Network Working Group
Request for Comments: 4515
Obsoletes: 2254
Category: Standards Track

M. Smith, Ed.
Pearl Crescent, LLC
T. Howes
Opsware, Inc.

June 2006

# Lightweight Directory Access Protocol (LDAP): String Representation of Search Filters

## Status of This Memo

## Copyright Notice

## Abstract

Lightweight Directory Access Protocol (LDAP) search filters are transmitted in the LDAP protocol using a binary representation that is appropriate for use on the network. This document defines a human-readable string representation of LDAP search filters that is appropriate for use in LDAP URLs (RFC 4516) and in other applications.

## Table of Contents

## 1. Introduction

The Lightweight Directory Access Protocol (LDAP) [RFC4510] defines a network representation of a search filter transmitted to an LDAP server. Some applications may find it useful to have a common way of representing these search filters in a human-readable form; LDAP URLs [RFC4516] are an example of one such application. This document defines a human-readable string format for representing the full range of possible LDAP version 3 search filters, including extended match filters.

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 replaces RFC 2254. Changes to RFC 2254 are summarized in Appendix A.

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].

## 2. LDAP Search Filter Definition

An LDAP search filter is defined in Section 4.5.1 of [RFC4511] as follows:

```
     Filter ::= CHOICE {
         and                [0] SET SIZE (1..MAX) OF filter Filter,
         or                 [1] SET SIZE (1..MAX) OF filter Filter,
         not                [2] Filter,
         equalityMatch      [3] AttributeValueAssertion,
         substrings         [4] SubstringFilter,
         greaterOrEqual     [5] AttributeValueAssertion,
         lessOrEqual        [6] AttributeValueAssertion,
         present            [7] AttributeDescription,
         approxMatch        [8] AttributeValueAssertion,
         extensibleMatch    [9] MatchingRuleAssertion }

     SubstringFilter ::= SEQUENCE {
         type    AttributeDescription,
         -- initial and final can occur at most once
         substrings    SEQUENCE SIZE (1..MAX) OF substring CHOICE {
          initial        [0] AssertionValue,
          any            [1] AssertionValue,
          final          [2] AssertionValue } }
```

```
        AttributeValueAssertion ::= SEQUENCE {
            attributeDesc   AttributeDescription,
            assertionValue  AssertionValue }

        MatchingRuleAssertion ::= SEQUENCE {
            matchingRule    [1] MatchingRuleId OPTIONAL,
            type            [2] AttributeDescription OPTIONAL,
            matchValue      [3] AssertionValue,
            dnAttributes    [4] BOOLEAN DEFAULT FALSE }

        AttributeDescription ::= LDAPString
                        -- Constrained to <attributedescription>
                        -- [RFC4512]

        AttributeValue ::= OCTET STRING

        MatchingRuleId ::= LDAPString

        AssertionValue ::= OCTET STRING

        LDAPString ::= OCTET STRING -- UTF-8 encoded,
                                    -- [Unicode] characters
```

The AttributeDescription, as defined in [RFC4511], is a string representation of the attribute description that is discussed in [RFC4512]. The AttributeValue and AssertionValue OCTET STRING have the form defined in [RFC4517]. The Filter is encoded for transmission over a network using the Basic Encoding Rules (BER) defined in [X.690], with simplifications described in [RFC4511].

## 3. String Search Filter Definition

The string representation of an LDAP search filter is a string of UTF-8 [RFC3629] encoded Unicode characters [Unicode] that is defined by the following grammar, following the ABNF notation defined in [RFC4234]. The productions used that are not defined here are defined in Section 1.4 (Common ABNF Productions) of [RFC4512] unless otherwise noted. The filter format uses a prefix notation.

```
      filter         = LPAREN filtercomp RPAREN
      filtercomp     = and / or / not / item
      and            = AMPERSAND filterlist
      or             = VERTBAR filterlist
      not            = EXCLAMATION filter
      filterlist     = 1*filter
      item           = simple / present / substring / extensible
      simple         = attr filtertype assertionvalue
      filtertype     = equal / approx / greaterorequal / lessorequal
```

```
      equal          = EQUALS
      approx         = TILDE EQUALS
      greaterorequal = RANGLE EQUALS
      lessorequal    = LANGLE EQUALS
      extensible     = ( attr [dnattrs]
                           [matchingrule] COLON EQUALS assertionvalue )
                       / ( [dnattrs]
                            matchingrule COLON EQUALS assertionvalue )
      present        = attr EQUALS ASTERISK
      substring      = attr EQUALS [initial] any [final]
      initial        = assertionvalue
      any            = ASTERISK *(assertionvalue ASTERISK)
      final          = assertionvalue
      attr           = attributedescription
                         ; The attributedescription rule is defined in
                         ; Section 2.5 of [RFC4512].
      dnattrs        = COLON "dn"
      matchingrule   = COLON oid
      assertionvalue = valueencoding
      ; The <valueencoding> rule is used to encode an <AssertionValue>
      ; from Section 4.1.6 of [RFC4511].
      valueencoding  = 0*(normal / escaped)
      normal         = UTF1SUBSET / UTFMB
      escaped        = ESC HEX HEX
      UTF1SUBSET     = %x01-27 / %x2B-5B / %x5D-7F
                          ; UTF1SUBSET excludes 0x00 (NUL), LPAREN,
                          ; RPAREN, ASTERISK, and ESC.
      EXCLAMATION    = %x21 ; exclamation mark ("!")
      AMPERSAND      = %x26 ; ampersand (or AND symbol) ("&")
      ASTERISK       = %x2A ; asterisk ("*")
      COLON          = %x3A ; colon (":")
      VERTBAR        = %x7C ; vertical bar (or pipe) ("|")
      TILDE          = %x7E ; tilde ("~")
```

Note that although both the <substring> and <present> productions in the grammar above can produce the "attr=*" construct, this construct is used only to denote a presence filter.

The <valueencoding> rule ensures that the entire filter string is a valid UTF-8 string and provides that the octets that represent the ASCII characters "*" (ASCII 0x2a), "(" (ASCII 0x28), ")" (ASCII 0x29), "\" (ASCII 0x5c), and NUL (ASCII 0x00) are represented as a backslash "\" (ASCII 0x5c) followed by the two hexadecimal digits representing the value of the encoded octet.

This simple escaping mechanism eliminates filter-parsing ambiguities and allows any filter that can be represented in LDAP to be represented as a NUL-terminated string. Other octets that are part of the <normal> set may be escaped using this mechanism, for example, non-printing ASCII characters.

For AssertionValues that contain UTF-8 character data, 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. For example, the filter checking whether the "cn" attribute contained a value with the character "*" anywhere in it would be represented as "(cn=*\2a*)".

As indicated by the <valueencoding> rule, implementations MUST escape all octets greater than 0x7F that are not part of a valid UTF-8 encoding sequence when they generate a string representation of a search filter. Implementations SHOULD accept as input strings that are not valid UTF-8 strings. This is necessary because RFC 2254 did not clearly define the term "string representation" (and in particular did not mention that the string representation of an LDAP search filter is a string of UTF-8-encoded Unicode characters).

## 4. Examples

This section gives a few examples of search filters written using this notation.

```
     (cn=Babs Jensen)
     (!(cn=Tim Howes))
     (&(objectClass=Person)(|(sn=Jensen)(cn=Babs J*)))
     (o=univ*of*mich*)
     (seeAlso=)
```

The following examples illustrate the use of extensible matching.

```
     (cn:caseExactMatch:=Fred Flintstone)
     (cn:=Betty Rubble)
     (sn:dn:2.4.6.8.10:=Barney Rubble)
     (o:dn:=Ace Industry)
     (:1.2.3:=Wilma Flintstone)
     (:DN:2.4.6.8.10:=Dino)
```

The first example shows use of the matching rule "caseExactMatch."

The second example demonstrates use of a MatchingRuleAssertion form without a matchingRule.

The third example illustrates the use of the ":oid" notation to indicate that the matching rule identified by the OID "2.4.6.8.10" should be used when making comparisons, and that the attributes of an entry's distinguished name should be considered part of the entry when evaluating the match (indicated by the use of ":dn").

The fourth example denotes an equality match, except that DN components should be considered part of the entry when doing the match.

The fifth example is a filter that should be applied to any attribute supporting the matching rule given (since the <attr> has been omitted).

The sixth and final example is also a filter that should be applied to any attribute supporting the matching rule given. Attributes supporting the matching rule contained in the DN should also be considered.

The following examples illustrate the use of the escaping mechanism.

```
(o=Parens R Us \28for all your parenthetical needs\29)
(cn=*\2A*)
(filename=C:\5cMyFile)
(bin=\00\00\00\04)
(sn=Lu\c4\8di\c4\87)
(1.3.6.1.4.1.1466.0=\04\02\48\69)
```

The first example shows the use of the escaping mechanism to represent parenthesis characters. The second shows how to represent a "*" in an assertion value, preventing it from being interpreted as a substring indicator. The third illustrates the escaping of the backslash character.

The fourth example shows a filter searching for the four-octet value 00 00 00 04 (hex), illustrating the use of the escaping mechanism to represent arbitrary data, including NUL characters.

The fifth example illustrates the use of the escaping mechanism to represent various non-ASCII UTF-8 characters. Specifically, there are 5 characters in the <assertionvalue> portion of this example: LATIN CAPITAL LETTER L (U+004C), LATIN SMALL LETTER U (U+0075), LATIN SMALL LETTER C WITH CARON (U+010D), LATIN SMALL LETTER I (U+0069), and LATIN SMALL LETTER C WITH ACUTE (U+0107).

The sixth and final example demonstrates assertion of a BER-encoded value.

## 5. Security Considerations

This memo describes a string representation of LDAP search filters. While the representation itself has no known security implications, LDAP search filters do. They are interpreted by LDAP servers to select entries from which data is retrieved. LDAP servers should take care to protect the data they maintain from unauthorized access.

Please refer to the Security Considerations sections of [RFC4511] and [RFC4513] for more information.

## 6. Normative References

[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.

[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."

## 7. Informative References

[RFC4516] Smith, M., Ed. and T. Howes, "Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator", RFC 4516, June 2006.

[X.690] Specification of ASN.1 encoding rules: Basic, Canonical, and Distinguished Encoding Rules, ITU-T Recommendation X.690, 1994.

## 8. Acknowledgements

This document replaces RFC 2254 by Tim Howes. RFC 2254 was a product of the IETF ASID Working Group.

Changes included in this revised specification are based upon discussions among the authors, discussions within the LDAP (v3) Revision Working Group (ldapbis), and discussions within other IETF Working Groups. The contributions of individuals in these working groups is gratefully acknowledged.

## Appendix A: Changes Since RFC 2254

### A.1.  Technical Changes

Replaced [ISO 10646] reference with [Unicode].

The following technical changes were made to the contents of the "String Search Filter Definition" section:

Added statement that the string representation is a string of UTF-8-encoded Unicode characters.

Revised all of the ABNF to use common productions from [RFC4512].

Replaced the "value" rule with a new "assertionvalue" rule within the "simple", "extensible", and "substring" ("initial", "any", and "final") rules.  This matches a change made in [RFC4517].

Added "(" and ")" around the components of the <extensible> subproductions for clarity.

Revised the "attr", "matchingrule", and "assertionvalue" ABNF to more precisely reference productions from the [RFC4512] and [RFC4511] documents.

"String Search Filter Definition" section: replaced "greater" and "less" with "greaterorequal" and "lessorequal" to avoid confusion.

Introduced the "valueencoding" and associated "normal" and "escaped" rules to reduce the dependence on descriptive text.  The "normal" production restricts filter strings to valid UTF-8 sequences.

Added a statement about expected behavior in light of RFC 2254's lack of a clear definition of "string representation."

### A.2.  Editorial Changes

Changed document title to include "LDAP:" prefix.

IESG Note: removed note about lack of satisfactory mandatory authentication mechanisms.

Header and "Authors' Addresses" sections: added Mark Smith as the document editor and updated affiliation and contact information.

"Table of Contents" and "Intellectual Property" sections: added.

Copyright: updated per latest IETF guidelines.

"Abstract" section: separated from introductory material.

"Introduction" section: new section; separated from the Abstract. Updated second paragraph to indicate that RFC 2254 is replaced by this document (instead of RFC 1960). Added reference to the [RFC4510] document.

"LDAP Search Filter Definition" section: made corrections to the LDAP search filter ABNF so it matches that used in [RFC4511].

Clarified the definition of 'value' (now 'assertionvalue') to take into account the fact that it is not precisely an AttributeAssertion from [RFC4511] Section 4.1.6 (special handling is required for some characters). Added a note that each octet of a character to be escaped is replaced by a backslash and two hex digits, which represent a single octet.

"Examples" section: added four additional examples: (seeAlso=), (cn:=Betty Rubble), (:1.2.3:=Wilma Flintstone), and (1.3.6.1.4.1.1466.0=\04\02\48\69). Replaced one occurrence of "a value" with "an assertion value". Corrected the description of this example: (sn:dn:2.4.6.8.10:=Barney Rubble). Replaced the numeric OID in the first extensible match example with "caseExactMatch" to demonstrate use of the descriptive form. Used "DN" (uppercase) in the last extensible match example to remind the reader to treat the <dnattrs> production as case insensitive. Reworded the description of the fourth escaping mechanism example to avoid making assumptions about byte order. Added text to the fifth escaping mechanism example to spell out what the non-ASCII characters are in Unicode terms.

"Security Considerations" section: added references to [RFC4511] and [RFC4513].

"Normative References" section: renamed from "References" per new RFC guidelines. Changed from [1] style to [RFC4511] style throughout the document. Added entries for [Unicode], [RFC2119], [RFC4513], [RFC4512], and [RFC4510] and updated the UTF-8 reference. Replaced RFC 822 reference with a reference to RFC 4234.

"Informative References" section: (new section) moved [X.690] to this section. Added a reference to [RFC4516].

"Acknowledgements" section: added.

"Appendix A: Changes Since RFC 2254" section: added.

Surrounded the names of all ABNF productions with "<" and ">" where they are used in descriptive text.

Replaced all occurrences of "LDAPv3" with "LDAP."

## Authors' Addresses

Mark Smith, Editor
Pearl Crescent, LLC
447 Marlpool Dr.
Saline, MI 48176
USA

Phone: +1 734 944-2856
EMail: mcs@pearlcrescent.com

Tim Howes
Opsware, Inc.
599 N. Mathilda Ave.
Sunnyvale, CA 94085
USA

Phone: +1 408 744-7509
EMail: howes@opsware.com

## Full Copyright Statement

## Intellectual Property

## Acknowledgement

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