FINEID - S5

Directory Specification

v3.0

Population Register Centre (VRK)
Certification Authority Services
P.O. Box 123
FIN-00531 Helsinki
Finland
http://www.fineid.fi

Table of Contents

1. Introduction

A central part of the FINEID Public Key Infrastructure and its services is the public directory where end entity certificates (citizen certificates, role certificates, server certificates), Certification Authority (CA) certificates (Root and intermediate CA certificates) and Certificate and Authority Revocation Lists (CRLs, ARLs) are stored and accessed by all users.

Root and intermediate CA certificates, CRLs and ARLs are also available through HTTP-proxy, using HTTP v1.1 protocol.

This document defines the directory schema, tree structure and the contents of the FINEID certificate directory.

Related FINEID specifications:

FINEID S2 – VRK (PRC) CA-model and certificate contents, v3.0

FINEID documentation is available at:

http://www.fineid.fi

References:

- IETF RFC 2256: A Summary of the X.500(96) User Schema for use with LDAPv3, M. Wahl Critical Angle Inc., December 1997
- IETF RFC 4510: Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map, K. Zeilenga (Ed.), OpenLDAP Foundation, June 2006
- IETF RFC 4511: Lightweight Directory Access Protocol (LDAP): The Protocol, J. Sermersheim (Ed.), Novell Inc., June 2006
- IETF RFC 4512: Lightweight Directory Access Protocol (LDAP): Directory Information Models, K. Zeilenga, OpenLDAP Foundation, June 2006
- IETF RFC 4513: Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms, R. Harrison (Ed.), Novell Inc., June 2006
- IETF RFC 4514: Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names, K. Zeilenga (Ed.), OpenLDAP Foundation, June 2006
- IETF RFC 4515: Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters, M. Smith (Ed.), Pearl Crescent, LLC, June 2006
- IETF RFC 4516: Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator, M. Smith (Ed.), Pearl Crescent, LLC, June 2006
- IETF RFC 4517: Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules, S. Legg (Ed.), eB2Bcom, June 2006
- IETF RFC 4518: Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation, K. Zeilenga, OpenLDAP Foundation, June 2006

- IETF RFC 4519: Lightweight Directory Access Protocol (LDAP): Schema for User Applications, A. Sciberras, (Ed.), eB2Bcom, June 2006
- IETF RFC 4523: Lightweight Directory Access Protocol (LDAP) Schema Definitions for X.509 Certificates, K. Zeilenga, OpenLDAP Foundation, June 2006
- IETF RFC 4524: COSINE LDAP/X.500 Schema, K. Zeilenga (Ed.), OpenLDAP Foundation, June 2006
- IETF RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, D. Cooper, NIST, et al., May 2008

2. Background

The FINEID directory schema has been designed to allow the storage of wide range of standardized attributes for each certification authority and user certificate. This makes it possible to extract optional attributes from certificates to directory entries. It also provides the possibility to include additional information from other sources into directory entries.

The use of standard attribute types and object classes provides good interoperability with common LDAP clients. FINEID specific attribute types have been defined to meet the requirements of FINEID directory service where standardized attribute types have not been available. FINEID object classes have been defined to enable packaging of useful information in one directory entry.

Root, intermediate CA and user (end entity) certificates are stored in the directory as separate objects (i.e. there is one directory entry for each certificate).

HTTP proxy can be used to retrieve some most commonly used objects from directory using HTTP v1.1 instead of LDAP.

3. Changes to previous FINEID S5 versions

3.1 Changes from version 2.2

Changes to FINEID S5 v3.0 directory specification are:

- IETF RFC references updated
- New IP-address for proxy.fineid.fi and ldap.fineid.fi services
- VRK postal address updated

3.2 Changes from version 2.1 to 2.2

Major changes to FINEID S5 v2.1 directory specification are:

- Changes related to the change of the Directory Service Provider (new IP)
- Attribute changes
  - object class fineidUserCertificate: removed obsolete dnQualifier and uniqueIdentifier attributes, added st attribute

- object class fineidCertificationAuthority: added st attribute

3.3 Changes from version 1.1 to 2.1

Major changes to FINEID S5 v1.1 directory specification are:

- String representation: Transition from T.61/Teletext string to UTF-8 string
- Additional object classes for improved compatibility
- HTTP v1.1 support for some key objects
- publicKeySHA1Hash attribute
- removal of email addresses from the directory entries containing non-repudiation certificates

4. LDAP settings

The FINEID directory service is accessible with LDAPv2 and LDAPv3 compliant client software. The client software needs to be configured to be able to use the FINEID directory service as follows:

4.1 LDAP parameters

```
LDAP server =  ldap.fineid.fi
(IP-address =  195.226.202.184)
LDAP port =    389
Search base =  dmdName=FINEID,c=FI
```

The directory service does **not** require client authentication.

The directory uses UTF-8 string representation.

The following chapters describe the FINEID directory schema in detail. Knowledge of the schema is essential for an LDAP client to make efficient searches.

The key attributes for efficient LDAP searches are:

- serialNumber (FINUID of subject)
- cn (of CA or subject)
- sn (of subject)
- mail (of subject)
- certificateSerialNumber (of subject)
- publicKeySHA1Hash (of subject)

Contents of the certificates are documented in the FINEID S2 –specification.

4.2 LDAP URLs

LDAP URLs can be used to access the information in the directory. The LDAP URL format is also used to provide CRL Distribution Point information in certificates.

The LDAP URL search base is:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI
```

Examples:

The following LDAP URL retrieves the CRL of the CA which commonName is VRK Gov. CA for Citizen Qualified Certificates:

```
ldap://ldap.fineid.fi/cn%3DVRK%20Gov.%20CA%20for%20Citizen%20Qualif
ied%20Certificates%2C%20ou%3DValtion%20kansalaisvarmenteet%2C%20o%3
DVaestorekisterikeskus%20CA%2CdmdName%3DFINEID%2Cc%3DFI?certificate
RevocationList
```

The following LDAP URL retrieves the CA certificate of the CA which commonName is VRK CA for Qualified Certificates:

```
ldap://ldap.fineid.fi/cn%3DVRK%20CA%20for%20Qualified%20Certificate
s%2C%20ou%3DOrganisaatiovarmenteet%2C%20o%3DVaestorekisterikeskus%2
0CA%2CdmdName%3DFINEID%2Cc%3DFI?caCertificate
```

The following LDAP URL retrieves the directory entry which contain certificate which certificate serial number is 123456:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?certificateser
ialnumber=123456
```

The following LDAP URL retrieves all directory entries containing certificates that have the same unique identifier (990000278) of subject, (the FINUID in the case of Citizen certificates):

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?serialnumber=9
90000278
```

The following LDAP URL retrieves all directory entries containing encryption certificate for user “John Doe”:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?(&(fineidCertK
eyUsageString=3)(cn=doe john*))
```

The following LDAP URL retrieves all directory entries containing nonRepudiation certificate for user “John Doe”:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?(&(fineidCertK
eyUsageString=1)(cn=doe john*))
```

The following LDAP URLs retrieves directory entry containing (email) encryption certificate for user whose email address is john.doe@company.fi:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?(&(fineidCertK
eyUsageString=3)(rfc822mailbox=john.doe@company.fi))
```

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?(rfc822mailbox
=john.doe@company.fi)
```

The following LDAP URL retrieves all directory entries containing nonRepudiation certificates for all users whose given name is “John”:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?(&(fineidCertK
eyUsageString=1)(givenName=john))
```

The following LDAP URL retrieve directory entry containing certificate for user whose SHA-1 hash calculated from public key modulus is D1:6A:BB:FC:94:A0:48:17:71:AB:C5:04:35:E7:86:0E:A6:10:44:DE:

```
ldap://ldap.fineid.fi/dmdName%3DFINEID%2Cc%3DFI??sub?
publicKeySHA1Hash=D16ABBFC94A0481771ABC50435E7860EA61044DE
```

The following LDAP URL retrieve certificate for user whose SHA-1 hash calculated from public key modulus is D1:6A:BB:FC:94:A0:48:17:71:AB:C5:04:35:E7:86:0E:A6:10:44:DE:

```
ldap://ldap.fineid.fi/dmdName=FINEID,c=FI?usercertificate?sub?publi
cKeySHA1Hash=D16ABBFC94A0481771ABC50435E7860EA61044DE
```

All search operations should be narrowed as much as possible by adding organization and organizationalUnit –directory levels into ldap-queries. Performance is better and search results do not contain unnecessary information.

The following query returns same results than previous example:

```
ldap://ldap.fineid.fi/OU=Testivarmenteet,O=Vaestorekisterikeskus
TEST,dmdName=FINEID,c=FI?usercertificate?sub?publicKeySHA1Hash=D16A
BBFC94A0481771ABC50435E7860EA61044DE
```

It is possible that an LDAP search returns multiple directory entries. For example people with the same names, user might have multiple valid certificates issued by the same CA, etc. Only the certificateSerialNumber and publicKeySHA1Hash attributes are guaranteed to be unique. For performance and data content security reasons the directory service might limit number of returned entries, for example to 100 entries.

The LDAP search filter syntax is defined in IETF RFC 4515: “Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters”.

5. HTTP settings

Some objects in the FINEID directory are also accessible using HTTP v1.1 protocol. The client software needs to be configured to be able to retrieve data from directory through the FINEID HTTP proxy:

5.1 HTTP parameters

HTTP server = proxy.fineid.fi
(IP-address = 195.226.202.185)
HTTP port = 80

The HTTP proxy service does **not** require client authentication.

The proxy service retrieves data from the FINEID directory in real time.

5.2 HTTP URLs

HTTP URLs can be used to retrieve Root and CA certificates and Certificate and Authority Revocation Lists from directory. HTTP URL format is also used in

certificates to provide issuer certificate location and ARL and CRL Distribution Point information.

The HTTP URL should be of the following form:

```
http://proxy.fineid.fi/ca/cacert.crt
http://proxy.fineid.fi/crl/cacertc.crl
http://proxy.fineid.fi/arl/cacerta.crl
```

MIME object types:

Root and CA-certificates:

```
application/x-x509-ca-cert
```

Authority Revocation Lists (ARL):

```
application/pkix-crl
```

Certificate Revocation Lists (CRL):

```
application/pkix-crl
```

Examples:

The following HTTP URL retrieves the VRK Gov. Root CA certificate:

```
http://proxy.fineid.fi/ca/vrkrootc.crt
```

The following HTTP URL retrieves the CA certificate of the CA which commonName is VRK Gov. CA for Citizen Qualified Certificates:

```
http://proxy.fineid.fi/ca/vrkcqc.crt
```

The following HTTP URL retrieves the CRL of the CA which commonName is VRK CA for Service Providers:

```
http://proxy.fineid.fi/crl/vrkspc.crl
```

The following HTTP URL retrieves the ARL issued by VRK Gov. Root CA:

```
http://proxy.fineid.fi/arl/vrkroota.crl
```

6. Directory Schema

6.1 General

The FINEID directory specification is based on international standards (ISO/IEC) and common specifications (IETF RFCs). Standard object classes and attributes are used if available. Additional and partly duplicate object classes have been added to the schema and into directory entries to support the largest possible number of client software packages.

Some FINEID specific object classes and attribute types have been defined in addition to the standardized ones. UTF-8 character representation is used in directory entries to support the full latin-1 (ISO 8859-1) character set in string type attributes and entry names according to LDAPv3.

6.2 FINEID Attribute types

6.2.1 fineidCertKeyUsageString

The fineidCertKeyUsageString attribute type (attributeType.3) is a string representation of the intended usage(s) of the certified key.

The key usage is stored in a (multivalue) fineidCertKeyUsageString attribute type. Each attribute value presents one possible usage for the key.

```
fineidCertKeyUsageString   ATTRIBUTE ::=   {
    WITH SYNTAX             PrintableString
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                      { fineidAttributeType 3 }
}
```

The allowed key usage values are presented in IETF RFC 3280 “Internet X.509 Public Key Infrastructure, Certificate and CRL Profile” as follows:

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

Example of usage:

In FINEID Electronic Identity application the smart card (token) contains two key pairs.

The first certified key that is used for digitalSignature, keyEncipherment and dataEncipherment has an associated directory entry where the fineidCertKeyUsageString attribute contains values ‘0’, ‘2’ and ‘3.

The second certified key that is used for nonRepudiation has a directory entry where the fineidCertKeyUsageString attribute contains value ‘1’.

(Both certificates contain the same unique identifier of the subject, the FINUID. This identifier binds the keys to the same subject.)

6.2.2 certificateSerialNumber

The certificateSerialNumber attribute type (attributeType.4) is a string representation of the certificate serial number.

The certificate serial number, which uniquely identifies the certificate (within the CA domain), is stored in a single value attribute type certificateSerialNumber.

```
certificateSerialNumber  ATTRIBUTE ::= {
    WITH SYNTAX             Printable String (SIZE (1..64))
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                      { fineidAttributeType 4 }
}
```

6.2.3 smartCardSerialNumber

The smartCardSerialNumber attribute type (attributeType.5) is a string representation of the smart card serial number.

The smart card serial number, which uniquely identifies the smart card (within the CA domain), is stored in a single value attribute type smartCardSerialNumber.

```
smartCardSerialNumber  ATTRIBUTE ::= {
    WITH SYNTAX             Printable String (SIZE (1..64))
    EQUALITY MATCHING RULE        caseIgnoreMatch
    SUBSTRINGS MATCHING RULE      caseIgnoreSubstringsMatch
    ID                      { fineidAttributeType 5 }
}
```

Note:

This attribute is reserved for future use (RFU).

6.2.4 serialNumber

The serialNumber attribute type is used to unambiguously identify the FINEID certificate holders. The attribute value is stored as a case insensitive string with syntax PrintableString.

Example of usage:

In the case of Citizen Certificates the serialNumber attribute value contains the FINUID of the person in question. The FINUID string consists of 9 characters where the last character is a checksum calculated from the first 8 digits.

For example:

```
12345678N
```

6.2.5 publicKeySHA1Hash

The publicKeySHA1Hash attribute type (attributeType.6) contains the SHA-1 hash calculated from modulus of certificate holders public RSA key in hexadecimal form. The attribute is stored as a case insensitive printable string. String length is 40 characters.

For example:

```
6CDDCFF38F10CFE8CC9DE28790063E682A12FD6F
```

6.3 FINEID Object Classes

Two object classes have been defined for storage of CA and user certificate information. The objective has been to define multipurpose object classes for all types of certificates (for citizens, organizational users, servers, etc.) and to allow storage of additional information of the subject. (Although additional information may not be available in the actual certificate content, it is possible to add information to the directory entry from other data sources.)

Structure rules are defined in paragraph 6.4.

6.3.1 fineidCertificationAuthority

The fineidCertificationAuthority is an object class representing one logical Certification Authority (CA). The CA entry contains the CA certificate and the CRL information. It is possible to store additional information about the CA like cross certification information, contact information etc.

In the object class definition below the attribute types are presented in LDAPv3 short form and old long LDAPv2 attribute names are presented in parentheses.

```
fineidCertificationAuthority  OBJECT-CLASS ::= {
    SUBCLASS OF top
    MUST CONTAIN      {cn | -- name of the CA (commonName)
                      cACertificate}
    MAY CONTAIN       {certificateRevocationList | --  v2
                      authorityRevocationList |
                      deltaRevocationList |
                      crossCertificatePair |
                      dMDName | -- of the CA
                      c | -- of the CA (countryName)
                      o | -- of the CA (organizationName)
                      ou | -- of the CA (organizationalUnitName)
                      l | -- (localityName)
                      st | -- (stateOrProvinceName)
                      street | -- (streetAddress)
                      postalAddress |
                      postalCode |
                      postOfficeBox |
                      mail | -- of the CA (rfc822Mailbox)
                      telephoneNumber |
                      facsimileTelephoneNumber |
                      labeledURI | -– pointer to information,
                                   certificate policy etc.
                      description |
                      seeAlso }
    ID                {fineidObjectClass 1}
}
```

Note:

This object will be updated each time a new CRL or ARL is issued, overwriting the certificateRevocationList or authorityRevocationList attribute.

CRL Distribution Points are presented in certificates as LDAP URLs which are used to retrieve the certificateRevocationList attribute of the CA entry.

ARL Distribution Points are presented in CA certificates as HTTP URLs which are used to retrieve the authorityRevocationList attribute.

The fineidCertificationAuthority object class is for backward compatibility. Usage of the object class certificationAuthority instead is recommended.

6.3.2 fineidUserCertificate

The fineidUserCertificate is an object class representing an end user certificate issued by a logical CA. End user can be any subject (individual person, role, server etc.) that can be certified according to the Certificate Policy (CP) of the CA.

In the object class definition below the attribute types are presented in LDAPv3 short form and old long LDAPv2 attribute names are presented in parentheses.

```
fineidUserCertificate  OBJECT-CLASS  ::=         {
    SUBCLASS OF top
    MUST CONTAIN      {certificateSerialNumber | -- certificate
                          serial number
                       userCertificate }
    MAY CONTAIN {c | -- (countryName) of the subject
                dMDName | --
                sn | -- (surname) of the subject
                givenName | -- of the subject
                initials | -- of the subject
                cn | -- (commonName) of the subject
                serialNumber | -- unique identifier of the subject
                smartcardSerialNumber |
                title |
                o | -- (organizationName)
                ou | -- (organizationalUnitName)
                l | -- (localityName)
                st | -- (stateOrProvinceName)
                street | -- (streetAddress)
                postalAddress |
                postalCode |
                postOfficeBox |
                mail | -- (rfc822Mailbox)
                telephoneNumber |
                mobile | -- (mobileTelephoneNumber)
                facsimileTelephoneNumber |
                labeledURI | -- CRL distribution point as URI
                fineidCertKeyUsageString |
                description |
                seeAlso |
                publicKeySHA1Hash }
    ID          { fineidObjectClass 2 }
}
```

Note:

The FINEID Citizen Electronic Identity application contains two key pairs (and therefore two certificates) on a smart card. These two certificates are stored in the directory and named with unique certificateSerialNumber. The connecting link between the certificates is FINUID, which is stored in the serialNumber attribute.

6.4 FINEID Naming rules

6.4.1 FINEID Certification Authority naming rule

The Issuer Name stored in the certificate is **not** the same as the distinguished name (DN) of the directory entry that represents the Issuer in question. In FINEID

application the dmdName is added into each Issuer name in order to form the distinguished name of this Issuer.

For example:

Issuer Name (in certificate):

```
c = FI
o = Vaestorekisterikeskus CA
ou = Valtion kansalaisvarmenteet
cn = VRK Gov. CA for Citizen Qualified Certificates
```

Distinguished Name (in directory):

```
c = FI
dmdName = FINEID
o = Vaestorekisterikeskus CA
ou = Valtion kansalaisvarmenteet
cn = VRK Gov. CA for Citizen Qualified Certificates
```

6.4.2 FINEID User Certificate naming rule

The user certificate naming rule specifies how directory entries of the fineidUserCertificate object class are named.

The certificateSerialNumber attribute value contains the certificate serial number in decimal form. This forms the RDN (Relative Distinguished Name) of the directory entry.

For example:

```
certificateSerialNumber=”12345678”
```

Note:

It should be noted that in general the naming of a certificate holder in the certificate (Subject Name) does **not** (necessarily) reflect the naming of the corresponding certificate entry in directory.

For example:

Issuer Name (in certificate):

```
c = FI
o = Vaestorekisterikeskus CA
ou = Valtion kansalaisvarmenteet
cn = VRK Gov. CA for Citizen Qualified Certificates
```

Subject Name (in certificate):

```
c = FI
cn = Doe John 99999534D ((sn = Doe) + (givenName = John) +
(serialNumber = 99999534D))
certificateSerialNumber = hex 01:E2:40 (dec 123456)
```

Distinguished Name (in directory):

```
c = FI
dmdName = FINEID
o = Vaestorekisterikeskus CA
ou = Valtion kansalaisvarmenteet
cn = VRK Gov. CA for Citizen Qualified Certificates
certificateSerialNumber = 123456
```

Note:

In FINEID Electronic Identity application the same Subject will have at least two certificates in the directory e.g. two directory entries named with certificate serial numbers represent certificates issued to one Subject (smart card holder). It is also possible for the Subject to have more than one valid certificate pair in the directory issued by one or more CAs (organizations).

6.4.3 FINEID Structure Rules

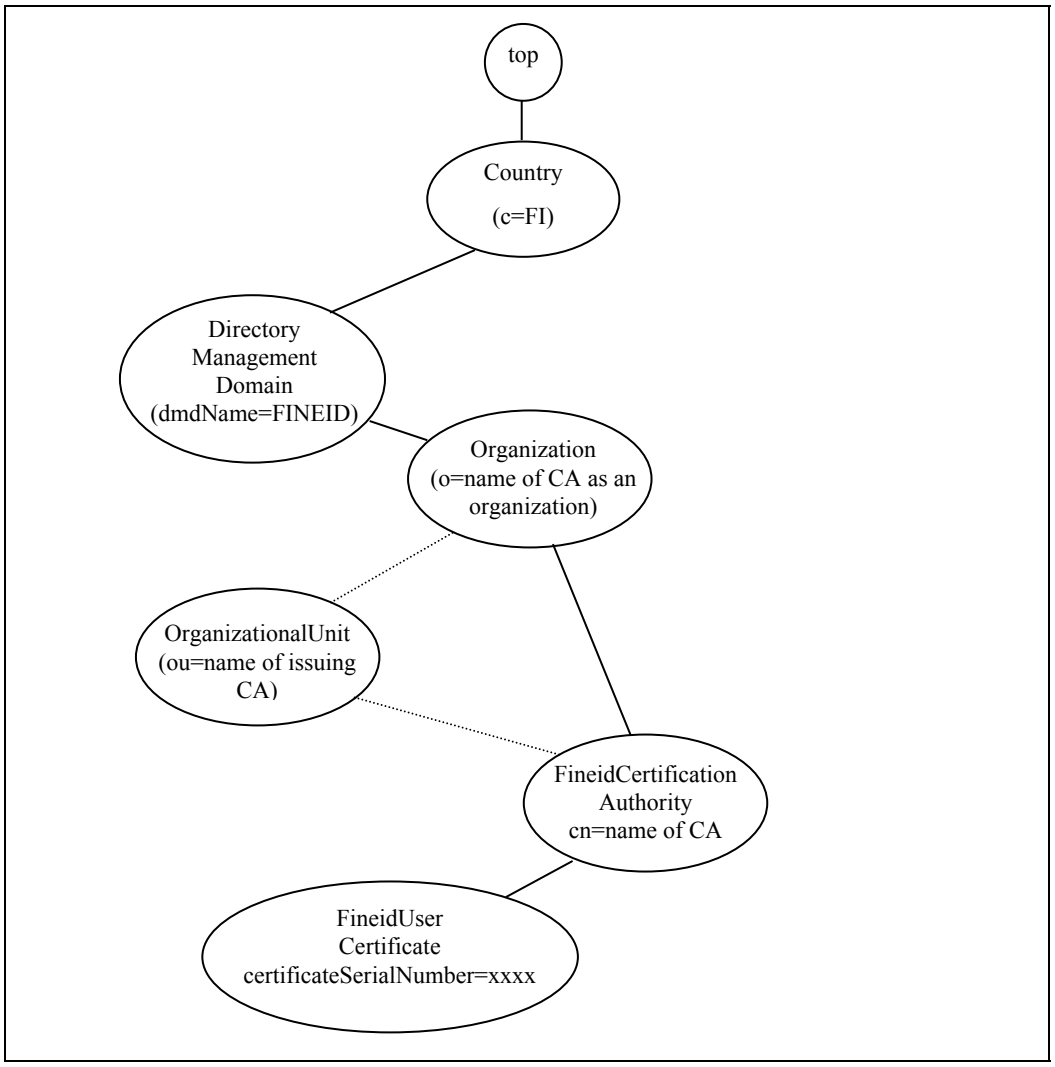
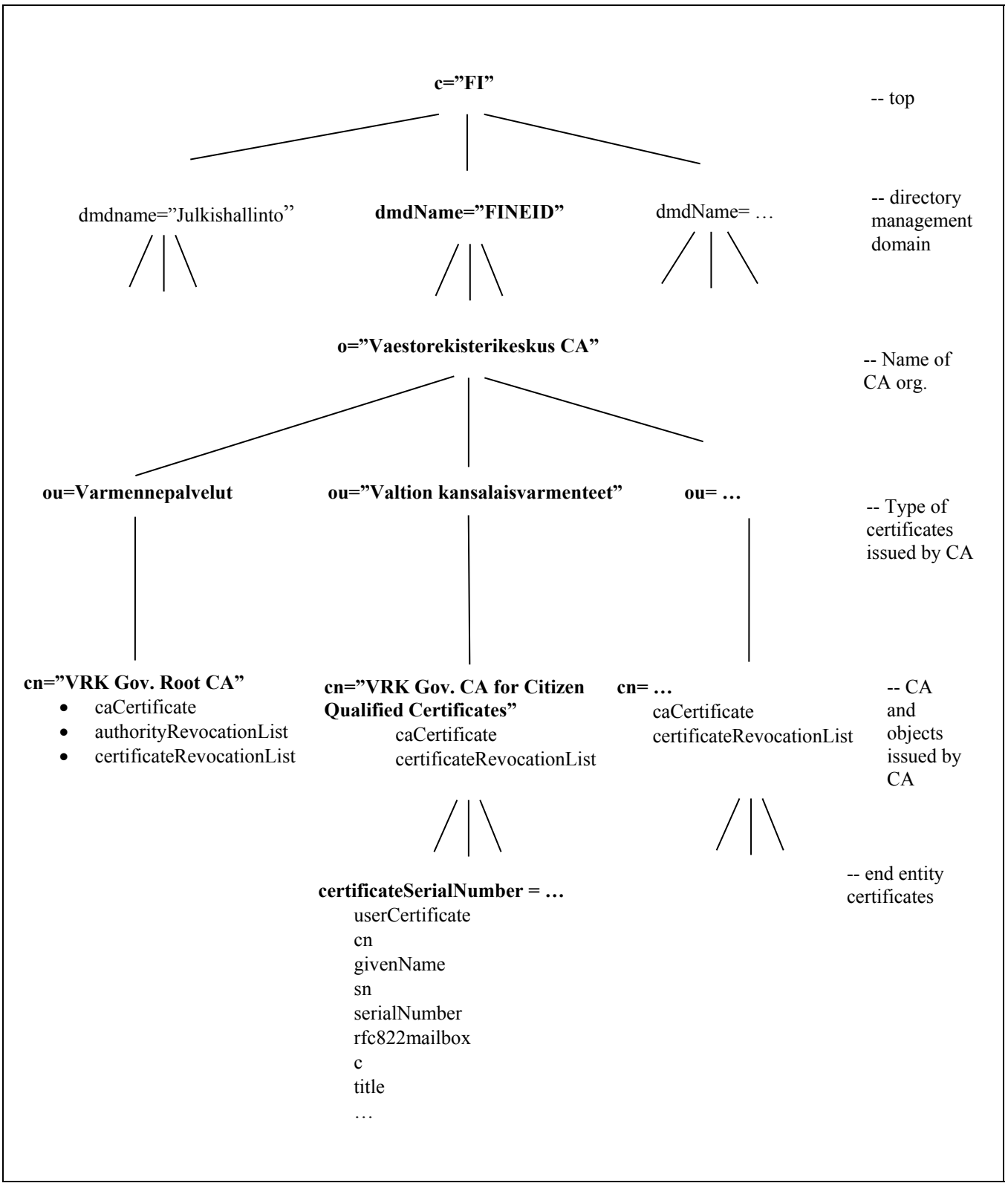


Figure 1: Structure rules for FINEID object classes.

OrganizationalUnit is used when multiple intermediate (sub) CA's are under one Root CA.

The following figure illustrates the FINEID Directory Information Tree:



The directory contains multiple directory management domains, organizations and certification authorities.

6.5 Email addresses in directory entries

The FINEID application always contains two different certificates and key pairs. Both certificates (one for authentication & encryption and one for non-repudiation digital signatures) are published into directory. Both certificates might contain the email address of the subject. However, the email address from non-repudiation certificate is not stored in the associated directory entry. This makes it easier for end users to select the correct certificate when searching for the recipient's certificate for email encryption (authentication & encryption certificate) and usage of fineidCertKeyUsageString is not possible or practical.

Note:

Email address is **not** a mandatory attribute in a certificate. If there is no email address in the certificate the corresponding directory entry does not contain an email address (mail, rfc822Mailbox attribute) at all.

6.6 FINEID Object Identifiers

The following Object Identifiers (OIDs) are specified for the FINEID application:

```
fineid                OBJECT IDENTIFIER  ::= {iso(1) memberBody(2)
                                               Finland(246)
                                               Väestörekisterikeskus(517)
                                               FINEID(1)}
fineidDirectory       OBJECT IDENTIFIER  ::= {fineid 5}
fineidAttributeType   OBJECT IDENTIFIER  ::= {fineidDirectory 4}
fineidObjectClass     OBJECT IDENTIFIER  ::= {fineidDirectory 6}
```

This table summaries the following Object IDs defined in this schema:

Object name	Object ID
Object classes	
fineidCertificationAuthority	fineidObjectClass.1
fineidUserCertificate	fineidObjectClass.2
Attribute types	
fineidCertKeyUsageString	fineidAttributeType.3
certificateSerialNumber	fineidAttributeType.4
smartCardSerialNumber	fineidAttributeType.5 (RFU)
publicKeySHA1Hash	fineidAttributeType.6

Appendix A. Attribute Types

Summary of Attribute Type definitions for the FINEID Directory Specification.

FINEID Attribute Types

fineidCertKeyUsageString

```
( 1.2.246.517.1.5.4.3 NAME 'fineidCertKeyUsageString'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{16}
)
```

certificateSerialNumber

```
( 1.2.246.517.1.5.4.4 NAME 'certificateSerialNumber'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64}
SINGLE-VALUE
)
```

smartCardSerialNumber

```
( 1.2.246.517.1.5.4.5 NAME 'smartCardSerialNumber'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64}
SINGLE-VALUE
)
```

publicKeySHA1Hash

```
( 1.2.246.517.1.5.4.6 NAME 'publicKeySHA1Hash'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{40}
SINGLE-VALUE
)
```

Attribute Types from RFC 2256

The following attribute types are extracted from IETF RFC 2256 “A Summary of the X.500(96) User Schema for use with LDAPv3”.

dmdName

```
The value of this attribute specifies a directory management domain
(DMD), the administrative authority which operates the directory
server.

    ( 2.5.4.54 NAME 'dmdName' SUP name )
```

Attribute Types from RFC 4512

The following attribute types are extracted from IETF RFC 4512 “Lightweight Directory Access Protocol (LDAP): Directory Information Models”.

objectClass

```
      ( 2.5.4.0 NAME 'objectClass'
        EQUALITY objectIdentifierMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
```

createTimestamp

```
   This attribute appears in entries that were added using the
protocol (e.g., using the Add operation).  The value is the time the
entry was added.

      ( 2.5.18.1 NAME 'createTimestamp'
        EQUALITY generalizedTimeMatch
        ORDERING generalizedTimeOrderingMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
        SINGLE-VALUE NO-USER-MODIFICATION
        USAGE directoryOperation )
```

modifyTimestamp

```
   This attribute appears in entries that have been modified using
the protocol (e.g., using the Modify operation).  The value is the
time the entry was last modified.

      ( 2.5.18.2 NAME 'modifyTimestamp'
        EQUALITY generalizedTimeMatch
        ORDERING generalizedTimeOrderingMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
        SINGLE-VALUE NO-USER-MODIFICATION
        USAGE directoryOperation )
```

Attribute Types from RFC 4519

The following attribute types are extracted from IETF RFC 4519 “Lightweight Directory Access Protocol (LDAP): Schema for User Applications”.

cn

```
    The 'cn' ('commonName' in X.500) attribute type contains names
of an object. Each name is one value of this multi-valued attribute.
If the object corresponds to a person, it is typically the person's
full name.

      ( 2.5.4.3 NAME 'cn' SUP name )
```

sn

```
    The 'sn' ('surname' in X.500) attribute type contains name
strings for the family names of a person. Each string is one value
of this multi-valued attribute.

      ( 2.5.4.4 NAME 'sn' SUP name )
```

serialNumber

```
   The 'serialNumber' attribute type contains the serial numbers of
devices.  In FINEID context the serialNumber attribute contains
unique identifier (FINUID) of the subject. Each serial number is one
value of this multi-valued attribute.

      ( 2.5.4.5 NAME 'serialNumber'
         EQUALITY caseIgnoreMatch
         SUBSTR caseIgnoreSubstringsMatch
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )
```

c

```
   The 'c' ('countryName' in X.500) attribute type contains a two-
letter ISO 3166 [ISO3166] country code.

      ( 2.5.4.6 NAME 'c'
         SUP name
         SYNTAX 1.3.6.1.4.1.1466.115.121.1.11
         SINGLE-VALUE )
```

l

The 'l' ('localityName' in X.500) attribute type contains names of a locality or place, such as a city, county, or other geographic region. Each name is one value of this multi-valued attribute.

```
( 2.5.4.7 NAME 'l' SUP name )
```

st

The 'st' ('stateOrProvinceName' in X.500) attribute type contains the full names of states or provinces. Each name is one value of this multi-valued attribute.

```
( 2.5.4.8 NAME 'st' SUP name )
```

street

The 'street' ('streetAddress' in X.500) attribute type contains site information from a postal address (i.e., the street name, place, avenue, and the house number). Each street is one value of this multi-valued attribute.

```
( 2.5.4.9 NAME 'street'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

o

The 'o' ('organizationName' in X.500) attribute type contains the names of an organization. Each name is one value of this multi-valued attribute.

```
( 2.5.4.10 NAME 'o' SUP name )
```

ou

The 'ou' ('organizationalUnitName' in X.500) attribute type contains the names of an organizational unit. Each name is one value of this multi-valued attribute.

```
( 2.5.4.11 NAME 'ou' SUP name )
```

title

The 'title' attribute type contains the title of a person in their organizational context. Each title is one value of this multi-valued attribute.

```
( 2.5.4.12 NAME 'title' SUP name )
```

description

The 'description' attribute type contains human-readable descriptive phrases about the object. Each description is one value of this multi-valued attribute.

```
( 2.5.4.13 NAME 'description'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

postalAddress

The 'postalAddress' attribute type contains addresses used by a Postal Service to perform services for the object. Each address is one value of this multi-valued attribute.

```
( 2.5.4.16 NAME 'postalAddress'
      EQUALITY caseIgnoreListMatch
      SUBSTR caseIgnoreListSubstringsMatch
      SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )
```

postalCode

The 'postalCode' attribute type contains codes used by a Postal Service to identify postal service zones. Each code is one value of this multi-valued attribute.

```
( 2.5.4.17 NAME 'postalCode'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

postOfficeBox

The 'postOfficeBox' attribute type contains postal box identifiers that a Postal Service uses when a customer arranges to receive mail at a box on the premises of the Postal Service. Each postal box identifier is a single value of this multi-valued attribute.

```
( 2.5.4.18 NAME 'postOfficeBox'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

seeAlso

The 'seeAlso' attribute type contains the distinguished names of objects that are related to the subject object. Each related object name is one value of this multi-valued attribute.

```
( 2.5.4.34 NAME 'seeAlso' SUP distinguishedName )
```

name

The 'name' attribute type is the attribute supertype from which user attribute types with the name syntax inherit. Such attribute types are typically used for naming. The attribute type is multi-valued.

It is unlikely that values of this type itself will occur in an entry. LDAP server implementations that do not support attribute subtyping need not recognize this attribute in requests. Client implementations MUST NOT assume that LDAP servers are capable of performing attribute subtyping.

```
( 2.5.4.41 NAME 'name'
   EQUALITY caseIgnoreMatch
   SUBSTR caseIgnoreSubstringsMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )
```

givenName

The 'givenName' attribute type contains name strings that are the part of a person's name that is not their surname. Each string is one value of this multi-valued attribute.

```
( 2.5.4.42 NAME 'givenName' SUP name )
```

initials

The 'initials' attribute type contains strings of initials of some or all of an individual's names, except the surname(s). Each string is one value of this multi-valued attribute.

```
( 2.5.4.43 NAME 'initials' SUP name )
```

distinguishedName

The 'distinguishedName' attribute type is not used as the name of the object itself, but it is instead a base type from which some user attribute types with a DN syntax can inherit.

It is unlikely that values of this type itself will occur in an entry. LDAP server implementations that do not support attribute subtyping need not recognize this attribute in requests. Client

implementations MUST NOT assume that LDAP servers are capable of performing attribute subtyping.

```
( 2.5.4.49 NAME 'distinguishedName'
   EQUALITY distinguishedNameMatch
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
```

Attribute Types from RFC 4523

The following attribute types are extracted from IETF RFC 4523 “Lightweight Directory Access Protocol (LDAP) Schema Definitions for X.509 Certificates”.

userCertificate

The userCertificate attribute holds the X.509 certificates issued to the user by one or more certificate authorities.

```
( 2.5.4.36 NAME 'userCertificate'
     DESC 'X.509 user certificate'
     EQUALITY certificateExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )
```

As required by this attribute type's syntax, values of this attribute are requested and transferred using the attribute description "userCertificate;binary".

cACertificate

The cACertificate attribute holds the X.509 certificates issued to the certificate authority (CA).

```
( 2.5.4.37 NAME 'cACertificate'
     DESC 'X.509 CA certificate'
     EQUALITY certificateExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )
```

As required by this attribute type's syntax, values of this attribute are requested and transferred using the attribute description "cACertificate;binary".

authorityRevocationList

The authorityRevocationList attribute holds certificate lists.

```
( 2.5.4.38 NAME 'authorityRevocationList'
     DESC 'X.509 authority revocation list'
     EQUALITY certificateListExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )
```

As required by this attribute type's syntax, values of this attribute are requested and transferred using the attribute description "authorityRevocationList;binary".

certificateRevocationList

The certificateRevocationList attribute holds certificate lists.

```
( 2.5.4.39 NAME 'certificateRevocationList'
     DESC 'X.509 certificate revocation list'
     EQUALITY certificateListExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )
```

As required by this attribute type's syntax, values of this attribute are requested and transferred using the attribute description "certificateRevocationList;binary".

crossCertificatePair

The crossCertificatePair attribute holds an X.509 certificate pair.

```
( 2.5.4.40 NAME 'crossCertificatePair'
     DESC 'X.509 cross certificate pair'
     EQUALITY certificatePairExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.10 )
```

As required by this attribute type's syntax, values of this attribute are requested and transferred using the attribute description "crossCertificatePair;binary".

deltaRevocationList

The deltaRevocationList attribute holds certificate lists.

```
( 2.5.4.53 NAME 'deltaRevocationList'
     DESC 'X.509 delta revocation list'
     EQUALITY certificateListExactMatch
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )
```

As required by this attribute type's syntax, values of this attribute MUST be requested and transferred using the attribute description "deltaRevocationList;binary".

Attribute Types from RFC 4524

The following attribute types are extracted from IETF RFC 4524 “COSINE LDAP/X.500 Schema”.

mail

The 'mail' (rfc822mailbox) attribute type holds Internet mail addresses in Mailbox [RFC2821] form (e.g., user@example.com).

```
( 0.9.2342.19200300.100.1.3 NAME 'mail'
  EQUALITY caseIgnoreIA5Match
  SUBSTR caseIgnoreIA5SubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )
```

The IA5String (1.3.6.1.4.1.1466.115.121.1.26) syntax and the 'caseIgnoreIA5Match' and 'caseIgnoreIA5SubstringsMatch' rules are described in [RFC4517].

Note that the directory will not ensure that values of this attribute conform to the <Mailbox> production [RFC2821]. It is the application's responsibility to ensure that domains it stores in this attribute are appropriately represented. Additionally, the directory will compare values per the matching rules named in the above attribute type description. As these rules differ from rules that normally apply to <Mailbox> comparisons, operational issues may arise. For example, the assertion (mail=joe@example.com) will match "JOE@example.com" even though the <local-parts> differ. Also, where a user has two <Mailbox>es whose addresses differ only by case of the <local-part>, both cannot be listed as values of the user's mail attribute (as they are considered equal by the 'caseIgnoreIA5Match' rule).

Also note that applications supporting internationalized domain names SHALL use the ToASCII method [RFC3490] to produce <sub-domain> components of the <Mailbox> production.

mobile

The 'mobile' (mobileTelephoneNumber) attribute specifies mobile telephone numbers (e.g., "+1 775 555 6789") associated with a person (or entity).

```
( 0.9.2342.19200300.100.1.41 NAME 'mobile'
  EQUALITY telephoneNumberMatch
  SUBSTR telephoneNumberSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.50 )
```

The telephoneNumber (1.3.6.1.4.1.1466.115.121.1.50) syntax and the 'telephoneNumberMatch' and 'telephoneNumberSubstringsMatch' rules are described in [RFC4517].

Appendix B. Object Classes

FINEID Object Classes

Summary of Object Class definitions for the FINEID Directory Specification.

fineidCertificationAuthority

```
( 1.2.246.517.1.5.6.1 NAME 'fineidCertificationAuthority'
SUP top
STRUCTURAL
MUST ( cn $ cACertificate )
MAY ( certificateRevocationList $ postalAddress $ postalCode
$ postOfficeBox $ mail $ telephoneNumber $
facsimileTelephoneNumber $ labeledURI $ c $ o $ ou $ l $
st $ street $ description $ seeAlso $ dmdName $
authorityRevocationList $ deltaRevocationList $
crossCertificatePair )
)
```

fineidUserCertificate

```
( 1.2.246.517.1.5.6.2 NAME 'fineidUserCertificate'
SUP top
STRUCTURAL
MUST ( userCertificate $ certificateSerialNumber )
MAY ( mail $ mobile $ fineidCertKeyUsageString $
labeledURI $ cn $ sn $ c $ l $ st $ street $ o $ ou $ title $
description $ postalAddress $ postalCode $
postOfficeBox $ telephoneNumber $ facsimileTelephoneNumber $
seeAlso $ givenName $ initials $ dmdName $
serialNumber $ smartCardSerialNumber $
publicKeySHA1Hash )
)
```

Object Classes from RFC 2256

The following object classes are extracted from IETF RFC 2256 “A Summary of the X.500(96) User Schema for use with LDAPv3”.

dmd

```
( 2.5.6.20 NAME 'dmd' SUP top STRUCTURAL MUST ( dmdName )
  MAY ( userPassword $ searchGuide $ seeAlso $
  businessCategory $ x121Address $ registeredAddress $
  destinationIndicator $ preferredDeliveryMethod $
  telexNumber $ teletexTerminalIdentifier $ telephoneNumber $
  internationaliSDNNumber $ facsimileTelephoneNumber $
  street $ postOfficeBox $ postalCode $ postalAddress $
  physicalDeliveryOfficeName $ st $ l $ description ) )
```

Object Classes from RFC 4512

The following object classes are extracted from IETF RFC 4512 “Lightweight Directory Access Protocol (LDAP): Directory Information Models”.

subschema

```
   Subschema is held in (sub)entries belonging to the subschema
auxiliary object class.

( 2.5.20.1 NAME 'subschema' AUXILIARY
  MAY ( dITStructureRules $ nameForms $ ditContentRules $
    objectClasses $ attributeTypes $ matchingRules $
    matchingRuleUse ) )
```

top

```
( 2.5.6.0 NAME 'top' ABSTRACT MUST objectClass )
```

Object Classes from RFC 4519

The following object classes are extracted from IETF RFC 4519 “Lightweight Directory Access Protocol (LDAP): Schema for User Applications”.

country

```
   The 'country' object class definition is the basis of an entry
that represents a country.
```

```
( 2.5.6.2 NAME 'country'
   SUP top
   STRUCTURAL
   MUST c
   MAY ( searchGuide $
         description ) )
```

organization

The 'organization' object class is the basis of an entry that represents a structured group of people.

```
( 2.5.6.4 NAME 'organization'
   SUP top
   STRUCTURAL
   MUST o
   MAY ( userPassword $ searchGuide $ seeAlso $
         businessCategory $ x121Address $ registeredAddress $
         destinationIndicator $ preferredDeliveryMethod $
         telexNumber $ teletexTerminalIdentifier $
         telephoneNumber $ internationalISDNNumber $
         facsimileTelephoneNumber $ street $ postOfficeBox $
         postalCode $ postalAddress $
         physicalDeliveryOfficeName $ st $ l $ description ) )
```

organizationalUnit

The 'organizationalUnit' object class is the basis of an entry that represents a piece of an organization.

```
( 2.5.6.5 NAME 'organizationalUnit'
   SUP top
   STRUCTURAL
   MUST ou
   MAY ( businessCategory $ description $
         destinationIndicator $ facsimileTelephoneNumber $
         internationalISDNNumber $ l $
         physicalDeliveryOfficeName $ postalAddress $
         postalCode $ postOfficeBox $
         preferredDeliveryMethod $ registeredAddress $
         searchGuide $ seeAlso $ st $ street $
         telephoneNumber $ teletexTerminalIdentifier $
         telexNumber $ userPassword $ x121Address ) )
```

person

The 'person' object class is the basis of an entry that represents a human being.

```
( 2.5.6.6 NAME 'person'
   SUP top
   STRUCTURAL
   MUST ( sn $
          cn )
   MAY ( userPassword $
          telephoneNumber $
          seeAlso $ description ) )
```

organizationalPerson

The 'organizationalPerson' object class is the basis of an entry that represents a person in relation to an organization.

```
( 2.5.6.7 NAME 'organizationalPerson'
   SUP person
   STRUCTURAL
   MAY ( title $ x121Address $ registeredAddress $
          destinationIndicator $ preferredDeliveryMethod $
          telexNumber $ teletexTerminalIdentifier $
          telephoneNumber $ internationalISDNNumber $
          facsimileTelephoneNumber $ street $ postOfficeBox $
          postalCode $ postalAddress $
          physicalDeliveryOfficeName $ ou $ st $ l ) )
```

Object Classes from RFC 4523

The following object classes are extracted from IETF RFC 4523 “Lightweight Directory Access Protocol (LDAP): Schema Definitions for X.509 Certificates”.

certificationAuthority

This object class is used to augment entries for objects that act as certificate authorities. This object class is deprecated in favor of pkiCA.

```
( 2.5.6.16 NAME 'certificationAuthority'
     DESC 'X.509 certificate authority'
     SUP top AUXILIARY
     MUST ( authorityRevocationList $
          certificateRevocationList $ cACertificate )
     MAY crossCertificatePair )
```

cRLDistributionPoint

This class is used to represent objects that act as CRL distribution points.

```
      ( 2.5.6.19 NAME 'cRLDistributionPoint'
           DESC 'X.509 CRL distribution point'
           SUP top STRUCTURAL
           MUST cn
           MAY ( certificateRevocationList $
                authorityRevocationList $ deltaRevocationList ) )
```

pkiUser

This object class is used in augment entries for objects that may be subject to certificates.

```
      ( 2.5.6.21 NAME 'pkiUser'
           DESC 'X.509 PKI User'
           SUP top AUXILIARY
           MAY userCertificate )
```

pkiCA

This object class is used to augment entries for objects that act as certificate authorities.

```
      ( 2.5.6.22 NAME 'pkiCA'
           DESC 'X.509 PKI Certificate Authority'
           SUP top AUXILIARY
           MAY ( cACertificate $ certificateRevocationList $
                authorityRevocationList $ crossCertificatePair ) )
```

Appendix C. Coding example of publicKeySHA1Hash

This example describes the scheme which is used to calculate the FINEID publicKeySHA1Hash attribute in directory entries.

```
Excerpt from certificate (1024 bit modulus):

SEQUENCE {
   SEQUENCE {
      OBJECT_IDENTIFIER { "1.2.840.113549.1.1.1" },  -- rsaEncryption
      NULL { "NULL" }
   },
   BIT_STRING [ "PRIMITIVE" ] {
      #00,
      SEQUENCE {
         INTEGER {
            #00EAC4081AD509AC14AE8648A2A876E840F5B874F6040B2DB98F46B077
            26C6F2FBEB4AE237273CD7A60AE838857243CC69A1CDAF64395272D7B747
            1D60AF52F67F4DA6CF493082C7D0FE39B4DD17E1F1257FCEF9D3E729380B
            CD977F2AD581F7875B253564F52292A70A41D0DB81DAEEF3ECC1DADB7857
            25A4ED87964608E2AB09
         },
         INTEGER { 65537 } -- exponent
      }
   }
},

Modulus (1024 bit):

      00EAC4081AD509AC14AE8648A2A876E840F5B874F6040B2DB98F46B077
      26C6F2FBEB4AE237273CD7A60AE838857243CC69A1CDAF64395272D7B747
      1D60AF52F67F4DA6CF493082C7D0FE39B4DD17E1F1257FCEF9D3E729380B
      CD977F2AD581F7875B253564F52292A70A41D0DB81DAEEF3ECC1DADB7857
      25A4ED87964608E2AB09

SHA-1 hash of modulus (=publicKeySHA1Hash):

      6CDDCFF38F10CFE8CC9DE28790063E682A12FD6F
```

publicKeySHA1Hash attribute is stored in directory as case insensitive printableString (40 characters).

Note: Contents of publicKeySHA1Hash is different than subjectKeyIdentifier in certificates. For more information about certificate contents, see the FINEID S2 specification.