LDAP: String Representation of Distinguished Names
<draft-ietf-ldapbis-dn-13.txt>

Status of Memo

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

   This document is intended to be, after appropriate review and
   revision, submitted to the RFC Editor as a Standard Track document
   replacing RFC 2253. Distribution of this memo is unlimited.
   Technical discussion of this document will take place on the IETF LDAP
   Revision (LDAPBIS) Working Group mailing list
   <ietf-ldapbis@openldap.org>. Please send editorial comments directly
   to the document editor <Kurt@OpenLDAP.org>.

   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/ietf/1id-abstracts.txt>. The list of
   Internet-Draft Shadow Directories can be accessed at
   <http://www.ietf.org/shadow.html>.

INTERNET-DRAFT   draft-ietf-ldapbis-dn-13.txt   15 Febrary 2004

Abstract

   The X.500 Directory uses distinguished names (DNs) as primary keys to
   entries in the directory. This document defines the string
   representation used in the Lightweight Directory Access Protocol
   (LDAP) to transfer distinguished names. The string representation is
   designed to give a clean representation of commonly used distinguished
   names, while being able to represent any distinguished name.

1.   Background and Intended Usage

   In X.500-based directory systems [X.500], including those accessed
   using the Lightweight Directory Access Protocol (LDAP) [Roadmap],
   distinguished names (DNs) are used to unambiguously refer to directory
   entries [X.501][Models].

   The structure of a DN [X.501] is described in terms of ASN.1 [X.680].
   In the X.500 Directory Access Protocol [X.511] (and other ITU-defined
   directory protocols), DNs are encoded using the Basic Encoding Rules
   (BER) [X.690]. In LDAP, DNs are represented in the string form
   described in this document.

   It is important to have a common format to be able to unambiguously
   represent a distinguished name. The primary goal of this
   specification is ease of encoding and decoding. A secondary goal is
   to have names that are human readable. It is not expected that LDAP
   implementations with a human user interface would display these
   strings directly to the user, but would most likely be performing
   translations (such as expressing attribute type names in the local
   national language).

   This document defines the string representation of Distinguished Names
   used in LDAP [Protocol][Syntaxes]. Section 2 details the RECOMMENDED
   algorithm for converting a DN from its ASN.1 structured representation
   to a string. Section 3 details how to convert a DN from a string to a
   ASN.1 structured representation.

   While other documents may define other algorithms for converting a DN
   from its ASN.1 structured representation to a string, all algorithms
   MUST produce strings which adhere to the requirements of Section 3.

   This document does not define a canonical string representation for

DNs. Comparison of DNs for equality is to be performed in accordance
with the distinguishedNameMatch matching rule [Syntaxes].

This document is an integral part of the LDAP Technical Specification
[Roadmap]. This document obsoletes RFC 2253. Changes since RFC 2253

are summarized in Appendix B.

This specification assumes familiarity with X.500 [X.500] and the
concept of Distinguished Name [X.501][Models].

1.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in BCP 14 [RFC2119].

Character names in this document use the notation for code points and
names from the Unicode Standard [Unicode]. For example, the letter
"a" may be represented as either <U+0061> or <LATIN SMALL LETTER A>.

Note: a glossary of terms used in Unicode can be found in [Glossary].
Information on the Unicode character encoding model can be found in
[CharModel].

2. Converting DistinguishedName from ASN.1 to a String

X.501 [X.501] defines the ASN.1 [X.680] structure of distinguished
name. The following is a variant provided for discussion purposes.

    DistinguishedName ::= RDNSequence

    RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

    RelativeDistinguishedName ::= SET SIZE (1..MAX) OF
        AttributeTypeAndValue

    AttributeTypeAndValue ::= SEQUENCE {
        type AttributeType,
        value AttributeValue }

This section defines the RECOMMENDED algorithm for converting a
distinguished name from an ASN.1 structured representation to an UTF-8
[RFC3629] encoded Unicode [Unicode] character string representation.
Other documents may describe other algorithms for converting a
distinguished name to a string, but only strings which conform to the
grammar defined in Section 3 SHALL be produced by LDAP
implementations.

2.1. Converting the RDNSequence

If the RDNSequence is an empty sequence, the result is the empty or
zero length string.

Otherwise, the output consists of the string encodings of each
RelativeDistinguishedName in the RDNSequence (according to Section
2.2), starting with the last element of the sequence and moving
backwards toward the first.

The encodings of adjoining RelativeDistinguishedNames are separated by
a comma (',' U+002C) character.

2.2. Converting RelativeDistinguishedName

When converting from an ASN.1 RelativeDistinguishedName to a string,
the output consists of the string encodings of each
AttributeTypeAndValue (according to Section 2.3), in any order.

Where there is a multi-valued RDN, the outputs from adjoining
AttributeTypeAndValues are separated by a plus sign ('+' U+002B)
character.

2.3. Converting AttributeTypeAndValue

The AttributeTypeAndValue is encoded as the string representation of
the AttributeType, followed by an equals ('=' U+003D) character,
followed by the string representation of the AttributeValue. The
encoding of the AttributeValue is given in Section 2.4.

If the AttributeType is defined to have a short name and that short name is known to be registered [REGISTRY][BCP64bis] as identifying the AttributeType, that short name, a <descr>, is used. Otherwise the AttributeType is encoded as the dotted-decimal encoding, a <numericoid>, of its OBJECT IDENTIFIER. The <descr> and <numericoid> is defined in [Models].

Implementations are not expected to dynamically update their knowledge of registered short names. However, implementations SHOULD provide a mechanism to allow its knowledge of registered short names to be updated.

2.4. Converting an AttributeValue from ASN.1 to a String

If the AttributeType is of the dotted-decimal form, the AttributeValue is represented by an number sign ('#' U+0023) character followed by the hexadecimal encoding of each of the octets of the BER encoding of

the X.500 AttributeValue. This form is also used when the syntax of the AttributeValue does not have a LDAP-specific [Syntaxes, Section 3.1] string encoding defined for it or the LDAP-specific string encoding is not restricted to UTF-8 encoded Unicode characters. This form may also be used in other cases, such as when a reversible string representation is desired (see Section 5.2).

Otherwise, if the AttributeValue is of a syntax which has a LDAP-specific string encoding, the value is converted first to a UTF-8 encoded Unicode string according to its syntax specification (see [Syntaxes, Section 3.3] for examples). If that UTF-8 encoded Unicode string does not have any of the following characters which need escaping, then that string can be used as the string representation of the value.

    - a space (' ' U+0020) or number sign ('#' U+0023) occurring at
      the beginning of the string;

    - a space (' ' U+0020) character occurring at the end of the
      string;

    - one of the characters '"', '+', ',', ';', '<', '>', or '\'

(U+0022, U+002B, U+002C, U+003B, U+003C, U+003E, or U+005C
            respectively);

        - the null (U+0000) character.

    Other characters may be escaped.

    Each octet of the character to be escaped is replaced by a backslash
    and two hex digits, which form a single octet in the code of the
    character. Alternatively, if and only if the character to be escaped
    is one of

        ' ', '"', '#', '+', ',', ';', '<', '=', '>', or '\'
        (U+0020, U+0022, U+0023, U+002B, U+002C, U+003B,
         U+003C, U+003D, U+003E, U+005C respectively)

    it can be prefixed by a backslash ('\' U+0005C).

    Examples of the escaping mechanism are shown in Section 4.

3. Parsing a String back to a Distinguished Name

    The string representation of Distinguished Names is restricted to
    UTF-8 [RFC3629] encoded Unicode [Unicode] characters. The structure
    of this string representation is specified using the following

    Augmented BNF [RFC2234] grammar:

        distinguishedName = [relativeDistinguishedName
            *(COMMA relativeDistinguishedName)]
        relativeDistinguishedName = attributeTypeAndValue
            *(PLUS attributeTypeAndValue)
        attributeTypeAndValue = attributeType EQUALS attributeValue
        attributeType = descr / numericoid
        attributeValue = string / hexstring

        ; The following characters are to be escaped when they appear
        ; in the value to be encoded: ESC, one of <escaped>, leading
        ; SHARP or SPACE, trailing SPACE, and NULL.
        string   = [(leadchar / pair)
                    [*(stringchar / pair) (trailchar / pair)]]

```
     leadchar   = LUTF1 / UTFMB
     LUTF1      = %x01-1F / %x21 / %x24-2A / %x2D-3A /
                  %x3D / %x3F-5B / %x5D-7F

     trailchar  = TUTF1 / UTFMB
     TUTF1      = %x01-1F / %x21 / %x23-2A / %x2D-3A /
                  %x3D / %x3F-5B / %x5D-7F

     stringchar = SUTF1 / UTFMB
     SUTF1      = %x01-21 / %x23-2A / %x2D-3A /
                  %x3D / %x3F-5B / %x5D-7F

     pair       = ESC ( ESC / special / hexpair )
     special    = escaped / SPACE / SHARP / EQUALS
     escaped    = DQUOTE / PLUS / COMMA / SEMI / LANGLE / RANGLE
     hexstring  = SHARP 1*hexpair
     hexpair    = HEX HEX
```

   where the productions <descr>, <numericoid>, <COMMA>, <DQUOTE>,
   <EQUALS>, <ESC>, <HEX>, <LANGLE>, <NULL>, <PLUS>, <RANGLE>, <SEMI>,
   <SPACE>, <SHARP>, <UTFMB> are defined in [Models].

   Each <attributeType>, either a <descr> or a <numericoid>, refers to an
   attribute type of an attribute value assertion (AVA). The
   <attributeType> is followed by a <EQUALS> and an <attributeValue>.
   The <attributeValue> is either in <string> or <hexstring> form.

   If in <string> form, a LDAP string representation asserted value can
   be obtained by replacing (left-to-right, non-recursively) each <pair>
   appearing in the <string> as follows:
       replace <ESC><ESC> with <ESC>;
       replace <ESC><special> with <special>;

       replace <ESC><hexpair> with the octet indicated by the <hexpair>.

   If in <hexstring> form, a BER representation can be obtained from
   converting each <hexpair> of the <hexstring> to the octet indicated by
   the <hexpair>.

   One or more attribute values assertions, separated by <PLUS>, for a
   relative distinguished name.

Zero or more relative distinguished names, separated by <COMMA>, for a
distinguished name.

Implementations MUST recognize AttributeType name strings
(descriptors) listed in the following table, but MAY recognize other
name strings.

```
    String  X.500 AttributeType
    ------  -------------------------------------------
    CN      commonName (2.5.4.3)
    L       localityName (2.5.4.7)
    ST      stateOrProvinceName (2.5.4.8)
    O       organizationName (2.5.4.10)
    OU      organizationalUnitName (2.5.4.11)
    C       countryName (2.5.4.6)
    STREET  streetAddress (2.5.4.9)
    DC      domainComponent (0.9.2342.19200300.100.1.25)
    UID     userId (0.9.2342.19200300.100.1.1)
```

Implementations MAY recognize other DN string representations (such as
that described in RFC 1779). However, as there is no requirement that
alternative DN string representations to be recognized (and, if so,
how), implementations SHOULD only generate DN strings in accordance
with Section 2 of this document.

4. Examples

   This notation is designed to be convenient for common forms of name.
   This section gives a few examples of distinguished names written using
   this notation. First is a name containing three relative
   distinguished names (RDNs):

       UID=jsmith,DC=example,DC=net

   Here is an example name containing three RDNs, in which the first RDN
   is multi-valued:

       OU=Sales+CN=J. Smith,DC=example,DC=net

   This example shows the method of escaping of a comma in a common name:

```
        CN=John Smith\, III,DC=example,DC=net
```

An example name in which a value contains a carriage return character:

```
        CN=Before\0dAfter,DC=example,DC=net
```

An example name in which an RDN was of an unrecognized type. The
value is the BER encoding of an OCTET STRING containing two octets
0x48 and 0x69.

```
        1.3.6.1.4.1.1466.0=#04024869,DC=example,DC=com
```

Finally, an example of an RDN commonName value consisting of 5
letters:

```
        Unicode Character                   Code     UTF-8   Escaped
        ------------------------------      ------   ------  --------
        LATIN CAPITAL LETTER L              U+004C   0x4C    L
        LATIN SMALL LETTER U                U+0075   0x75    u
        LATIN SMALL LETTER C WITH CARON     U+010D   0xC48D  \C4\8D
        LATIN SMALL LETTER I                U+0069   0x69    i
        LATIN SMALL LETTER C WITH ACUTE     U+0107   0xC487  \C4\87
```

could be written in printable ASCII (useful for debugging purposes):

```
        CN=Lu\C4\8Di\C4\87
```

5. Security Considerations

   The following security considerations are specific to the handling of
   distinguished names. LDAP security considerations are discussed in
   [Protocol] and other documents comprising the LDAP Technical
   Specification [Roadmap].

5.1. Disclosure

   Distinguished Names typically consist of descriptive information about
   the entries they name, which can be people, organizations, devices or
   other real-world objects. This frequently includes some of the
   following kinds of information:

     - the common name of the object (i.e. a person's full name)
     - an email or TCP/IP address
     - its physical location (country, locality, city, street address)

   - organizational attributes (such as department name or affiliation)

   Most countries have privacy laws regarding the publication of
   information about people.

5.2. Use of Distinguished Names in Security Applications

   The transformations of an AttributeValue value from its X.501 form to
   an LDAP string representation are not always reversible back to the
   same BER (Basic Encoding Rules) or DER (Distinguished Encoding rules)
   form. An example of a situation which requires the DER form of a
   distinguished name is the verification of an X.509 certificate.

   For example, a distinguished name consisting of one RDN with one AVA,
   in which the type is commonName and the value is of the TeletexString
   choice with the letters 'Sam' would be represented in LDAP as the
   string <CN=Sam>. Another distinguished name in which the value is
   still 'Sam' but of the PrintableString choice would have the same
   representation <CN=Sam>.

   Applications which require the reconstruction of the DER form of the
   value SHOULD NOT use the string representation of attribute syntaxes
   when converting a distinguished name to the LDAP format. Instead,
   they SHOULD use the hexadecimal form prefixed by the number sign ('#')
   as described in the first paragraph of Section 2.3.

6. Acknowledgment

   This document is an update to RFC 2253, by Mark Wahl, Tim Howes, and
   Steve Kille. RFC 2253 was a product of the IETF ASID Working Group.

   This document is a product of the IETF LDAPBIS Working Group.

7. Document Editor's Address

   Kurt D. Zeilenga
   OpenLDAP Foundation
   <Kurt@OpenLDAP.org>

8. Normative References

   [X.501]   International Telecommunication Union -

Telecommunication Standardization Sector, "The Directory
-- Models," X.501(1993) (also ISO/IEC 9594-2:1994).

    [X.680]   International Telecommunication Union -
                  Telecommunication Standardization Sector, "Abstract
                  Syntax Notation One (ASN.1) - Specification of Basic
                  Notation", X.680(1997) (also ISO/IEC 8824-1:1998).

    [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
                  Requirement Levels", BCP 14 (also RFC 2119), March 1997.

    [RFC2234]   Crocker, D. and P. Overell, "Augmented BNF for Syntax
                  Specifications: ABNF", RFC 2234, November 1997.

    [RFC3329]   Yergeau, F., "UTF-8, a transformation format of ISO
                  10646", RFC 3329 (also STD 64), November 2003.

    [Unicode]   The Unicode Consortium, "The Unicode Standard, Version
                  3.2.0" is defined by "The Unicode Standard, Version 3.0"
                  (Reading, MA, Addison-Wesley, 2000. ISBN 0-201-61633-5),
                  as amended by the "Unicode Standard Annex #27: Unicode
                  3.1" (http://www.unicode.org/reports/tr27/) and by the
                  "Unicode Standard Annex #28: Unicode 3.2"
                  (http://www.unicode.org/reports/tr28/).

    [Models]   Zeilenga, K. (editor), "LDAP: Directory Information
                  Models", draft-ietf-ldapbis-models-xx.txt, a work in
                  progress.

    [Roadmap]   Zeilenga, K. (editor), "LDAP: Technical Specification
                  Road Map", draft-ietf-ldapbis-roadmap-xx.txt, a work in
                  progress.

    [Protocol]   Sermersheim, J. (editor), "LDAP: The Protocol",
                  draft-ietf-ldapbis-protocol-xx.txt, a work in progress.

    [Syntaxes]   Legg, S. (editor), "LDAP: Syntaxes and Matching Rules",
                  draft-ietf-ldapbis-syntaxes-xx.txt, a work in progress.

    [Schema]   Dally, K. (editor), "LDAP: User Schema",
                  draft-ietf-ldapbis-user-schema-xx.txt, a work in
                  progress.

[REGISTRY]   IANA, Object Identifier Descriptors Registry,
                    <http://www.iana.org/...>.

9. Informative References

     [ASCII]   Coded Character Set--7-bit American Standard Code for
                    Information Interchange, ANSI X3.4-1986.

     [X.500]   International Telecommunication Union -
                    Telecommunication Standardization Sector, "The Directory
                    -- Overview of concepts, models and services,"
                    X.500(1993) (also ISO/IEC 9594-1:1994).

     [X.690]   International Telecommunication Union -
                    Telecommunication Standardization Sector, "Specification
                    of ASN.1 encoding rules: Basic Encoding Rules (BER),
                    Canonical Encoding Rules (CER), and Distinguished
                    Encoding Rules (DER)", X.690(1997) (also ISO/IEC
                    8825-1:1998).

     [RFC2849]   Good, G., "The LDAP Data Interchange Format (LDIF) -
                    Technical Specification", RFC 2849, June 2000.

     [BCP64bis]   Zeilenga, K., "IANA Considerations for LDAP", draft-
                    ietf-ldapbis-bcp64-xx.txt, a work in progress.

     [CharModel]   Whistler, K. and M. Davis, "Unicode Technical Report
                    #17, Character Encoding Model", UTR17,
                    <http://www.unicode.org/unicode/reports/tr17/>, August
                    2000.

     [Glossary]   The Unicode Consortium, "Unicode Glossary",
                    <http://www.unicode.org/glossary/>.

Appendix A.   Presentation Issues

   This appendix is provided for informational purposes only, it is not a
   normative part of this specification.

The string representation described in this document is not intended
to be presented to humans without translation. However, at times it
may be desirable to present non-translated DN strings to users. This
section discusses presentation issues associated with non-translated
DN strings. Presentation of translated DN strings issues are not
discussed in this appendix. Transcoding issues are also not discussed
in this appendix.

This appendix provides guidance for applications presenting DN strings
to users. This section is not comprehensive, it does not discuss all
presentation issues which implementors may face.

Not all user interfaces are capable of displaying the full set of
Unicode characters. Some Unicode characters are not displayable.

It is recommended that human interfaces use the optional hex pair
escaping mechanism (Section 2.3) to produce a string representation
suitable for display to the user. For example, an application can
generate a DN string for display which escapes all non-printable
characters appearing in the AttributeValue's string representation (as
demonstrated in the final example of Section 4).

When a DN string is displayed in free form text, it is often necessary
to distinguish the DN string from surrounding text. While this is
often done with white space (as demonstrated in Section 4), it is
noted that DN strings may end with white space. Careful readers of
Section 3 will note that characters '<' (U+003C) and '>' (U+003E) may
only appear in the DN string if escaped. These characters are
intended to be used in free form text to distinguish a DN string from
surrounding text. For example, <CN=Sam\ > distinguished the string
representation of the DN comprised of one RDN consisting of the AVA:
the commonName (CN) value 'Sam ' from the surrounding text. It should
be noted to the user that the wrapping '<' and '>' characters are not
part of the DN string.

DN strings can be quite long. It is often desirable to line-wrap
overly long DN strings in presentations. Line wrapping should be done
by inserting white space after the RDN separator character or, if
necessary, after the AVA separator character. It should be noted to
the user that the inserted white space is not part of the DN string

and is to be removed before use in LDAP. For example,

        The following DN string is long:
            CN=Kurt D. Zeilenga,OU=Engineering,L=Redwood Shores,
            O=OpenLDAP Foundation,ST=California,C=US
        so it has been line-wrapped for readability. The extra white
        space is to be removed before the DN string is used in LDAP.

    It is not advised to insert white space otherwise as it may not be
    obvious to the user which white space is part of the DN string and
    which white space was added for readability.

    Another alternative is to use the LDAP Data Interchange Format (LDIF)
    [RFC2849]. For example,

            # This entry has a long DN...
            dn: CN=Kurt D. Zeilenga,OU=Engineering,L=Redwood Shores,
             O=OpenLDAP Foundation,ST=California,C=US
            CN: Kurt D. Zeilenga
            SN: Zeilenga
            objectClass: person

Appendix B. Changes made since RFC 2253

   This appendix is provided for informational purposes only, it is not a
   normative part of this specification.

   The following substantive changes were made to RFC 2253:
     - Removed IESG Note. The IESG Note has been addressed.
     - Replaced all references to ISO 10646-1 with [Unicode].
     - Clarified (in Section 1) that this document does not define a
       canonical string representation.
     - Revised specification (in Section 2) to allow short names of any
       registered attribute type to appear in string representations of
       DNs instead of being restricted to a "published table". Remove
       "as an example" language. Added statement (in Section 3) allowing
       recognition of additional names but require recognization of those
       names in the published table. The table is now published in
       Section 3.
     - Replaced specification of additional requirements for LDAPv2

implementations which also support LDAPv3 ([RFC 2253, Section 4](#))
            with a statement (in [Section 3](#)) allowing recognition of
            alternative string representations.
        - Updated [Section 2.3](#) to indicate attribute type name strings are
          case insensitive.
        - Updated [Section 2.4](#) to allow hex pair escaping of all characters
          and clarified escaping for when multiple octet UTF-8 echodings are
          present.
        - Rewrote [Section 3](#) to use ABNF as defined in [RFC 2234](#).
        - Rewrote [Section 3](#) ABNF to be consistent with 2.4.
        - Updated [Section 3](#) to describe how to parse elements of the
          grammar.
        - Rewrote examples.
        - Added reference to documentations containing general LDAP security
          considerations.
        - Added discussion of presentation issues (Appendix A).
        - Added this appendix.

    In addition, numerous editorial changes were made.

Intellectual Property Rights

    The IETF takes no position regarding the validity or scope of any
    intellectual property 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; neither does it represent that it has made any
    effort to identify any such rights. Information on the IETF's
    procedures with respect to rights in standards-track and

    standards-related documentation can be found in [BCP-11](#). Copies of
    claims of rights made available for publication 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 implementors or users of this specification can be obtained
    from the IETF Secretariat.

    The IETF invites any interested party to bring to its attention any
    copyrights, patents or patent applications, or other proprietary
    rights which may cover technology that may be required to practice
    this standard. Please address the information to the IETF Executive

Director.

Full Copyright

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