Internet X.509 Public Key Infrastructure
LDAP Schema for X.509 Attribute Certificates
<draft-ietf-pkix-ldap-ac-schema-02.txt>

Copyright (C) The Internet Society (2004). All Rights Reserved.

"By submitting this Internet-Draft, we certify that any applicable patent or other IPR claims of which we are aware have been disclosed, or will be disclosed, and any of which we become aware will be disclosed, in accordance with RFC 3668 [22]."

STATUS OF THIS MEMO

This document is an Internet-Draft and is in full conformance with all the provisions of Section 10 of RFC2026 [1].

Comments and suggestions on this document are encouraged. Comments on this document should be sent to the PKIX working group discussion list <ietf-pkix@imc.org> or directly to the authors.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

ABSTRACT

This document describes an LDAP schema for X.509 attribute certificates (ACs). Each AC is broken down into a set of attribute types. These attributes can then be stored in an AC entry. An object class is defined for this AC entry. Each attribute type uses an existing LDAP syntax, so that no new matching rules need to be defined.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", ...
It currently isn’t possible to search LDAP servers for X.509 attributes (public key certificates, CRLs etc.) as no matching rules have been defined for them. A couple of Internet Drafts [9,10] have been specified, but implementation of them is complex. Component matching [19] defines a mechanism for matching against complex syntaxes, by defining generic matching rules that can match against any user selected component parts in an attribute value of any arbitrarily complex attribute syntax. This might prove to be the proper way to solve LDAP search problems in the longer term, but it will take a long time until such ASN.1 based mechanisms are implemented in all LDAP servers and clients. Even when this has happened the mechanism proposed in this document will still be useful to some applications such as CIP [20].

A simple and easy to implement mechanism is needed today to search for X.509 attributes. Rather than search for an X.509 attribute in an entry, it suggests the directory administrative user creates an entry (in the case of public key and attribute certificates) or a subtree (in the case of CRLs) from the X.509 attribute. The attributes of these new entries will be created from fields of the X.509 attribute (e.g. the issuer field), and if these new attributes are defined using existing LDAP syntaxes and matching rules, then it will be possible to use existing LDAP server technology to search for fields in X.509 attributes.
This document is one of a set comprising:
i) the LDAP schema for X.509 public key certificates [7]
ii) the LDAP schema for X.509 attribute certificates (this document)
iii) the LDAP schema for X.509 CRLs [8]

Schema definitions are provided using LDAPv3 description formats from RFC2252 [3]. Definitions provided here are formatted (line wrapped) for readability. The specifications use the augmented Backus-Naur Form (ABNF) as described in RFC2234 [4].

2. DIT Structure and Naming

If the schema presented in this document is used to store information about ACs in an LDAP directory, each AC SHOULD be stored as a direct subordinate of the AC holder’s entry. These entries SHOULD be named using either the x509ACNameForm i.e. by a multi-valued RDN formed by the AC issuer and serial number, or by the x509ACAltNameForm i.e. by a single valued RDN formed by concatenating the AC issuer and serial number, as these are the only ways to enforce unique RDNs under the holder’s entry. Exceptionally, if it can be guaranteed that only ACs from a single issuer will be stored under the holder’s entry, the x509ACserialNumberNameForm MAY be used, i.e. the single valued RDN formed from the AC serial number.

(1.2.826.0.1.3344810.1.3.3
  NAME ’x509ACNameForm’
  OC x509AC
  MUST ( x509serialNumber $ x509issuer ) )

(1.2.826.0.1.3344810.1.3.4
  NAME ’x509ACAltNameForm’
  OC x509AC
  MUST ( x509issuerSerial ) )

(1.2.826.0.1.3344810.1.3.5
  NAME ’x509ACserialNumberNameForm’
  OC x509AC
  MUST ( x509serialNumber ) )

The following attribute description describes the attribute used to hold the alternative RDN name form.

(1.2.826.0.1.3344810.1.1.60
  NAME ’x509issuerSerial’
  DESC ’Used to hold the RDN of a certificate entry, formed by concatenating the AC serial number and issuer fields’
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE )

When encoding DNs that contain an x509issuer field, the string representation must be made according to [13]. These strings contain RFC2253 special characters and must therefore be escaped. For example, the issuer name in a certificate may be:
When used in the x509IssuerSerial attribute of a DN, this may appear as:

dn: x509IssuerSerial=123456,OU=VeriSign Trust Network,OU=(c) 1998 VeriSign Inc.-For authorized use only,OU=Class 1 Public Primary Certification Authority-G2,O=VeriSign Inc.,C=US

3. X.509 Schema Object Classes

The object classes have been designed to form a logical set and be extensible in an orderly way as new PKC/CRL/AC extensions are defined. The methodology is as follows. Every X.509 entry (for a PKC, CRL or AC) is of the x509base abstract object class. There is then an additional abstract object class for each, derived from x509base, which holds the attributes extracted from the basic PKC/AC/CRL ASN.1 structure (excluding all extensions). Further, there is an auxiliary object class for the extensions defined in X.509 [6], and an additional auxiliary object class (if needed) for extensions defined in existing Internet RFCs. As new extensions are defined, then new auxiliary object classes and attributes will need to be defined to cater for the attributes to be extracted from these. Finally there are several structural object classes for each, which allow the X.509 DER encoded attribute to be stored in the entry.

3.1 X.509 AC object class

The X.509AC abstract object class is used to hold the attributes extracted from the basic fields of an attribute certificate. The X.509 AC object class is the abstract object class from which the structural object classes for attributeCertificate entries and AA certificate entries are derived.

(1.2.826.0.1.3344810.1.0.16
 NAME 'x509AC'
 SUP x509base
 ABSTRACT
 MUST { x509version $,
 x509serialNumber $,
 x509signatureAlgorithm $,
 x509issuer $,
 x509validityNotBefore $,
 x509validityNotAfter }

 MAY { x509ACHolderPKCSerialNumber $,
 x509ACHolderPKCissuerDN $,
 x509ACHolderRfc822Name $,
 x509ACHolderDNSName $,
 x509ACHolderDN $,
 x509ACHolderURI $,
 x509ACHolderIPAddress $,
 x509ACHolderRegisteredID $,
 x509authorityCertIssuer $,
 x509authorityCertSerialNumber $,
 x509authorityKeyIdentifier $ }
The definition of x509base can be found in [7].

3.2 X.509 attribute certificate object class

This structural object class is for entries of attribute certificates belonging to holders.

(1.2.826.0.1.3344810.1.0.17
 NAME ‘x509attributeCertificate’
 SUP x509AC
 MUST attributeCertificateAttribute )

The attributeCertificateAttribute is defined in [10].

3.3 X.509 AA certificate object class

This structural object class is for entries of attribute certificates belonging to Attribute Authorities.

(1.2.826.0.1.3344810.1.0.18
 NAME ‘x509aACertificate’
 SUP x509AC
 MUST aACertificate )

The aACertificate attribute is defined in [10].

3.4 Embedded attributes

The x509AC object class does not contain the attributes embedded in the attribute certificate, since these can be attributes of any type. Therefore the LDAP entry created to hold the AC should also be of the auxiliary object classes appropriate for the attributes embedded in the AC. One pragmatic solution to this is to make the entry of object class extensibleObject [3].

3.5 X.509 AC extensions auxiliary object class

The x509ACext auxiliary object class is used to hold the attributes extracted from the AC extensions defined in the X.509 standard [6] and profiled in [5].

Note. If an AC holds additional extensions to these, then another auxiliary object class and supporting attributes will need to be defined.
4. Common X.509 Attribute Types

The following attribute types defined in [7] are used to hold the corresponding fields of ACs:

- `x509serialNumber` - used to hold the serial number of the AC
- `x509version` - used to hold the version of the AC
- `x509signatureAlgorithm` - used to hold the OID of the algorithm used to sign the CRL
- `x509issuer` - used to hold the DN of the AC issuer
- `x509validityNotBefore` - used to hold the not before validity time of the AC (note that only the Generalized Time format is permitted)
- `x509validityNotAfter` - used to hold the not after validity time of the AC (note that only the Generalized Time format is permitted)
- `x509authorityCertIssuer` - used in conjunction with `x509authorityCertSerialNumber` to identify the public key certificate of the AC issuer
- `x509authorityCertSerialNumber` - used in conjunction with `x509authorityCertIssuer` to identify the public key certificate of the AC issuer
- `x509authorityCertIssuer` - used in conjunction with `x509authorityCertSerialNumber` to identify the public key certificate of the AC issuer
- `x509issuerRfc822Name` - used to hold the email address of the AC issuer
- `x509issuerDNSName` - used to hold the DNS name of the AC issuer
- `x509issuerURI` - used to hold a URI for the AC issuer
- `x509IssuerIPAddress` - used to hold the IP address of the AC issuer
- `x509IssuerRegisteredID` - used to hold a registered OID of the AC issuer
- `x509AuthorityKeyIdentifier` - used to hold the identifier of the public key used to sign the AC, taken from the attribute cert issuer object digest field

5. Attribute Types for AC Specific Fields

The following attribute types may be used to store basic fields of an AC. The following basic fields are supported:

- `x509ACHolderPKCSerialNumber` and `x509ACHolderPKCissuerDN` - used to identify the holder via their public key certificate
- `x509ACHolderRfc822Name` - identifies the holder via their email address
- `x509ACHolderDNSName` - identifies the holder via their DNS name
- `x509ACHolderDN` - identifies the holder via their DN
- `x509ACHolderURI` - identifies the holder via their URI
- `x509ACHolderIPAddress` - identifies the holder via their IP address
- `x509ACHolderRegisteredID` - identifies the holder via a registered OID
- `x509ACObjectDigest`, `x509ACDigestAlgorithm` and `x509ACDigestedObjectType` - identifies the holder via a hash of information directly associated with the holder

5.1 AC holder PKC

The `x509ACHolderPKCSerialNumber` and `x509ACHolderPKCissuerDN` attributes are to hold the contents of the holder base certificate ID fields, in order to identify the holder via their public key certificate

5.1.1 AC holder PKC serial number

```plaintext
(1.2.826.0.1.3344810.1.1.61
 NAME 'x509ACHolderPKCSerialNumber'
  DESC 'The serial number of the PKC of the AC holder, see RFC3281 4.2.2'
  EQUALITY integerMatch
  ORDERING integerOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
  SINGLE-VALUE )
```

5.1.2 AC holder PKC issuer DN

```plaintext
(1.2.826.0.1.3344810.1.1.62
 NAME 'x509ACHolderPKCissuerDN'
  DESC 'Distinguished name of the issuer of the PKC belonging to the AC holder, see RFC3281 4.2.2'
  EQUALITY distinguishedNameMatch
  ORDERING distinguishedNameOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
```

5.2 AC Holder general names

The following attributes are used to hold the alternative forms of the general name of the holder. Separate attribute types are defined for all choices of the ASN.1 type "GeneralName" except for "otherName", "x400Address" and "ediPartyName".
5.2.1 Holder RFC 822 name

(1.2.826.0.1.3344810.1.1.63
 NAME ‘x509ACHolderRfc822Name’
 DESC ‘Internet electronic mail address of the AC holder, see RFC3281 4.2.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC 822 [11].

5.2.2 Holder DNS name

(1.2.826.0.1.3344810.1.1.64
 NAME ‘x509ACHolderDNSName’
 DESC ‘Internet domain name of the AC Holder, see RFC3281 4.2.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded as Internet domain names in accordance with RFC1035 [12].

5.2.3 Holder directory name

(1.2.826.0.1.3344810.1.1.65
 NAME ‘x509ACHolderDN’
 DESC ‘Distinguished name of the AC Holder, see RFC3281 4.2.2’
 EQUALITY distinguishedNameMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute type must be encoded according to the syntax given in RFC2253 [13].

5.2.4 Holder uniform resource identifier

(1.2.826.0.1.3344810.1.1.66
 NAME ‘x509ACHolderURI’
 DESC ‘Uniform Resource Identifier of the AC Holder, see RFC3281 4.2.2’
 EQUALITY caseExactIA5Match
 SUBSTR caseExactIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2396 [14].
5.2.5 Holder IP address

(1.2.826.0.1.3344810.1.1.67
 NAME ‘x509ACHolderIPAddress’
 DESC ‘Internet Protocol address of the AC Holder, see RFC3281 4.2.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute type must be stored in the syntax given in Appendix B of RFC2373 [16].

5.2.6 Holder registered ID

(1.2.826.0.1.3344810.1.1.68
 NAME ‘x509ACHolderRegisteredID’
 DESC ‘Any registered OID of the AC holder, see RFC3281 4.2.2’
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

registeredID is an identifier of any registered object assigned in accordance with ITU-T Rec. X.660. [17]

5.3 AC object digest

x509ACObjectDigest, x509ACDigestAlgorithm and x509ACDigestedObjectType are used to hold the contents of the holder object digest info fields. They are used to identify the holder via a hash of information directly associated with the holder.

5.3.1 Object digest

( 1.2.826.0.1.3344810.1.1.69
 NAME ‘x509ACObjectDigest’
 DESC ‘Holds the hash value of the object identified by x509ACDigestedObjectType, see RFC 3281, section 7.3’
 EQUALITY bitStringMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 SINGLE-VALUE )

5.3.2 Object digest algorithm

( 1.2.826.0.1.3344810.1.1.70
 NAME ‘x509ACDigestAlgorithm’
 DESC ‘OID of the hashing algorithm used to create the Object digest, see RFC3281, section 7.3’
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 SINGLE-VALUE )
5.3.3 Object type

(1.2.826.0.1.3344810.1.1.71
   NAME 'x509ACDigestedObjectType'
   DESC 'Type of object being digested, see RFC3281, section 7.3'
   EQUALITY integerMatch
   ORDERING integerOrderingMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
   SINGLE-VALUE )

6. Attributes for Selected AC Extensions

In line with the AC profile RFC 3281 [5], the following AC extensions are supported:
- Audit Identity (defined here)
- AC targets (defined here)
- Authority Key Identifier (defined in [7])
- Authority Information Access (defined in [7])
- CRL distribution points (defined here)
- No revocation (defined here)

(Note. The CRL distribution point attributes defined in [7] were inadequate for our needs)

6.1 Audit identity

This attribute may be used to store the sequence number of the CRL.

(1.2.826.0.1.3344810.1.1.72
   NAME 'x509ACAuditID'
   DESC 'Identity of holder used in audit trails, see RFC3281 4.3.1'
   EQUALITY octetStringMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
   SINGLE-VALUE )

6.2 AC targets

ACs can be targeted at specific objects, or groups of objects. Objects and groups of objects are identified by their general names. Separate sets of attributes are specified for individual targets and groups of targets. Attribute types are defined for all choices of the ASN.1 type "GeneralName" except for "otherName", "x400Address" and "ediPartyName".

6.2.1 Target RFC 822 name

(1.2.826.0.1.3344810.1.1.73
   NAME 'x509ACTargetRfc822Name'
   DESC 'Internet electronic mail address of the ACs Target, see RFC3281 4.3.2'
   EQUALITY caseIgnoreIA5Match
   SUBSTR caseIgnoreIA5SubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )
6.2.2 Target DNS name

(1.2.826.0.1.3344810.1.1.74
 NAME ‘x509ACTargetDNSName’
 DESC ‘Internet domain name of the ACs Target, see RFC3281 4.3.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute type must be encoded according to the syntax given in RFC822 [11].

6.2.3 Target directory name

(1.2.826.0.1.3344810.1.1.75
 NAME ‘x509ACTargetDN’
 DESC ‘Distinguished name of the ACs Target, see RFC3281 4.3.2’
 EQUALITY distinguishedNameMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute must be encoded as Internet domain names in accordance with RFC1035 [12].

6.2.4 Target uniform resource identifier

(1.2.826.0.1.3344810.1.1.76
 NAME ‘x509ACTargetURI’
 DESC ‘Uniform Resource Identifier of the ACs Target, see RFC3281 4.3.2’
 EQUALITY caseExactIA5Match
 SUBSTR caseExactIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2253 [13].

6.2.5 Target IP address

(1.2.826.0.1.3344810.1.1.77
 NAME ‘x509ACTargetIPAddress’
 DESC ‘Internet Protocol address of the ACs Target, see RFC3281 4.3.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2396 [14].
Values of this attribute type must be stored in the syntax given in Appendix B of RFC2373 [16].

6.2.6 Target registered ID

(1.2.826.0.1.3344810.1.1.78
 NAME ‘x509ACTargetRegisteredID’
 DESC ‘Any registered OID of the ACs Target, see RFC3281 4.3.2’
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

registeredID is an identifier of any registered object assigned in accordance with ITU-T Rec. X.660. [17]

6.2.7 Target group RFC 822 name

(1.2.826.0.1.3344810.1.1.79
 NAME ‘x509ACTargetGroupRfc822Name’
 DESC ‘Internet electronic mail address of the ACs Target group see RFC3281 4.3.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC 822 [11].

6.2.8 Target group DNS name

(1.2.826.0.1.3344810.1.1.80
 NAME ‘x509ACTargetGroupDNSName’
 DESC ‘Internet domain name of the ACs Target group, see RFC3281 4.3.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded as Internet domain names in accordance with RFC1035 [12].

6.2.9 Target group directory name

(1.2.826.0.1.3344810.1.1.81
 NAME ‘x509ACTargetGroupDN’
 DESC ‘Distinguished name of the AC’s Target group, see RFC3281 4.3.2’
 EQUALITY distinguishedNameMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute type must be encoded according to the syntax given in RFC2253 [13].
6.2.10 Target group uniform resource identifier

(1.2.826.0.1.3344810.1.1.82
 NAME ‘x509ACTargetGroupURI’
 DESC ‘Uniform Resource Identifier of the AC’s Target group
 see RFC3281 4.3.2’
 EQUALITY caseExactIA5Match
 SUBSTR caseExactIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2396 [14].

6.2.11 Target group IP address

(1.2.826.0.1.3344810.1.1.83
 NAME ‘x509ACTargetGroupIPAddress’
 DESC ‘Internet Protocol address of the ACs Target group, see RFC3281 4.3.2’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute type must be stored in the syntax given in Appendix B of RFC2373 [16].

6.2.12 Target group registered ID

(1.2.826.0.1.3344810.1.1.84
 NAME ‘x509ACTargetGroupRegisteredID’
 DESC ‘Any registered OID of the ACs Target group, see RFC3281 4.3.2’
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

registeredID is an identifier of any registered object assigned in accordance with ITU-T Rec. X.660. [17]

6.3 No revocation

(1.2.826.0.1.3344810.1.1.85
 NAME ‘x509ACNoRevocation’
 DESC ‘If true, the AC will never be revoked, see RFC3281 section 4.3.6’
 EQUALITY booleanMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 )

6.4 CRL distribution points

The CRL distribution point extension indicates the locations where CRLs will be
published for this AC. It comprises the general name of the DP, plus optionally
the general name of the CRL issuer (if different from the AC issuer) plus the
reason codes that will be published at this DP. Separate attribute types are
defined for all choices of the ASN.1 type "GeneralName" except for "otherName",
"x400Address" and "ediPartyName". Note that because there can be multiple
distribution points, the multi-valued attributes defined here will not be able
to link each DP with its corresponding reasons and issuer.

If

6.4.1 Distribution point RFC 822 name

(1.2.826.0.1.3344810.1.1.86
 NAME ’x509DPRfc822Name’
 DESC ’Internet electronic mail address of the
distribution point, see RFC3280 section 4.2.1.14’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC

6.4.2 Distribution point DNS name

(1.2.826.0.1.3344810.1.1.87
 NAME ’x509DPDNSName’
 DESC ’Internet domain name of the distribution point, see
 RFC3280 section 4.2.1.14’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded as Internet domain names in accordance
with RFC1035 [12].

6.4.3 Distribution point directory name

(1.2.826.0.1.3344810.1.1.88
 NAME ’x509DPDN’
 DESC ’Distinguished name of the distribution point, see
 RFC3280 section 4.2.1.14’
 EQUALITY distinguishedNameMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute type must be encoded according to the syntax given in
RFC2253 [13].

6.4.4 Distribution point uniform resource identifier

(1.2.826.0.1.3344810.1.1.89
 NAME ’x509DPURI’
 DESC ’Uniform Resource Identifier of the distribution
point, see RFC3280 section 4.2.1.14.
EQUALITY caseExactIA5Match
SUBSTR caseExactIA5SubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2396 [14].

6.4.5 Distribution point IP address

(1.2.826.0.1.3344810.1.1.90
 NAME 'x509DPIPAddress'
 DESC 'Internet Protocol address of the distribution point, see RFC3280 section 4.2.1.14'
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute type must be stored in the syntax given in Appendix B of RFC2373 [16].

6.4.6 Distribution point registered ID

(1.2.826.0.1.3344810.1.1.91
 NAME 'x509DPRegisteredID'
 DESC 'Any registered OID of the distribution point, see RFC3280 section 4.2.1.14'
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

registeredID is an identifier of any registered object assigned in accordance with ITU-T Rec. X.660. [17]

6.4.7 Distribution point name relative to CRL issuer

(1.2.826.0.1.3344810.1.1.92
 NAME 'x509DPrelativeToIssuer'
 DESC 'RDN of the distribution point, relative to the issuer, see RFC3280 section 4.2.1.14'
 EQUALITY distinguishedNameMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute type must be encoded according to the syntax given in RFC2253 [13].

6.4.8 Distribution point CRL issuer RFC 822 name

(1.2.826.0.1.3344810.1.1.93
 NAME 'x509DPissuerRfc822Name'
 DESC 'Internet electronic mail address of the distribution point CRL issuer, see RFC3280 section 4.2.1.14'
 EQUALITY caseIgnoreIA5Match
Values of this attribute must be encoded according to the syntax given in RFC 822 [11].

6.4.9 Distribution point CRL issuer DNS name

(1.2.826.0.1.3344810.1.1.94
 NAME ’x509DPissuerDNSName’
 DESC ’Internet domain name of the distribution point CRL issuer,
 see RFC3280 section 4.2.1.14’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded as Internet domain names in accordance with RFC1035 [12].

6.4.10 Distribution point CRL issuer directory name

(1.2.826.0.1.3344810.1.1.95
 NAME ’x509DPissuerDN’
 DESC ’Distinguished name of the distribution point CRL issuer,
 see RFC3280 section 4.2.1.14’
 EQUALITY distinguishedNameMatch
 SUBSTR caseIgnoreIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

Values of this attribute type must be encoded according to the syntax given in RFC2253 [13].

6.4.11 Distribution point CRL issuer uniform resource identifier

(1.2.826.0.1.3344810.1.1.96
 NAME ’x509DPissuerURI’
 DESC ’Uniform Resource Identifier of the distribution point CRL issuer,
 see RFC3280 section 4.2.1.14’
 EQUALITY caseExactIA5Match
 SUBSTR caseExactIA5SubstringsMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

Values of this attribute must be encoded according to the syntax given in RFC2396 [14].

6.4.12 Distribution point CRL issuer IP address

(1.2.826.0.1.3344810.1.1.97
 NAME ’x509DPissuerIPAddress’
 DESC ’Internet Protocol address of the distribution point CRL issuer,
 see RFC3280 section 4.2.1.14’
 EQUALITY caseIgnoreIA5Match
 SUBSTR caseIgnoreIA5SubstringsMatch
6.4.13 Distribution point CRL issuer registered ID

(1.2.826.0.1.3344810.1.1.98
 NAME ‘x509DPissuerRegisteredID’
 DESC ‘Any registered OID of the distribution point CRL issuer,
     see RFC3280 section 4.2.1.14’
 EQUALITY objectIdentifierMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

registeredID is an identifier of any registered object assigned in accordance
with ITU-T Rec. X.660. [17]

6.4.14 Distribution point reason codes

This attribute is used to indicate the reason codes associated with the various
DPs.

(1.2.826.0.1.3344810.1.1.99
 NAME ‘x509DPReasonCodes’
 DESC ‘The reason codes used by a DP, see
     RFC3280 section 4.2.1.14’
 EQUALITY bitStringMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 )

7. Security Considerations

This [Internet Draft/Standard] describes the subschema for the storage
and matching of PKI attributes derived from CRLs. It does not address the
protocol for the storage and retrieval of this information.

LDAP servers SHOULD use authentication and access control mechanisms to protect
the information during its storage and retrieval.

8. IANA Considerations

This document uses the OID node 1.2.826.0.1.3344810 to identify the LDAP schema
elements described here. This OID is assigned to TrueTrust Ltd, under its BSI
assigned English/Welsh Registered Company number [18].

9. References

Normative


Informative


10. Intellectual Property Notice

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 [21] and BCP 79 [22].

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

11. Acknowledgments

The authors would like to thank Peter Gietz for his help and co-operation, and in particular his willingness to align [7] with this document.

The authors would also like to thank TERENA, CESNET, SURFnet, UNINETT, RedIRIS and SWITCH, who jointly partially funded this work, and without whose support
this work would not have been possible.

12. Copyright

Copyright (C) The Internet Society (2004). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

"This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTORS, THE ORGANIZATION THEY REPRESENT OR ARE SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

"Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than a "work in progress."

The list of current Internet-Drafts can be accessed at
http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at
http://www.ietf.org/shadow.html"

13. Authors’ Addresses

David Chadwick, Mikhail Sahalayev
IS Institute
University of Salford
Salford
England
M5 4WT
14. Changes

Changes from `<draft-ietf-pkix-ldap-ac-schema-00.txt>`

1. Have added a section about attributes embedded in ACs.
2. Have aligned schema with `<draft-klasen-ldap-x509certificate-schema-02.txt>`
3. Have altered object class structure to introduce auxiliary object classes for certificate extensions
4. Have adjusted upper/lower case of components of attribute type names to be consistent
5. Have changed matching rules of x509ACTargetGroupDNSName to be IA5 matching rules
6. Minor editorial corrections
7. Changed from Standards Track to Informational after discussions with area and WG leaders.
8. Have added an IANA considerations section and Acknowledgment section

Changes from `<draft-ietf-pkix-ldap-ac-schema-01.txt>`

1. Updated references.