LDAP: Internationalized String Preparation
<draft-ietf-ldapbis-strprep-06.txt>

Status of this Memo

   This document is intended to be published as a Standard Track RFC.
   Distribution of this memo is unlimited. Technical discussion of this
   document will take place on the IETF LDAP Revision Working Group
   mailing list <ietf-ldapbis@openldap.org>. Please send editorial
   comments directly to the editor <Kurt@OpenLDAP.org>.

   By submitting this Internet-Draft, each author represents that any
   applicable patent or other IPR claims of which he or she is aware have
   been or will be disclosed, and any of which he or she becomes aware
   will be disclosed, in accordance with Section 6 of BCP 79.

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

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

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

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

Internet-Draft   draft-ietf-ldapbis-strprep-06   30 September 2005

Abstract

   The previous Lightweight Directory Access Protocol (LDAP) technical
   specifications did not precisely define how character string matching
   is to be performed. This led to a number of usability and
   interoperability problems. This document defines string preparation
   algorithms for character-based matching rules defined for use in LDAP.

Conventions and Terms

   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>.
   In the lists of mappings and the prohibited characters, the "U+" is
   left off to make the lists easier to read. The comments for character
   ranges are shown in square brackets (such as "[CONTROL CHARACTERS]")
   and do not come from the standard.

   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].

   The term "combining mark", as used in this specification, refers to
   any Unicode [Unicode] code point which has a mark property (Mn, Mc,
   Me). Appendix A provides a complete list of combining marks.

1. Introduction

1.1. Background

   A Lightweight Directory Access Protocol (LDAP) [Roadmap] matching rule
   [Syntaxes] defines an algorithm for determining whether a presented
   value matches an attribute value in accordance with the criteria
   defined for the rule. The proposition may be evaluated to True,
   False, or Undefined.

        True   - the attribute contains a matching value,

False   - the attribute contains no matching value,

        Undefined – it cannot be determined whether the attribute contains
                    a matching value or not.

   For instance, the caseIgnoreMatch matching rule may be used to compare
   whether the commonName attribute contains a particular value without
   regard for case and insignificant spaces.

1.2. X.500 String Matching Rules

   "X.520: Selected attribute types" [X.520] provides (amongst other
   things) value syntaxes and matching rules for comparing values
   commonly used in the Directory. These specifications are inadequate
   for strings composed of Unicode [Unicode] characters.

   The caseIgnoreMatch matching rule [X.520], for example, is simply
   defined as being a case insensitive comparison where insignificant
   spaces are ignored. For printableString, there is only one space
   character and case mapping is bijective, hence this definition is
   sufficient. However, for Unicode string types such as
   universalString, this is not sufficient. For example, a case
   insensitive matching implementation which folded lower case characters
   to upper case would yield different different results than an
   implementation which used upper case to lower case folding. Or one
   implementation may view space as referring to only SPACE (U+0020), a
   second implementation may view any character with the space separator
   (Zs) property as a space, and another implementation may view any
   character with the whitespace (WS) category as a space.

   The lack of precise specification for character string matching has
   led to significant interoperability problems. When used in
   certificate chain validation, security vulnerabilities can arise. To
   address these problems, this document defines precise algorithms for
   preparing character strings for matching.

1.3. Relationship to "stringprep"

   The character string preparation algorithms described in this document

are based upon the "stringprep" approach [StringPrep]. In
"stringprep", presented and stored values are first prepared for
comparison and so that a character-by-character comparison yields the
"correct" result.

The approach used here is a refinement of the "stringprep"
[StringPrep] approach. Each algorithm involves two additional
preparation steps.

a) prior to applying the Unicode string preparation steps outlined in
   "stringprep", the string is transcoded to Unicode;

b) after applying the Unicode string preparation steps outlined in
   "stringprep", the string is modified to appropriately handle
   characters insignificant to the matching rule.

Hence, preparation of character strings for X.500 matching involves
the following steps:

    1) Transcode
    2) Map
    3) Normalize
    4) Prohibit
    5) Check Bidi (Bidirectional)
    6) Insignificant Character Handling

These steps are described in Section 2.

1.4. Relationship to the LDAP Technical Specification

This document is a integral part of the LDAP technical specification
[Roadmap] which obsoletes the previously defined LDAP technical
specification [RFC3377] in its entirety.

This document details new LDAP internationalized character string
preparation algorithms used by [Syntaxes] and possible other technical
specifications defining LDAP syntaxes and/or matching rules.

1.5. Relationship to X.500

LDAP is defined [Roadmap] in X.500 terms as an X.500 access mechanism. As such, there is a strong desire for alignment between LDAP and X.500 syntax and semantics. The character string preparation algorithms described in this document are based upon "Internationalized String Matching Rules for X.500" [XMATCH] proposal to ITU/ISO Joint Study Group 2.

2. String Preparation

The following six-step process SHALL be applied to each presented and attribute value in preparation for character string matching rule evaluation.

       1) Transcode
       2) Map
       3) Normalize
       4) Prohibit

       5) Check bidi
       6) Insignificant Character Handling

   Failure in any step causes the assertion to evaluate to Undefined.

   The character repertoire of this process is Unicode 3.2 [Unicode].

2.1. Transcode

   Each non-Unicode string value is transcoded to Unicode.

   PrintableString [X.680] value are transcoded directly to Unicode.

   UniversalString, UTF8String, and bmpString [X.680] values need not be transcoded as they are Unicode-based strings (in the case of bmpString, a subset of Unicode).

   TeletexString [X.680] values are transcoded to Unicode. As there is no standard for mapping TelexString values to Unicode, the mapping is left a local matter.

For these and other reasons, use of TeletexString is NOT RECOMMENDED.

The output is the transcoded string.

2.2. Map

SOFT HYPHEN (U+00AD) and MONGOLIAN TODO SOFT HYPHEN (U+1806) code points are mapped to nothing. COMBINING GRAPHEME JOINER (U+034F) and VARIATION SELECTORs (U+180B-180D, FF00-FE0F) code points are also mapped to nothing. The OBJECT REPLACEMENT CHARACTER (U+FFFC) is mapped to nothing.

CHARACTER TABULATION (U+0009), LINE FEED (LF) (U+000A), LINE TABULATION (U+000B), FORM FEED (FF) (U+000C), CARRIAGE RETURN (CR) (U+000D), and NEXT LINE (NEL) (U+0085) are mapped to SPACE (U+0020).

All other control code (e.g., Cc) points or code points with a control function (e.g., Cf) are mapped to nothing. The following is a complete list of these code points: U+0000-0008, 000E-001F, 007F-0084, 0086-009F, 06DD, 070F, 180E, 200C-200F, 202A-202E, 2060-2063, 206A-206F, FEFF, FFF9-FFFB, 1D173-1D17A, E0001, E0020-E007F.

ZERO WIDTH SPACE (U+200B) is mapped to nothing. All other code points with Separator (space, line, or paragraph) property (e.g, Zs, Zl, or Zp) are mapped to SPACE (U+0020). The following is a complete list of

these code points: U+0020, 00A0, 1680, 2000-200A, 2028-2029, 202F, 205F, 3000.

For case ignore, numeric, and stored prefix string matching rules, characters are case folded per B.2 of [StringPrep].

The output is the mapped string.

2.3. Normalize

The input string is be normalized to Unicode Form KC (compatibility composed) as described in [UAX15]. The output is the normalized string.

2.4. Prohibit

All Unassigned code points are prohibited. Unassigned code points are listed in Table A.1 of [StringPrep].

Characters which, per Section 5.8 of [Stringprep], change display properties or are deprecated are prohibited. These characters are listed in Table C.8 of [StringPrep].

Private Use code points are prohibited. These characters are listed in Table C.3 of [StringPrep].

All non-character code points are prohibited. These code points are listed in Table C.4 of [StringPrep].

Surrogate codes are prohibited. These characters are listed in Table C.5 of [StringPrep].

The REPLACEMENT CHARACTER (U+FFFD) code point is prohibited.

The step fails if the input string contains any prohibited code point. Otherwise, the output is the input string.

2.5. Check bidi

Bidirectional characters are ignored.

2.6. Insignificant Character Handling

In this step, the string is modified to ensure proper handling of

characters insignificant to the matching rule. This modification differs from matching rule to matching rule.

Section 2.6.1 applies to case ignore and exact string matching.
Section 2.6.2 applies to numericString matching.
Section 2.6.3 applies to telephoneNumber matching.

2.6.1. Insignificant Space Handling

   For the purposes of this section, a space is defined to be the SPACE
   (U+0020) code point followed by no combining marks.

   NOTE - The previous steps ensure that the string cannot contain any
          code points in the separator class, other than SPACE (U+0020).

   If the input string contains at least one non-space character, then
   the string is modified such that the string starts with exactly one
   space character, ends with exactly one SPACE character, and that any
   inner (non-empty) sequence of space characters is replaced with
   exactly two SPACE characters. For instance, the input strings
   "foo<SPACE>bar<SPACE><SPACE>", results in the output
   "<SPACE>foo<SPACE><SPACE>bar<SPACE>".

   Otherwise, if the string being prepared is an initial, any, or final
   substring, then the output string is exactly one SPACE character, else
   the output string is exactly two SPACEs.

   Appendix B discusses the rationale for the behavior.

2.6.2. numericString Insignificant Character Handling

   For the purposes of this section, a space is defined to be the SPACE
   (U+0020) code point followed by no combining marks.

   All spaces are regarded as insignificant and are to be removed.

   For example, removal of spaces from the Form KC string:
       "<SPACE><SPACE>123<SPACE><SPACE>456<SPACE><SPACE>"
   would result in the output string:
       "123456"
   and the Form KC string:
       "<SPACE><SPACE><SPACE>"
   would result in the output string:
       "" (an empty string).

2.6.3. telephoneNumber Insignificant Character Handling

For the purposes of this section, a hyphen is defined to be
   HYPHEN-MINUS (U+002D), ARMENIAN HYPHEN (U+058A), HYPHEN (U+2010),
   NON-BREAKING HYPHEN (U+2011), MINUS SIGN (U+2212), SMALL HYPHEN-MINUS
   (U+FE63), or FULLWIDTH HYPHEN-MINUS (U+FF0D) code point followed by no
   combining marks and a space is defined to be the SPACE (U+0020) code
   point followed by no combining marks.

   All hyphens and spaces are considered insignificant and are to be
   removed.

   For example, removal of hyphens and spaces from the Form KC string:
       "<SPACE><HYPHEN>123<SPACE><SPACE>456<SPACE><HYPHEN>"
   would result in the output string:
       "123456"
   and the Form KC string:
       "<HYPHEN><HYPHEN><HYPHEN>"
   would result in the (empty) output string:
       "".

3. Security Considerations

   "Preparation for International Strings ('stringprep')" [StringPrep]
   security considerations generally apply to the algorithms described
   here.

4. Acknowledgments

   The approach used in this document is based upon design principles and
   algorithms described in "Preparation of Internationalized Strings
   ('stringprep')" [StringPrep] by Paul Hoffman and Marc Blanchet. Some
   additional guidance was drawn from Unicode Technical Standards,
   Technical Reports, and Notes.

   This document is a product of the IETF LDAP Revision (LDAPBIS) Working
   Group.

5. Author's Address

   Kurt D. Zeilenga
   OpenLDAP Foundation

   Email: Kurt@OpenLDAP.org

6. References

[[Note to the RFC Editor: please replace the citation tags used in referencing Internet-Drafts with tags of the form RFCnnnn where possible.]]

6.1. Normative References

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

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

[StringPrep] Hoffman P. and M. Blanchet, "Preparation of
              Internationalized Strings ('stringprep')",
              draft-hoffman-rfc3454bis-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.

[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/).

[UAX15]   Davis, M. and M. Duerst, "Unicode Standard Annex #15:
              Unicode Normalization Forms, Version 3.2.0".
              <http://www.unicode.org/unicode/reports/tr15/tr15-22.html>,
              March 2002.

[X.680]   International Telecommunication Union -
              Telecommunication Standardization Sector, "Abstract
              Syntax Notation One (ASN.1) - Specification of Basic
              Notation", X.680(2002) (also ISO/IEC 8824-1:2002).

6.2. Informative References

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

   [X.520]   International Telecommunication Union –
                   Telecommunication Standardization Sector, "The
                   Directory: Selected Attribute Types", X.520(1993) (also
                   ISO/IEC 9594-6:1994).

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

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

   [XMATCH]   Zeilenga, K., "Internationalized String Matching Rules
                   for X.500", draft-zeilenga-ldapbis-strmatch-xx.txt, a
                   work in progress.

   [RFC1345]   Simonsen, K., "Character Mnemonics & Character Sets",
                   RFC 1345, June 1992.

Appendix A.   Combining Marks

   This appendix is normative.

   0300-034F 0360-036F 0483-0486 0488-0489 0591-05A1 05A3-05B9 05BB-05BC
   05BF 05C1-05C2 05C4 064B-0655 0670 06D6-06DC 06DE-06E4 06E7-06E8
   06EA-06ED 0711 0730-074A 07A6-07B0 0901-0903 093C 093E-094F 0951-0954
   0962-0963 0981-0983 09BC 09BE-09C4 09C7-09C8 09CB-09CD 09D7 09E2-09E3
   0A02 0A3C 0A3E-0A42 0A47-0A48 0A4B-0A4D 0A70-0A71 0A81-0A83 0ABC
   0ABE-0AC5 0AC7-0AC9 0ACB-0ACD 0B01-0B03 0B3C 0B3E-0B43 0B47-0B48
   0B4B-0B4D 0B56-0B57 0B82 0BBE-0BC2 0BC6-0BC8 0BCA-0BCD 0BD7 0C01-0C03
   0C3E-0C44 0C46-0C48 0C4A-0C4D 0C55-0C56 0C82-0C83 0CBE-0CC4 0CC6-0CC8
   0CCA-0CCD 0CD5-0CD6 0D02-0D03 0D3E-0D43 0D46-0D48 0D4A-0D4D 0D57
   0D82-0D83 0DCA 0DCF-0DD4 0DD6 0DD8-0DDF 0DF2-0DF3 0E31 0E34-0E3A

```
0E47-0E4E 0EB1 0EB4-0EB9 0EBB-0EBC 0EC8-0ECD 0F18-0F19 0F35 0F37 0F39
0F3E-0F3F 0F71-0F84 0F86-0F87 0F90-0F97 0F99-0FBC 0FC6 102C-1032
1036-1039 1056-1059 1712-1714 1732-1734 1752-1753 1772-1773 17B4-17D3
180B-180D 18A9 20D0-20EA 302A-302F 3099-309A FB1E FE00-FE0F FE20-FE23
1D165-1D169 1D16D-1D172 1D17B-1D182 1D185-1D18B 1D1AA-1D1AD
```

Appendix B.   Substrings Matching

   In absence of substrings matching, the insignificant space handling
   for case ignore/exact matching could be simplified. Specifically,
   the handling could be as require all sequences of one or more spaces
   be replaced with one space and, if string contains non-space
   characters, removal of all all leading spaces and trailing spaces.

   In the presence of substrings matching, this simplified space handling
   this simplified space handling would lead to unexpected and
   undesirable matching behavior. For instance:
   1) (CN=foo\20*\20bar) would match the CN value "foobar" but not
      "foo<SPACE>bar" nor "foo<SPACE><SPACE>bar";
   2) (CN=*\20foobar\20*) would match "foobar", but (CN=*\20*foobar*\20*)
      would not;
   3) (CN=foo\20*\20bar) would match "foo<SPACE>X<SPACE>bar" but not
      "foo<SPACE><SPACE>bar".

   The first case illustrates that this simplified space handling would
   cause leading and trailing spaces in substrings of the string to be
   regarded as insignificant. However, only leading and trailing (as
   well as multiple consecutive spaces) of the string (as a whole) are
   insignificant.

   The second case illustrates that this simplified space handling would
   cause sub-partitioning failures. That is, if a prepared any substring
   matches a partition of the attribute value, then an assertion
   constructed by subdividing that substring into multiple substrings
   should also match.

   The third case illustrates that this simplified space handling causes
   another partitioning failure. Though both the initial or final
   strings match different portions of "foo<SPACE>X<SPACE>bar" with
   neither matching the X portion, they don't match a string consisting

of the two matched portions less the unmatched X portion.

In designing an appropriate approach for space handling for substrings matching, one must study key aspects of X.500 case exact/ignore matching. X.520 [X.520] says:
    The [substrings] rule returns TRUE if there is a partitioning of the attribute value (into portions) such that:
    - the specified substrings (initial, any, final) match different portions of the value in the order of the strings sequence;
    - initial, if present, matches the first portion of the value;
    - final, if present, matches the last portion of the value;
    - any, if present, matches some arbitrary portion of the value.

That is, the substrings assertion (CN=foo\20*\20bar) matches the attribute value "foo<SPACE><SPACE>bar" as the value can be partitioned into the portions "foo<SPACE>" and "<SPACE>bar" meeting the above

requirements.

X.520 also says:
    [T]he following spaces are regarded as not significant:
    - leading spaces (i.e. those preceding the first character that is not a space);
    - trailing spaces (i.e. those following the last character that is not a space);
    - multiple consecutive spaces (these are taken as equivalent to a single space character).

This statement applies to the assertion values and attribute values as whole strings, and not individually to substrings of an assertion value. In particular, the statements should be taken to mean that if an assertion value and attribute value match without any consideration to insignificant characters, then that assertion value should also match any attribute value which differs only by inclusion or removal of insignificant characters.

Hence, the assertion (CN=foo\20*\20bar) matches "foo<SPACE><SPACE><SPACE>bar" and "foo<SPACE>bar" as these values only differ from "foo<SPACE><SPACE>bar" by the inclusion or removal of insignificant spaces.

Astute readers of this text will also note that there are special

cases where the specified space handling does not ignore spaces
which could be considered insignificant. For instance, the assertion
(CN=\20*\20*\20) does not match "<SPACE><SPACE><SPACE>"
(insignificant spaces present in value) nor " " (insignificant
spaces not present in value). However, as these cases have no
practical application that cannot be met by simple assertions, e.g.
(cn=\20), and this minor anomaly can only be fully addressed by a
preparation algorithm to be used in conjunction with
character-by-character partitioning and matching, the anomaly is
considered acceptable.

Intellectual Property Rights

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

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

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

Full Copyright

   Copyright (C) The Internet Society (2005).