	Base update extension	41
8.6.2.6	Freshest CRL extension.....	41
9	Delta CRL relationship to base	41
10	Certification path processing procedure	42
10.1	Path processing inputs	42
10.2	Path processing outputs	43
10.3	Path processing variables	43
10.4	Initialization step	43
10.5	Certificate processing	44
10.5.1	Basic certificate checks	44
10.5.2	Processing intermediate certificates	44
10.5.3	Explicit policy indicator processing	45
10.5.4	Final processing.....	46
11	PKI directory schema	46
11.1	PKI directory object classes and name forms	46
11.1.1	PKI user object class	46
11.1.2	PKI CA object class.....	46
11.1.3	CRL distribution points object class and name form.....	46
11.1.4	Delta CRL object class	47
11.1.5	Certificate Policy & CPS object class	47
11.1.6	PKI certificate path object class	47
11.2	PKI directory attributes	47
11.2.1	User certificate attribute	47
11.2.2	CA certificate attribute	47
11.2.3	Cross certificate pair attribute.....	48
11.2.4	Certificate revocation list attribute	48
11.2.5	Authority revocation list attribute.....	48
11.2.6	Delta revocation list attribute	48
11.2.7	Supported algorithms attribute	48
11.2.8	Certification practice statement attribute.....	49
11.2.9	Certificate policy attribute	49
11.2.10	PKI path attribute	49
11.3	PKI directory matching rules.....	50
11.3.1	Certificate exact match	50
11.3.2	Certificate match	50
11.3.3	Certificate pair exact match.....	51
11.3.4	Certificate pair match	51
11.3.5	Certificate list exact match	52

11.3.6	Certificate list match.....	52
11.3.7	Algorithm identifier match.....	53
11.3.8	Policy match.....	53
11.3.9	PKI path match.....	53
SECTION 3 – ATTRIBUTE CERTIFICATE FRAMEWORK		53
12	Attribute certificates	54
12.1	Attribute certificate structure.....	54
12.2	Attribute certificate paths	56
13	Attribute Authority, SOA and Certification Authority relationship	56
13.1	Privilege in attribute certificates.....	57
13.2	Privilege in public-key certificates.....	57
14	PMI models.....	58
14.1	General model	58
14.1.1	PMI in access control context.....	59
14.1.2	PMI in a non-repudiation context.....	59
14.2	Control model.....	59
14.3	Delegation model	60
14.4	Roles model.....	61
14.4.1	Role attribute	61
15	Privilege management certificate extensions.....	62
15.1	Basic privilege management extensions.....	62
15.1.1	Requirements.....	62
15.1.2	Basic privilege management extension fields.....	63
15.1.2.1	Time specification extension.....	63
15.1.2.2	Targeting information extension	63
15.1.2.3	User notice extension	64
15.1.2.4	Acceptable privilege policies extension	64
15.2	Privilege revocation extensions	65
15.2.1	Requirements.....	65
15.2.2	Privilege revocation extension fields.....	65
15.2.2.1	CRL distribution points extension.....	65
15.2.2.2	No revocation information extension	66
15.3	Source of Authority extensions	65
15.3.1	Requirements.....	65
15.3.2	SOA extension fields.....	65
15.3.2.1	SOA identifier extension	65
15.3.2.2	Attribute descriptor extension	66
15.4	Role extensions.....	67
15.4.1	Requirements.....	67
15.4.2	Role extension fields	67
15.4.2.1	Role specification certificate identifier extension	67
15.5	Delegation extensions.....	68
15.5.1	Requirements.....	68
15.5.2	Delegation extension fields	68
15.5.2.1	Basic attribute constraints extension	68
15.5.2.2	Delegated name constraints extension.....	70
15.5.2.3	Acceptable certificate policies extension.....	70
15.5.2.4	Authority attribute identifier extension	71
16	Privilege path processing procedure	71
16.1	Basic processing procedure	72
16.2	Role processing procedure	72
16.3	Delegation processing procedure.....	72
16.3.1	Verify integrity of domination rule.....	73
16.3.2	Establish valid delegation path.....	73
16.3.3	Verify privilege delegation.....	73
16.3.4	Pass/fail determination	74

17	PMI directory schema	74
17.1	PMI directory object classes	74
17.1.1	PMI user object class	74
17.1.2	PMI AA object class	74
17.1.3	PMI SOA object class	74
17.1.4	Attribute certificate CRL distribution point object class	74
17.1.5	PMI delegation path object class	75
17.1.6	Privilege policy object class	75
17.2	PMI Directory attributes	75
17.2.1	Attribute certificate attribute	75
17.2.2	AA certificate attribute	75
17.2.3	Attribute descriptor certificate attribute	75
17.2.4	Attribute certificate revocation list attribute	75
17.2.5	AA certificate revocation list attribute	76
17.2.6	Delegation path attribute	76
17.2.7	Privilege policy attribute	76
17.3	PMI general directory matching rules	76
17.3.1	Attribute certificate exact match	76
17.3.2	Attribute certificate match	76
17.3.3	Holder issuer match	77
17.3.4	Delegation path match	77
	SECTION 4 – DIRECTORY USE OF PUBLIC-KEY & ATTRIBUTE CERTIFICATE FRAMEWORKS	77
18	Directory authentication	78
18.1	Simple authentication procedure	78
18.1.1	Generation of protected identifying information	79
18.1.2	Procedure for protected simple authentication	79
18.1.3	User password attribute type	80
18.2	Strong Authentication	80
18.2.1	Obtaining public-key certificates from the directory	81
18.2.1.1	Example	81
18.2.2	Strong authentication procedures	83
18.2.2.1	One-way authentication	84
18.2.2.2	Two-way authentication	84
18.2.2.3	Three-way authentication	85
19	Access control	86
20	Protection of Directory operations	86
	Annex A – Public-Key and Attribute Certificate Frameworks	87
	Annex B – CRL Generation and Processing Rules	107
B.1	Introduction	107
B.1.1	CRL types	107
B.1.2	CRL processing	108
B.2	Determine parameters for CRLs	108
B.3	Determine CRLs required	109
B.3.1	End-entity with critical CRL DP	109
B.3.2	End-entity with no critical CRL DP	109
B.3.3	CA with critical CRL DP	110
B.3.4	CA with no critical CRL DP	110
B.4	Obtain CRLs	110
B.5	Process CRLs	110
B.5.1	Validate base CRL scope	110
B.5.1.1	Complete CRL	111
B.5.1.2	Complete EPRL	111
B.5.1.3	Complete CARL	111
B.5.1.4	Distribution point based CRL/EPRL/CARL	112
B.5.2	Validate delta CRL scope	112
B.5.3	Validity and currency checks on the base CRL	113
B.5.4	Validity and checks on the delta CRL	113

Annex C – Examples of Delta CRL Issuance	114
C.1 Introduction	114
Annex D – Privilege Policy and Privilege Attribute Definition Examples	116
D.1 Introduction	116
D.2 Sample syntaxes	116
D.2.1 First example	116
D.2.2 Second example	118
D.3 Privilege attribute example	120
Annex E – An introduction to public key cryptography	121
Annex F – Reference definition of algorithm object identifiers	123
Annex G – Examples of use of certification path constraints	124
G.1 Example 1: Use of basic constraints	124
G.2 Example 2: Use of name constraints	124
G.3 Example 3: Use of policy mapping and policy constraints	124
Annex H – Alphabetical list of information item definitions	126
Annex I – Amendments and corrigenda	129

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. 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.

Users and implementors should note the existence of a "defect resolution" procedure in ISO/IEC JTC 1 to identify and correct errors in International Standards through the publication of Technical Corrigenda. Identical corrections are made to the corresponding ITU-T Recommendations through Corrigenda and may also be made in the form of Implementors' Guides. Details of Technical Corrigenda to International Standards are available on the ISO website; published Technical Corrigenda can be obtained via the ISO webstore or from the ISO and IEC national bodies. Corrigenda and Implementors' Guides to ITU-T Recommendations can be obtained from the ITU-T website.

International Standard 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 ITU-T Recommendation X.509.

This fourth edition of ISO/IEC 9594-8 constitutes a technical revision of the third edition (ISO/IEC 9594-8:1998), which is provisionally retained in order to support implementations based on the third edition. This edition also incorporates Corrigendum 1:2000.

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*
- *Part 10: Use of systems management for administration of the Directory*

Annexes A, B and F form a normative part of this part of ISO/IEC 9594. Annexes C, D, E, G, H and I are for information only.

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. That framework includes 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 Specification, while it defines some critical components of a Public-key Infrastructure (PKI), it does not define a PKI in its entirety. However, this Specification 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 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 Specification, while it defines some critical components of a Privilege Management Infrastructure (PMI), it does not define a PMI in its entirety. However, this Specification 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 fourth edition technically revises and enhances, but does not replace, the third edition of this Recommendation | International Standard. Implementations may still claim conformance to the third edition. However, at some point, the third edition will not be supported (i.e. reported defects will no longer be resolved). It is recommended that implementations conform to this fourth edition as soon as possible.

This fourth edition specifies version 1 and 2 of the Directory protocols.

The first and second editions specified only version 1. Most of the services and protocols specified in this edition are designed to function under version 1. However some enhanced services and protocols, e.g. signed errors, will not function unless all Directory entities involved in the operation have negotiated version 2. Whichever version has been negotiated, differences between the services and between the protocols defined in the four editions, except for those specifically assigned to version 2, are accommodated using the rules of extensibility defined in this edition of ITU-T Rec. X.519 | ISO/IEC 9594-5.

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

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

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

ISO/IEC 9594-8:2001(E)

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

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

Annex F, 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 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, contains an alphabetical list of information item definitions in this Specification.

Annex I, 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.

The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this part of ISO/IEC 9594 may involve the use of a patent concerning the CRL distribution point facility.

ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO and IEC that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from:

Entrust Technologies
Mr. Mike Morgan, Senior Legal Counsel
750 Heron Road, Suite E08
Ottawa, Ontario K1V 1A7
Canada

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 9594 may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

**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;
- Authentication services.

The public-key certificate framework defined in this Recommendation | International Standard includes definition of the information objects for Public Key Infrastructure (PKI), including public-key certificates, and Certificate Revocation List (CRL). The attribute certificate framework includes definition of the information objects for Privilege Management Infrastructure (PMI), including attribute certificates, and Attribute Certificate Revocation List (ACRL). This Specification 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, signed and/or encrypted operations, and for storage of signed and/or encrypted 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 Specification, 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 Directory Specifications.

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 ITU-T Rec. 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 ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation X.411 (1999) | ISO/IEC 10021-4:1999, *Information technology – Message Handling Systems (MHS) – Message transfer system: Abstract service definition and procedures.*
- ITU-T Recommendation X.500 (2001) | ISO/IEC 9594-1:2001, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services.*¹⁾
- ITU-T Recommendation X.501 (2001) | ISO/IEC 9594-2:2001, *Information technology – Open Systems Interconnection – The Directory: Models.*
- ITU-T Recommendation X.511 (2001) | ISO/IEC 9594-3:2001, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition.*
- ITU-T Recommendation X.518 (2001) | ISO/IEC 9594-4:2001, *Information technology – Open Systems Interconnection – The Directory: Procedures for distributed operation.*
- ITU-T Recommendation X.519 (2001) | ISO/IEC 9594-5:2001, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications.*
- ITU-T Recommendation X.520 (2001) | ISO/IEC 9594-6:2001, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types.*
- ITU-T Recommendation X.521 (2001) | ISO/IEC 9594-7:2001, *Information technology – Open Systems Interconnection – The Directory: Selected object classes.*
- ITU-T Recommendation X.525 (2001) | ISO/IEC 9594-9:2001, *Information technology – Open Systems Interconnection – The Directory: Replication.*
- ITU-T Recommendation X.530 (2001) | ISO/IEC 9594-10:2001, *Information technology – Open Systems Interconnection – The Directory: Use of systems management for administration of the Directory.*
- CCITT Recommendation X.660 (1992) | ISO/IEC 9834-1:1993, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures.*
- ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*
- ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*

¹⁾ For each of the X.500 series of Recommendations | 9594 parts referenced in this clause, the 4th edition of those Specifications should be used when they become available.

- ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:1998, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.
- ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*.
- ITU-T Recommendation X.691 (1997) | ISO/IEC 8825-2:1998, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.
- ITU-T Recommendation X.812 (1995) | ISO/IEC 10181-3:1996, *Information technology – Open Systems Interconnection – Security frameworks for open systems: Access control framework*.
- ITU-T Recommendation X.813 (1996) | ISO/IEC 10181-4:1997, *Information technology – Open Systems Interconnection – Security frameworks for open systems: Non-repudiation framework*.
- ITU-T Recommendation X.880 (1994) | ISO/IEC 13712-1:1995, *Information technology – Remote Operations: Concepts, model and notation*.
- ITU-T Recommendation X.881 (1994) | ISO/IEC 13712-2:1995, *Information technology – Remote Operations: OSI realizations – Remote Operations Service Element (ROSE) service definition*.

2.2 Paired Recommendations | International Standards equivalent in technical content

- CCITT Recommendation 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*.

3 Definitions

For the purposes of this ITU-T 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:

- a) asymmetric (encipherment);
- b) authentication exchange;
- c) authentication information;
- d) confidentiality;
- e) credentials;
- f) cryptography;
- g) data origin authentication;
- h) decipherment;
- i) encipherment;
- j) key;
- k) password;
- l) peer-entity authentication;
- m) symmetric (encipherment).

3.2 Directory model definitions

The following terms are defined in ITU-T Rec. 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.

3.3 Definitions

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

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

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

3.3.3 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.

3.3.4 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.

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

3.3.6 authority: An entity, responsible for the issuance of certificates. Two types are defined in this Specification; certification authority which issues public-key certificates and attribute authority which issues attribute certificates.

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

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

3.3.9 CA-certificate: A certificate for one CA issued by another CA.

3.3.10 certificate policy: A named set of rules that indicates 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 applicability of a type of certificate to the authentication of electronic data interchange transactions for the trading of goods within a given price range.

3.3.11 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.

3.3.12 certificate user: An entity that needs to know, with certainty, the public key of another entity.

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

3.3.14 certificate-using system: An implementation of those functions defined in this Directory Specification that are used by a certificate-user.

3.3.15 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.

3.3.16 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 users' keys.

3.3.17 Certification Authority Revocation List (CARL): A revocation list containing a list of public-key certificates issued to certification authorities, that are no longer considered valid by the certificate issuer.

- 3.3.18 certification path:** An ordered sequence of certificates of objects in the DIT which, together with the public key of the initial object in the path, can be processed to obtain that of the final object in the path.
- 3.3.19 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.
- 3.3.20 cryptographic system, cryptosystem:** A collection of transformations from plain text into ciphertext and vice versa, the particular transformation(s) to be used being selected by keys. The transformations are normally defined by a mathematical algorithm.
- 3.3.21 data confidentiality:** This service can be used to provide for 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.
- 3.3.22 delegation:** Conveyance of privilege from one entity that holds such privilege, to another entity.
- 3.3.23 delegation path:** An ordered sequence of certificates which, together with authentication of a privilege asserter's identity can be processed to verify the authenticity of a privilege asserter's privilege.
- 3.3.24 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.
- 3.3.25 end entity:** A certificate subject that uses its private key for purposes other than signing certificates or an entity that is a relying party.
- 3.3.26 End-entity Attribute Certificate Revocation List (EARL):** A revocation list containing a list of attribute certificates issued to holders, that are not also AAs, that are no longer considered valid by the certificate issuer.
- 3.3.27 End-entity Public-key Certificate Revocation List (EPRL):** A revocation list containing a list of public-key certificates issued to subjects, that are not also CAs, that are no longer considered valid by the certificate issuer.
- 3.3.28 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).
- 3.3.29 full CRL:** A complete revocation list that contains entries for all certificates that have been revoked for the given scope.
- 3.3.30 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.
- 3.3.31 holder:** An entity to whom some privilege has been delegated either directly from the Source of Authority or indirectly through another Attribute Authority.
- 3.3.32 indirect CRL (iCRL):** A revocation list that at least contains revocation information about certificates issued by authorities other than that which issued this CRL.
- 3.3.33 key agreement:** A method for negotiating a key value on-line 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).
- 3.3.34 object method:** An action that can be invoked on a resource (e.g. a file system may have read, write and execute object methods).
- 3.3.35 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 y for which finding x is not computationally difficult.
- 3.3.36 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.
- 3.3.37 private key; secret key (deprecated):** (In a public key cryptosystem) that key of a user's key pair which is known only by that user.

- 3.3.38 privilege:** An attribute or property assigned to an entity by an authority.
- 3.3.39 privilege asserter:** A privilege holder using their attribute certificate or public-key certificate to assert privilege.
- 3.3.40 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.
- 3.3.41 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.
- 3.3.42 privilege verifier:** An entity verifying certificates against a privilege policy.
- 3.3.43 public-key:** (In a public key cryptosystem) that key of a user's key pair which is publicly known.
- 3.3.44 public-key certificate:** The public key of a user, together with some other information, rendered unforgeable by encipherment with the private key of the certification authority which issued it.
- 3.3.45 Public Key Infrastructure (PKI):** The infrastructure able to support the management of public keys able to support authentication, encryption, integrity or non-repudiation services.
- 3.3.46 relying party:** A user or agent that relies on the data in a certificate in making decisions.
- 3.3.47 role assignment certificate:** A certificate that contains the role attribute, assigning one or more roles to the certificate subject/holder.
- 3.3.48 role specification certificate:** A certificate that contains the assignment of privileges to a role.
- 3.3.49 sensitivity:** Characteristic of a resource that implies its value or importance.
- 3.3.50 simple authentication:** Authentication by means of simple password arrangements.
- 3.3.51 security policy:** The set of rules laid down by the security authority governing the use and provision of security services and facilities.
- 3.3.52 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.
- 3.3.53 strong authentication:** Authentication by means of cryptographically derived credentials.
- 3.3.54 trust:** Generally, an entity can be said to "trust" a second entity when it (the first entity) makes the assumption that the second entity will behave exactly as the first entity expects. This trust may apply only for some specific function. The key role of trust in this framework is to describe the relationship between an authenticating entity and a authority; an entity shall be certain that it can trust the authority to create only valid and reliable certificates.

4 Abbreviations

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

AA	Attribute Authority
AARL	Attribute Authority Revocation List
ACRL	Attribute Certificate Revocation List
CA	Certification Authority
CARL	Certification Authority Revocation List
CRL	Certificate Revocation List
dCRL	Delta Certificate Revocation List
DIB	Directory Information Base
DIT	Directory Information Tree
DSA	Directory System Agent

DUA	Directory User Agent
EARL	End-entity Attribute certificate Revocation List
EPRL	End-entity Public-key certificate Revocation List
iCRL	Indirect Certificate Revocation List
PKCS	Public-Key Cryptosystem
PKI	Public-Key Infrastructure
PMI	Privilege Management Infrastructure
SOA	Source of Authority

5 Conventions

With minor exceptions this Directory Specification has been prepared according to the "Presentation of ITU-T/ISO/IEC common text" guidelines in the Guide for ITU-T and ISO/IEC JTC 1 Cooperation, March 1996.

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

This Directory Specification uses the term "1988 edition systems" to refer to systems conforming to the first edition of the 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 "1993 edition systems" to refer to systems conforming to the second edition of the 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 "1997 edition systems" to refer to systems conforming to the third edition of the 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 "4th edition systems" to refer to systems conforming to this fourth edition of the Directory Specifications", i.e. the 2001 editions of 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 ITU-T X.509, and parts 1-10 of the ISO/IEC 9594:2001 edition.

This Directory Specification presents ASN.1 notation in the bold Helvetica typeface. When ASN.1 types and values are referenced in normal text, they are differentiated from normal text by presenting them in the bold Helvetica typeface. The names of procedures, typically referenced when specifying the semantics of processing, are differentiated from normal text by displaying them in bold Times. Access control permissions are presented in italicized Times.

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

The notation used in this Directory Specification is defined in Table 1 below.

6 Frameworks overview

This 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 certificate by the certificate user. The validation includes:

- establishing a trusted path of certificates between the certificate user and the certificate subject;
- verifying the digital signatures on each certificate in the path; and
- validating all the certificates along that path (i.e. that they were not expired or not revoked at a given time).

This Specification defines a framework for obtaining and trusting privilege attributes of an entity in order to determine whether or not they are authorized to access a particular resource. The framework includes the issuance of a certificate by an Attribute Authority (AA) and the validation of that 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.

Table 1 – Notation

Notation	Meaning
X_p	Public key of a user X.
X_s	Private key of X.
$X_p[I]$	Encipherment of some information, I, using the public key of X.
$X_s[I]$	Encipherment of I using the private key of X.
$X\{I\}$	The signing of I by user X. It consists of I with an enciphered summary appended.
$CA(X)$	A certification authority of user X.
$CA^n(X)$	(Where $n > 1$): $CA(CA(\dots n \text{ times } \dots (X)))$
$X_1 \langle X_2 \rangle$	The certificate of user X_2 issued by certification authority X_1 .
$X_1 \langle X_2 \rangle X_2 \langle X_3 \rangle$	A chain of certificates (can be of arbitrary length), where each item is the certificate for the certification authority which produced the next. It is functionally equivalent to the following certificate $X_1 \langle X_{n+1} \rangle$. For example, possession of $A \langle B \rangle B \langle C \rangle$ provides the same capability as $A \langle C \rangle$, namely the ability to find out C_p given A_p .
$X_{1p} \circ X_1 \langle X_2 \rangle$	The operation of unwrapping a certificate (or certificate chain) to extract a public key. It is an infix operator, whose left operand is the public key of a certification authority, and whose right operand is a certificate issued by that certification authority. The outcome is the public key of the user whose certificate is the right operand. For example: $A_p \circ A \langle B \rangle B \langle C \rangle$ denotes the operation of using the public key of A to obtain B's public key, B_p , from its certificate, followed by using B_p to unwrap C's certificate. The outcome of the operation is the public key of C, C_p .
$A \rightarrow B$	A certification path from A to B, formed of a chain of certificates, starting with $CA(A) \langle CA^2(A) \rangle$ and ending with $CA(B) \langle B \rangle$.
NOTE – In the table, the symbols X, X_1 , X_2 , etc. occur in place of the names of users, while the symbol I occurs in place of arbitrary information.	

Although PKI and PMI are separate infrastructures and may be established independently from one another, they are related. This 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.

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 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 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 the tokens are signed in PKI, PMI and in the Directory.

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:

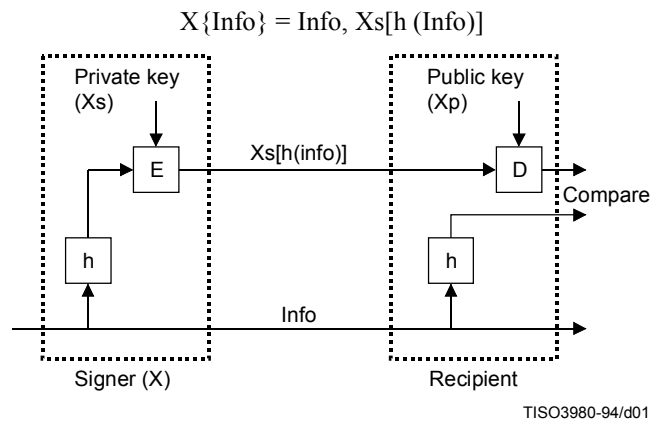


Figure 1 – Digital signatures

NOTE 1 – 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 so as 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 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 as a result 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.

The signed information includes indicators that identify the hashing algorithm and the encryption algorithm used to compute the digital signature.

The encipherment of some 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 – ITU-T Rec. X.690 (1997) | ISO/IEC 8825-1:1998) of the value of the **ToBeEnciphered** type and applying an encipherment procedure to those octets.

NOTE 2 – 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 3 – 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 4 – 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 some 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,
  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 octets --
  -- of a value of -- ToBeSigned -- and then applying an encipherment procedure to those octets --

```

SIGNATURE { ToBeSigned } ::= **SEQUENCE {**
algorithmIdentifier **AlgorithmIdentifier,**
encrypted **ENCRYPTED-HASH { ToBeSigned }}**

NOTE 5 – The encryption procedure requires the agreements listed in Note 2, and also 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 }}**

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 ITU-T Rec. X.690 (1997) | ISO/IEC 8825:1998, with the following restrictions:

- a) the definite form of length encoding shall be used, encoded in the minimum number of octets;
- b) for string types, the constructed form of encoding shall not be used;
- c) if the value of a type is its default value, it shall be absent;
- d) 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"16;
- 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 ITU-T Rec. X.690 (1997) | ISO/IEC 8825-1:1998;
- j) the encoding of a Generalized time shall be as specified in ITU-T Rec. X.690 (1997) | ISO/IEC 8825-1:1998.

Generating a distinguished encoding requires the abstract syntax of the data to be encoded to be fully understood. The Directory may be required to sign data or check the signature of data that contains unknown protocol extensions or unknown attribute syntaxes. The Directory 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 public key certificate-using system, needs to validate a certificate prior to using that certificate for an application. Procedures for performing that validation are also defined here, including verifying the integrity of the 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, integrity and confidentiality of directory operations; as well as
- integrity and authentication of stored data.

7 Public-keys and public-key certificates

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 the possession of the private key and other procedures related to the issuance of public-key certificates are outside the scope of this 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 certification authority can recover the public key which was certified;
- no party other than the certification authority 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 1 – 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.

A certification authority produces the certificate of a user by signing (see 6.1) a collection of information, including the user's distinguished name and public key, as well as an optional *unique identifier* containing additional information about the user. The exact form of the unique identifier contents is unspecified here and left to the certification authority and might be, for example, an object identifier, a certificate, a date, or some other form of certification on the validity of the distinguished name. Specifically, the certificate of a user with distinguished name A and unique identifier UA, produced by the certification authority with name CA and unique identifier UCA, has the following form:

$$CA\langle\langle A \rangle\rangle = CA\{V, SN, AI, CA, UCA, A, UA, Ap, T^A\}$$

where V is the version of the certificate, SN is the serial number of the certificate, AI is the identifier of the algorithm used to sign the certificate, UCA is the optional unique identifier of the CA, UA is the optional unique identifier of the user A, T^A indicates the period of validity of the certificate, and consists of two dates, the first and last on which the certificate is valid. The certificate validity period is the time interval during which the CA warrants that it will maintain information about the status of the certificate, i.e. publish revocation data. Since T^A is assumed to be changed in periods not less than 24 hours, it is expected that systems would use Coordinated Universal Time as a reference time base. The signature in the certificate can be checked for validity by any user with knowledge of CAp. The following ASN.1 data type can be used to represent certificates:

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

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

CertificateSerialNumber ::= **INTEGER**

```

AlgorithmIdentifier ::= 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 ::= { ... }

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

SubjectPublicKeyInfo ::= SEQUENCE {
  algorithm AlgorithmIdentifier,
  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
  -- contains a DER encoding of a value of type &ExtnType
  -- for the extension object identified by extnId -- }

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

NOTE 2 – 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.

version is the version of the encoded certificate. If the **extensions** component is present in the certificate, version shall be v3. If the **issuerUniqueIdentifier** or **subjectUniqueIdentifier** component is present version shall be v2 or v3.

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 7.5.2.2 in ITU-T Rec. X.519 | ISO/IEC 9594-5.

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

signature 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.)

issuer identifies the entity that has signed and issued the certificate.

validity is the time interval during which the CA warrants that it will maintain information about the status of the certificate.

subject identifies the entity associated with the public-key found in the subject public key field.

subjectPublicKeyInfo is used to carry the public key being certified and to identify the algorithm which this public key is an instance of (e.g. rsaEncryption, dhpPublicNumber, id-dsa, etc.)

issuerUniqueIdentifier is used to uniquely identify an issuer in case of name re-use.

subjectUniqueIdentifier is used to uniquely identify a subject in case of name re-use.

NOTE 3 – In situations where a distinguished name might be reassigned to a different user by the Naming Authority, CAs can use the unique identifier to distinguish between reused instances. However, if the same user is provided certificates by multiple CAs, it is recommended that the CAs coordinate on the assignment of unique identifiers as part of their user registration procedures.

The **extensions** field allows 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 an implementation processing a certificate does not recognize an extension, if 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 certificate to fail. When a certificate-using implementation recognizes and is able to process an extension, then the certificate-using implementation shall process the extension regardless of the value of the criticality flag. Note that any extension that is flagged non-critical will cause inconsistent behaviour between certificate-using systems that will process the extension and certificate-using systems that do not recognize the extension and will ignore it.

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 7.5.2.2 in ITU-T Rec. X.519 | ISO/IEC 9594-5.

A CA 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 validation engine has two possible actions to take with respect to an extension:

- i) it can ignore the extension and accept the certificate (all other things being equal);
- ii) it can 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 validation engine that understands the extension, processes it and acceptance/rejection of the certificate is dependent (at least in part) on the content of the extension. A validation engine that does not understand the extension rejects the certificate.

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

Some extensions can be marked critical or non-critical. In these cases a validation engine that understands the extension processes it and acceptance/rejection of the certificate is dependent (at least in part) on the content of the extension, regardless of the criticality flag. A validation engine 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 certificate, a CA would flag the extension critical. This is done with the realization that any validation engine 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 extension 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 ITU-T Rec. X.660 | ISO/IEC 9834-1. Standard extensions for certificates are defined in clause 8 of this Directory Specification.

The following object class is used to define specific extensions.

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

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

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

A CA-certificate is a certificate issued by a CA to a subject that is itself a CA and therefore is capable of issuing public-key certificates. CA-certificates can be themselves categorized by the following types:

- Self-issued certificate – This is a 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 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.

The directory entry of each user, A, who is participating in strong authentication, contains the certificate(s) of A. Such a certificate is generated by a Certification Authority of A, which is an entity in the DIT. A Certification Authority of A, which may not be unique, is denoted CA(A), or simply CA if A is understood. The public key of A can thus be discovered by any user knowing the public key of CA. Discovering public keys is thus recursive.

If user A, trying to obtain the public key of user B, has already obtained the public key of CA(B), then the process is complete. In order to enable A to obtain the public key of CA(B), the directory entry of each Certification Authority, X, contains a number of certificates. These certificates are of two types. First, there are forward certificates of X generated by other Certification Authorities. Second, there are reverse certificates generated by X itself which are the certified public keys of other certification authorities. The existence of these certificates enables users to construct certification paths from one point to another.

A list of certificates needed to allow a particular user to obtain the public key of another, is known as a *certification path*. Each item in the list is a certificate of the certification authority of the next item in the list. A certification path from A to B (denoted A→B):

- starts with a certificate produced by CA(A), namely CA(A)«X1» for some entity X1;
- continues with further certificates Xi«Xi+1»;
- ends with the certificate of B.

A certification path logically forms an unbroken chain of trusted points in the Directory Information Tree between two users wishing to authenticate. The precise method employed by users A and B to obtain certification paths A→B and B→A may vary. One way to facilitate this is to arrange a hierarchy of CAs, which may or may not coincide with all or part of the DIT hierarchy. The benefit of this is that users who have CAs in the hierarchy may establish a certification path between them using the Directory without any prior information. In order to allow for this each CA may store one certificate and one reverse certificate designated as corresponding to its superior CA.

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

Certificates	::= SEQUENCE {
userCertificate	Certificate,
certificationPath	CertPath OPTIONAL }
CertificationPath	::= SEQUENCE {
userCertificate	Certificate,
theCACertificates	SEQUENCE OF CertificatePair OPTIONAL }

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.

CertPath	::= SEQUENCE OF CrossCertificates
CrossCertificates	::= SET OF Certificate

Each certificate in a certification path shall be unique. No 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 **CertPath**.

7.1 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 user 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 certification authority's public key. The use of this card shall additionally be secured by, e.g. at least 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, 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 certification authority 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.

7.2 Public-key certificate creation

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

- a) a certification authority shall be satisfied of the identity of a user before creating a certificate for it;
- b) a certification authority shall not issue certificates for two users with the same name.

It is important that the transfer of information to the certification authority 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 certificate for a user with a public key that had been tampered with.
- b) If the means of generation of key pairs of 7.1 b) or of 7.1 c) is employed, the user's private key shall be transferred to the user in a secure manner.
- c) If the means of generation of key pairs of 7.1 a) or of 7.1 b) is employed, the user may use different methods (on-line or off-line) to communicate its public key to the CA in a secure manner. On-line 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 transportation to the Directory. As it is produced by an off-line certification authority on behalf of a user who shall be given a copy of it, the user need only store this information in its directory entry on a subsequent access to the Directory. Alternatively the CA could lodge the certificate for the user, in which case this agent shall be given suitable access rights.

7.3 Certificate Validity

The authority that issues certificates (public-key or attribute) also has the responsibility to indicate the validity of certificates it issues. Generally, certificates are subject to possible subsequent revocation. This revocation, and 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 authority to perform revocation.

Authorities that do revoke certificates are required to state, through some similar means, what mechanism(s) can be used by relying parties to obtain revocation status information about certificates issued by that authority. This Specification defines a Certificate Revocation List (CRL) mechanism but does not preclude the use of alternative mechanisms. Relying parties check revocation status information, as appropriate, for all certificates considered during path processing procedure described in clause 10 and the delegation path processing procedure described in clause 16 to validate a certificate.

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.

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 of time 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, or the user is no longer to be certified by the authority, or if the authority's certificate is assumed to be compromised. The revocation of a user certificate or authority certificate shall be made known by the authority, and a new certificate shall be made available, if appropriate. The authority may then inform the holder of the certificate about its revocation by some off-line procedure.

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, on-line certificate status protocol or some other mechanism for the publication of revocation status information;
- c) if using CRLs, shall maintain and publish CRLs even if the lists of revoked certificates are empty.

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 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:

- Certificate revocation list;
- Authority revocation list;
- Delta revocation list;
- Attribute certificate revocation list;
- Attribute authority revocation list.

CertificateList version signature issuer thisUpdate nextUpdate	::=	SIGNED { SEQUENCE { Version OPTIONAL, <i>-- if present, version shall be v2</i> AlgorithmIdentifier, Name, Time, Time OPTIONAL,
---	-----	---

revokedCertificates	SEQUENCE OF SEQUENCE {
serialNumber	CertificateSerialNumber,
revocationDate	Time,
crlEntryExtensions	Extensions OPTIONAL } OPTIONAL,
crlExtensions [0]	Extensions OPTIONAL }}

version is the version of the encoded revocation list. If the **extensions** component flagged as critical is present in the revocation list, version shall be v2. If no **extensions** component flagged as critical is present in the revocation list, version may either be absent or present as v2.

signature contains the algorithm identifier for the algorithm used by the authority to sign the revocation list.

issuer identifies the entity that has signed and issued the revocation list.

thisUpdate is the date/time at which this revocation list was issued.

nextUpdate, if present, indicates 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 will not be issued any later than the indicated time.

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

crlExtensions, if present, contains one or more CRL extensions.

NOTE 1 – 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 2 – 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 timestamped revocation lists are archived and certified by a current authority.

NOTE 3 – If any extensions included in a **CertificateList** are defined as critical, the version element of the **CertificateList** shall be present. If no extensions defined as critical are included, the version element may be absent. If **version** is absent, this may permit a implementation that only supports version 1 CRLs to still use the CRL if in its examination of the **revokedCertificates** sequence in the CRL, it does not encounter an extension. An implementation that supports version 2 (or greater) CRLs, in the absence of version, may also be able to optimize its processing if it can determine early in processing that no critical extensions are present in the CRL.

NOTE 4 – When an implementation processing a certificate revocation list does not recognize a critical extension in the **crlEntryExtensions** field, it shall assume that, at a minimum, the identified certificate has been revoked and is no longer valid and perform additional actions concerning that revoked certificate as dictated by local policy. When an implementation does not recognize a critical extension in the **crlExtensions** field, it shall assume that identified certificates have been revoked and are no longer valid. However in the latter case, since the list may not be complete, certificates that have not been identified as being revoked cannot be assumed to be valid. In this case local policy shall dictate the action to be taken. In any case local policy may dictate actions in addition to and/or stronger than those stated in this Specification.

NOTE 5 – 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 that extension shall be indicated as critical in the **crlExtensions** field regardless of where the extension is placed in the CRL. An extension indicated in the **crlEntryExtensions** field of an entry shall be placed in that entry and shall affect only the certificate(s) specified in that entry.

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

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:

- 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.
- 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 certificate user 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 interoperation 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 certificate users 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 certificate-using system does not recognize the extension field type or does not implement the semantics of the extension, then that system shall consider the certificate invalid. If an extension is flagged non-critical, a certificate-using system 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 certificate-using system 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 certificate-using implementations which do not need to use such extensions to omit support for them without jeopardizing the ability to interoperate with all certification authorities.

NOTE – A certificate-using system 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 certificate user or may be a CA policy rule indicated to the certificate-using system 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 certificate user, the certification authority 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 can be referenced by a unique identifier, which may be included in the certificate policies extension of the certificate issued by the certification authority, to the end-entity and upon which the certificate user relies. A certificate may be issued in accordance with one or more than one policy. Definition of the policy, and assignment of the identifier, are 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 Specification does not prescribe the style or contents of the certificate policy.

The certificate user may be bound to its obligations under the certificate policy by the act of importing an authority public key and using it as a trust anchor, or by relying on a certificate that includes the associated policy identifier. The certification authority 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 some other means.

For an entity to simply 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 so stated in the policy, certificate users may accept the certification authority'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 Specification.

A certification authority 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 certificate users, 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-certification

A certification authority may be the subject of a certificate issued by another certification authority. In this case, the certificate is called a cross-certificate, the certification authority that is the subject of the certificate is called the subject certification authority and the certification authority that issues the cross-certificate is called an intermediate certification authority (see Figure 2). Both the cross-certificate and the end-entity's certificate may contain a certificate policies extension.

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

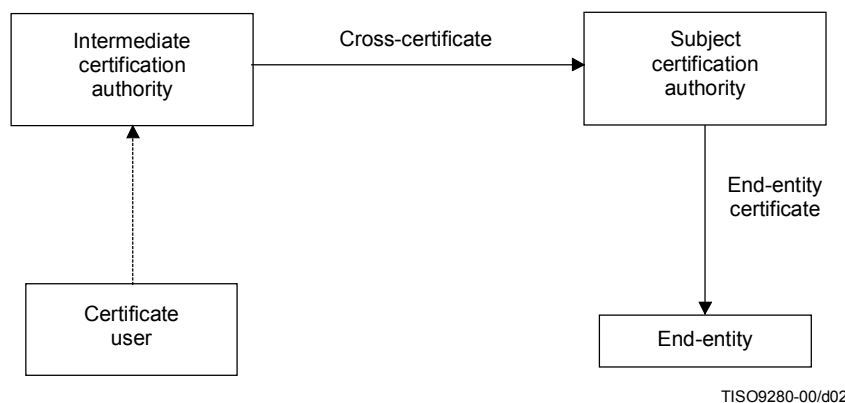


Figure 2 – Cross-certification

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

An intermediate certification authority may, in turn, be the subject of a certificate issued by another certification authority, thereby creating certification paths of length greater than two certificates. And, since trust suffers dilution as certificate paths grow in length, controls are required to ensure that end-entity certificates with an unacceptably low associated trust level will be rejected by the certificate user. This is part of the function of the certification path processing procedure.

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

- a) the certification authority does not use the certificate policies extension to convey its policy requirements to certificate users; and
- b) the certificate user or intermediate certification authority delegates the job of controlling policy to the next authority in the path.

In the first case, the 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. Certificate users shall still ensure that they are using the certificate in conformance with the policies of the authorities in the path.

In the second case, the certificate user or intermediate certification authority should include the special value *any-policy* in the *initial-policy-set* or cross-certificate. Where a 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 certificate user can ensure that all its obligations are conveyed in accordance with the standard by setting the *initial-explicit-policy* indicator. In this way, only authorities that use the standard certificate policies extension as their way of achieving binding are accepted in the path, and certificate users have no additional obligations. Because authorities also attract obligations when they act as, or on behalf of, a certificate user, they can ensure that all their obligations are conveyed in accordance with the standard by setting **requireExplicitPolicy** 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 certification authority issues certificates to end-entities, even though the policy authorities under which the two certification authorities operate may have selected different unique identifiers for these materially equivalent policies. In this case, the intermediate certification authority may include a policy mappings extension in the cross-certificate. In the policy mappings extension, the intermediate certification authority assures the certificate user that it will continue to enjoy the familiar warranties, and that it should continue to fulfill its familiar obligations, even though subsequent entities in the certification path operate in a different policy domain. The intermediate certification authority 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 certification authority operates is identical to those according to which the intermediate certification authority 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 certificate user.

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

Certificate users may determine that certificates issued in a policy domain other than its own should not be relied upon, even though a trusted intermediate certification authority 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 certification authority may make a similar determination on behalf of its certificate users. In order to ensure that certificate users correctly enforce this requirement, it can set **inhibitPolicyMapping** in a policy constraints extension.

8.1.4 Certification path processing

The certificate user 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 certificate user knows, a priori, the set of policies that are acceptable for its intended use.

The second strategy may be most appropriate when the certificate user 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 certificate user 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 certificate user can cause the procedure to return a valid result if the path is valid under any (unspecified) policy.

8.1.5 Self-issued certificates

There are three circumstances under which a certification authority may issue a certificate to itself:

- a) as a convenient way of encoding its public key for communication to, and storage by, its certificate users;
- b) for certifying key usages other than certificate and CRL signing (such as time-stamping); and
- c) for replacing its own expired certificates.

These types of certificate 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 purposes of path validation, self-issued certificates of type a) are 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 certification authority 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 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 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.

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 certificate-using system 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 certificate user 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 certificate-using system 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 signature verifier 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 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 noted otherwise, these extensions may be used in both CA-certificates and end-entity certificates.

8.2.2.1 Authority key identifier extension

This field, which may be used as either a certificate extension or CRL extension, identifies the public key to be used to verify the signature on this certificate or CRL. It enables distinct keys used by the same CA to be distinguished (e.g. as key updating occurs). This field is defined as follows:

```

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 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 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, which indicates the purpose for which the certified public key is used, is defined as follows:

```

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

KeyUsage ::= BIT STRING {
  digitalSignature      (0),
  nonRepudiation        (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 have purposes other than those identified in b), f), or g) below;
- b) **nonRepudiation**: for verifying digital signatures used in providing a non-repudiation service which protects against the signing entity falsely denying some action (excluding certificate or CRL signing, as in f) or g) below);
- 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;
- g) **cRLSign**: for verifying an authority's signature on CRLs.
- 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);

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

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

If the extension is flagged critical, 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, 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 certificate-using system recognizes and processes the **keyUsage** extension type, then the certificate-using system shall ensure that the certificate shall be used only for a purpose for which the corresponding key usage bit is set to one. A bit set to zero indicates that the key is not intended for that purpose. If all bits are zero, it indicates the key is intended for some purpose other than those listed.

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

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

This extension may, at the option of the certificate issuer, 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 certificate-using system recognizes and processes the **extendedKeyUsage** extension type, then the certificate-using system 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.

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 certificate-using system 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 certificate user 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 certificate user, 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 CCITT Rec. X.660 | ISO/IEC 9834-1. A CA may assert any-policy by using the **anyPolicy** 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 Specification. No object identifiers will be assigned in this Specification for specific certificate policies. That assignment is the responsibility of the entity that defines the certificate policy.

anyPolicy OBJECT IDENTIFIER ::= { 2 5 29 32 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 certificate user 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 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, X.400 originator/recipient addresses, and EDI party names. It is therefore necessary to be able to securely associate multiple names of a variety of name forms with a certificate subject or a certificate or CRL issuer.
- b) A certificate user may need to securely 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 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 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]     OAddress,
  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]     DirectoryString {ub-name} OPTIONAL,
  partyName          [1]     DirectoryString {ub-name} }

```

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

- **otherName** is a name of any form defined as an instance of the **OTHER-NAME** information object class;
- **rfc822Name** is an Internet electronic mail address defined in accordance with Internet RFC 822;
- **dnsName** is an Internet domain name defined in accordance with Internet RFC 1035;
- **x400Address** is an O/R address defined in accordance with ITU-T Rec. X.411 | ISO/IEC 10021-4;
- **directoryName** is a directory name defined in accordance with ITU-T Rec. X.501 | ISO/IEC 9594-2;
- **ediPartyName** 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;

- **uniformResourceIdentifier** is a Uniform Resource Identifier for the World-Wide Web defined in accordance with Internet RFC 1630;
- **iPAddress** is an Internet Protocol address defined in accordance with Internet RFC 791, represented as a binary string.
- **registeredID** is an identifier of any registered object assigned in accordance with CCITT Rec. 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 certificate issuer and certificate users.

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 certificate-using system is permitted to ignore any name with an unrecognized or unsupported name form. It is recommended that, provided the subject field of the certificate contains a directory 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 ITU-T Rec. X.681 | ISO/IEC 8824-2.

NOTE 2 – If this extension field is present and is flagged critical, the **subject** field of the 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.

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 is present shall be recognized and processed, otherwise the certificate or CRL shall be considered invalid. Apart from the preceding restriction, a certificate-using system 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 directory name that unambiguously identifies the issuing authority, this field be flagged non-critical.

NOTE – If this extension field is present and is flagged critical, the **issuer** field of the certificate or CRL may contain a null name (e.g. a sequence of zero relative distinguished names) in which case the issuer is identified only by the name or names in this extension.

8.3.2.3 Subject directory attributes extension

This field conveys any desired Directory attribute values for 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

This extension is always non-critical.

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:

- End-entity 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.
- A CA needs to be able to specify constraints which allow a certificate user 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 certificate user 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 certificate user.

- 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 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. Directory name structures considered natural for organizations or geographical areas shall not need adjustment in order to accommodate certification authority requirements.
- i) Certificate extension fields need to be backward-compatible with the unconstrained certification path approach system as specified in earlier editions of ITU-T Rec. X.509 | ISO/IEC 9594-8.
- j) A CA needs to be able to inhibit use of policy mapping and to require explicit certificate policy identifiers to be present in subsequent certificates in a certification path.

NOTE – In any certificate-using system, 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) *Basic constraints*;
- b) *Name constraints*;
- c) *Policy constraints*;
- d) *Inhibit any policy*.

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 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 indicates if the certified public key may be used to verify certificate signatures.

The **pathLenConstraint** component shall be present only if **cA** is set to true. It gives the maximum number of CA-certificates that may follow this certificate in a certification path. Value 0 indicates that the subject of this certificate may issue certificates only to end-entities and not to further CAs. If no **pathLenConstraint** field appears in any certificate of a certification path, there is no limit to the allowed length of the certification path.

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 an entity which is not authorized to be a CA may issue certificates and a certificate-using system 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 certificate-using system, 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 certificate-using system shall check that the certification path being processed is consistent with the value of **pathLenConstraint**.

NOTE 1 – If this extension is not present, or is flagged non-critical and is not recognized by a certificate-using system, then the certificate is to be considered an end-entity certificate and cannot be used to verify certificate signatures.

NOTE 2 – To constrain a certificate subject to being only an end entity, i.e. not a CA, the issuer can 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 a name space within which all subject names in subsequent certificates in a certification path shall be located. 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 }

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

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

BaseDistance ::= INTEGER (0..MAX)

```

If present, the **permittedSubtrees** and **excludedSubtrees** components each specify one or more naming subtrees, each defined by the name of the root of the subtree and, optionally, within that subtree, an area that is bounded by upper and/or lower levels. If **permittedSubtrees** is present, of all the certificates issued by the subject CA and subsequent CAs in the certification path, only those certificates with subject names within these subtrees are 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 and the name spaces overlap, the exclusion statement takes precedence.

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. Conformant implementations are not required to recognize all possible name forms. If the extension is flagged 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 non-critical and a certificate-using implementation does not recognize a name form used in any **base** component, then that subtree specification may be ignored. When a certificate subject has multiple names of the same name form (including, in the case of the **directoryName** name form, the name in the subject field of the certificate if non-null) then all such names shall be tested for consistency with a name constraint of that name form.

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

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 naming 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 naming subtree excludes all nodes except the subtree base and its immediate subordinates.

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 certificate user may not check that subsequent certificates in a certification path are located in the name space intended by the issuing CA.

If this extension is present and is flagged critical, or is flagged non-critical but is recognized by the certificate-using system, then the certificate-using system shall check that the certification path being processed is consistent with the value in this 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 }

SkipCerts ::= INTEGER (0..MAX)

```

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 subject 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 certificate user 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 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 certificate user 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) Certificate users need to be able to track all CRLs issued from a CRL issuer or CRL distribution point (see 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 certificate user. 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.
- e) Certificate users 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.
- f) Issuers need the ability to dynamically change the partitioning of CRLs and to refer certificate users to the new location for relevant CRLs if the partitioning changes.
- g) Delta CRLs may also be available that update a given base CRL. Certificate users 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.

8.5.2 CRL and CRL entry extension fields

The following extension fields are defined:

- a) *CRL number*;
- b) *Reason code*;
- c) *Hold instruction code*;
- d) *Invalidity date*;
- e) *CRL scope*;
- f) *Status referral*;
- g) *CRL stream identifier*;
- h) *Ordered list*;
- i) *Delta information*.

The CRL number, CRL scope, status referral, CRL stream identifier, ordered list and delta information shall be used only as a CRL extension field and the other fields shall be used only as CRL entry extension fields.

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 Reason code extension

This CRL entry extension field identifies a reason for the certificate revocation. The reason code may be used by applications to decide, based on local policy, how to react to posted revocations. This field is defined as follows:

```
reasonCode EXTENSION ::= {
  SYNTAX          CRLReason
  IDENTIFIED BY   id-ce-reasonCode }
```

```
CRLReason ::= ENUMERATED {
  unspecified      (0),
  keyCompromise   (1),
  cACompromise     (2),
  affiliationChanged (3),
  superseded      (4),
  cessationOfOperation (5),
  certificateHold  (6),
  removeFromCRL   (8),
  privilegeWithdrawn (9),
  aaCompromise    (10) }
```

The following reason code values indicate why a certificate was revoked:

- **keyCompromise** is used in revoking an end-entity certificate; it indicates that it is known or suspected that the subject's private key, or other aspects of the subject validated in the certificate, have been compromised;
- **cACompromise** is used in revoking a CA-certificate; it indicates that it is known or suspected that the subject's private key, or other aspects of the subject validated in the certificate, have been compromised;
- **affiliationChanged** indicates that the subject's name or other information in the certificate has been modified but there is no cause to suspect that the private key has been compromised;
- **superseded** indicates that the certificate has been superseded but there is no cause to suspect that the private key has been compromised;
- **cessationOfOperation** indicates that the certificate is no longer needed for the purpose for which it was issued but there is no cause to suspect that the private key has been compromised;
- **privilegeWithdrawn** indicates that a certificate (public-key or attribute certificate) was revoked because a privilege contained within that certificate has been withdrawn;
- **aaCompromise** indicates that it is known or suspected that aspects of the AA validated in the attribute certificate, have been compromised.

A certificate may be placed on hold by issuing a CRL entry with a reason code of **certificateHold**. The certificate hold notice may include an optional hold instruction code to convey additional information to certificate users (see 8.5.2.3). Once a hold has been issued, it may be handled in one of three ways:

- a) it may remain on the CRL with no further action, causing users to reject transactions issued during the hold period; or,
- b) it may be replaced by a (final) revocation for the same certificate, in which case the reason shall be one of the standard reasons for revocation, the revocation date shall be the date the certificate was placed on hold, and the optional instruction code extension field shall not appear; or,
- c) it may be explicitly released and the entry removed from the CRL.

The **removeFromCRL** reason code is for use with delta-CRLs (see 8.6) only and indicates that an existing CRL entry should now be removed owing to certificate expiration or hold release. An entry with this reason code shall be used in delta-CRLs for which the corresponding base CRL or any subsequent (delta or complete for scope) CRL contains an entry for the same certificate with reason code **certificateHold**.

This extension is always non-critical.

8.5.2.3 Hold instruction code extension

This CRL entry extension field provides for inclusion of a registered instruction identifier to indicate the action to be taken on encountering a held certificate. It is applicable only in an entry having a **certificateHold** reason code. This field is defined as follows:

```
holdInstructionCode EXTENSION ::= {
  SYNTAX          HoldInstruction
  IDENTIFIED BY   id-ce-instructionCode }
```

```
HoldInstruction ::= OBJECT IDENTIFIER
```