Network Working Group
Request for Comments: 4514
Obsoletes: 2253
Category: Standards Track

K. Zeilenga, Ed.
OpenLDAP Foundation

June 2006

Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names

Status of This Memo

Copyright Notice

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) [RFC4510], distinguished names (DNs) are used to unambiguously refer to directory entries [X.501][RFC4512].

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 that they 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 [RFC4511][RFC4517]. 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 an 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 that 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 [RFC4517].

This document is a integral part of the LDAP technical specification [RFC4510], which obsoletes the previously defined LDAP technical specification, RFC 3377, in its entirety. 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][RFC4512].

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 a 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 that 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 sign ('=' 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 (descriptor) [RFC4512] and that short name is known to be registered [REGISTRY] [RFC4520] 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> are defined in [RFC4512].

Implementations are not expected to dynamically update their knowledge of registered short names. However, implementations SHOULD provide a mechanism to allow their 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 an LDAP-specific ([RFC4517], 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 that has a LDAP-specific string encoding, the value is converted first to a UTF-8-encoded Unicode string according to its syntax specification (see [RFC4517], Section 3.3, for examples). If that UTF-8-encoded Unicode string does not have any of the following characters that 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+005C).

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 [RFC4234] 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>, and <UTFMB> are defined in [RFC4512].

Each <attributeType>, either a <descr> or a <numericoid>, refers to an attribute type of an attribute value assertion (AVA). The <attributeType> is followed by an <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>.

There is one or more attribute value assertions, separated by <PLUS>, for a relative distinguished name.

There is 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)

These attribute types are described in [RFC4519].

Implementations MAY recognize other DN string representations. However, as there is no requirement that alternative DN string representations 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 of a 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 special characters appearing in a common name:

```
CN=James \"Jim\" Smith\, III,DC=example,DC=net
```

The following shows the method for encoding a value that contains a carriage return character:

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

In this RDN example, the type in the RDN is unrecognized, and 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
```

Finally, this example shows an RDN whose commonName value consists 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

This could be encoded in printable ASCII [ASCII] (useful for debugging purposes) as:

```
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 [RFC4511] and other documents comprising the LDAP Technical Specification [RFC4510].

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)

In some cases, such information can be considered sensitive. In many countries, privacy laws exist that prohibit disclosure of certain kinds of descriptive information (e.g., email addresses). Hence, server implementers are encouraged to support Directory Information Tree (DIT) structural rules and name forms [RFC4512], as these provide a mechanism for administrators to select appropriate naming attributes for entries. Administrators are encouraged to use mechanisms, access controls, and other administrative controls that may be available to restrict use of attributes containing sensitive information in naming of entries. Additionally, use of

authentication and data security services in LDAP [RFC4513][RFC4511] should be considered.

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 that 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 is of the PrintableString choice, would have the same representation <CN=Sam>.

Applications that 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 ('#' U+0023) as described in the first paragraph of Section 2.4.

6. Acknowledgements

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. References

7.1. Normative References

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

[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/).

[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, RFC 2119, March 1997.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC4234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005.

[RFC4510] Zeilenga, K., Ed., "Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map", RFC 4510, June 2006.

[RFC4511] Sermersheim, J., Ed., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

[RFC4512] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): Directory Information Models", RFC 4512, June 2006.

[RFC4513] Harrison, R., Ed., "Lightweight Directory Access Protocol (LDAP): Authentication Methods and Security Mechanisms", RFC 4513, June 2006.

[RFC4517] Legg, S., Ed., "Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules", RFC 4517, June 2006.

[RFC4519] Sciberras, A., Ed., "Lightweight Directory Access Protocol (LDAP): Schema for User Applications", RFC 4519, June 2006.

[RFC4520] Zeilenga, K., "Internet Assigned Numbers Authority (IANA) Considerations for the Lightweight Directory Access Protocol (LDAP)", BCP 64, RFC 4520, June 2006.

7.2. Informative References

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

[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/>.

[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.511] International Telecommunication Union - Telecommunication Standardization Sector, "The Directory: Abstract Service Definition", X.511(1993) (also ISO/IEC 9594-3:1993).

[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.

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. Issues with presentation of translated DN strings 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 that implementers 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 that 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 whitespace (as demonstrated in Section 4), it is noted that DN strings may end with whitespace. Careful readers of Section 3 will note that the 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\ > distinguishes the string representation of the DN composed 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 whitespace after the RDN separator character or, if necessary, after the AVA separator character. It should be noted to the user that the inserted whitespace 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 whitespace is to be removed before the DN string is used in LDAP.

Inserting whitespace is not advised because it may not be obvious to the user which whitespace is part of the DN string and which whitespace 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.
- Clarified that Section 2 describes the RECOMMENDED encoding algorithm and that alternative algorithms are allowed. Some encoding options described in RFC 2253 are now treated as alternative algorithms in this specification.
- 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". Removed "as an example" language. Added statement (in Section 3) allowing recognition of additional names but require recognition of those names in the published table. The table now appears in Section 3.
- Removed specification of additional requirements for LDAPv2 implementations which also support LDAPv3 (RFC 2253, Section 4) as LDAPv2 is now Historic.
- Allowed recognition of alternative string representations.
- Updated Section 2.4 to allow hex pair escaping of all characters and clarified escaping for when multiple octet UTF-8 encodings

are present. Indicated that null (U+0000) character is to be escaped. Indicated that equals sign ('=' U+003D) character may be escaped as '\='.
- Rewrote Section 3 to use ABNF as defined in RFC 4234.
- Updated the Section 3 ABNF. Changes include:
  + allowed AttributeType short names of length 1 (e.g., 'L'),
  + used more restrictive <oid> production in AttributeTypes,
  + did not require escaping of equals sign ('=' U+003D) characters,
  + did not require escaping of non-leading number sign ('#' U+0023) characters,
  + allowed space (' ' U+0020) to be escaped as '\ ',
  + required hex escaping of null (U+0000) characters, and
  + removed LDAPv2-only constructs.
- 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.

Editor's Address

Kurt D. Zeilenga
OpenLDAP Foundation

EMail: Kurt@OpenLDAP.org

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).