版本信息

日期	版本	描述	作者
2004-01-30	v1.0	最初版本	LDAPChina.com

The String Representation of LDAP Search Filters

LDAP 查询过滤器的字符串表示法

1、本备忘录的状态（Status of this Memo）

本文档定义了一个用于 Internet 通讯的 Internet 标准跟踪协议，为了发展的需要讨论和建议。对于这个协议的状况和地位请参照 Internet 官方协议标准（"Internet Official Protocol Standards"（STD 1））的当前版。这个备忘录的传播是不受限制的。

1.1、版权提示（Copyright Notice）

1.2、IESG 提示（IESG Note）

本文档将描述一种同时提供读和更新访问的目录访问协议。更新访问需要安全认证，但这个文档并不强制实现任何安全认证机制。

与 RFC2026 的 4.4.1 节相同，本规范正在被 IESG 批准期间，作为被提议的标准，可能并不限于本文档所述内容。原因如下：

a、 鼓励在它们被发布前，实现和交互测试这些协议（带有或没有更新访问）；

b、 鼓励在只读的应用程序中配置和使用这些协议。（例如，在某些应用程序中，目录的更新访问使用某些其它安全的机制而不是 LDAP，而使用 LDAPv3 被作为对目录的查询语言）；

c、 避免阻碍别的 Internet 标准追踪协议的发展和发布。（这些协议需要 LDAPv3 的目录服务器的查询能力，而不是更新能力）

需要警告读者的是，直到强制的验证机制被标准化之前，根据本规范编写的客户端和服务器实现了更新功能的话，它们的互操作性可能是不可靠的，或者仅提供在认证需要极度弱化的时候的互操作性。

因此在具有强制认证的 LDAPv3 未成为一个 RFC 而被批准或发布之前，不鼓励实现者发布一个实现了更新功能的 LDAPv3 的客户端和服务器。

目　录

2、摘要（Abstract）

轻型目录访问协议（LDAP）参考文档[1]中定义了一种通过网络传输到 LDAP 服务器端的查询过滤器的表示法。人们可能想找一种用简单易懂的公式表示这些查询过滤器的方法来用到自己的应用中。本文档就为表示 LDAP 查询过滤器而定义了一种易懂的字符串格式。

这篇文档代替 RFC1960，并扩展了字符串 LDAP 过滤器的定义，其中包括对在 LDAPv3 中新增的匹配过滤器的提供了支持，还对可能在完全范围中进行查询的过滤器的表示法提供了支持。

3、LDAP 查询过滤器定义（LDAP Search Filter Definition）

参考文档[1]的第 4.5.1 节对 LDAPv3 查询过滤器有如下定义：

```
Filter ::= CHOICE {
        and                 [0] SET OF Filter,
        or                  [1] SET OF 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,
        SEQUENCE OF CHOICE {
                initial         [0] LDAPString,
                any             [1] LDAPString,
                final           [2] LDAPString
        }
}

AttributeValueAssertion ::= SEQUENCE {
        attributeDesc     AttributeDescription,
        attributeValue    AttributeValue
}
```

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

AttributeDescription ::= LDAPString

AttributeValue ::= OCTET STRING

MatchingRuleID ::= LDAPString

AssertionValue ::= OCTET STRING

LDAPString ::= OCTET STRING
```

上面的 LDAPString 拥有必须以 ISO10646 字符集（参考文档[4]）的 UTF-8 方式编码而成的限制。AttributeDescription 是属性描述的字符串表示法，在参考文档[1]中对它有详细的定义。参考文档[2]定义了 AtrributeValue 和 AssertionValue 的八进制字符串（OCTET STRING）。通过网络传输的过滤器用参考文档[3]中定义的基本编码规则（BER）编码。基本编码规则在参考文档[1]中有简要的描述。

4、字符串查询过滤器定义（String Search Filter Definition）

下面的语法定义了 LDAP 查询过滤器的字符串表示法，其中的 ABNF 符号在参考文档[5]中有相关定义。过滤器的格式使用了前缀符号。

```
filter          = "(" filtercomp ")"
filtercomp      = and / or / not / item
and             = "&" filterlist
or              = "|" filterlist
not             = "!" filter
filterlist      = 1*filter
item            = simple / present / substring / extensible
simple          = attr filtertype value
filtertype      = equal / approx / greater / less
equal           = "="
approx          = "~="
greater         = ">="
less            = "<="
extensible      = attr [":dn"] [":" matchingrule] ":=" value
```

```
                    / [":dn"] ":" matchingrule ":=" value
present       = attr "=*"
substring     = attr "=" [initial] any [final]
initial       = value
any           = "*" *(value "*")
final         = value
attr          = AttributeDescription from Section 4.1.5 of [1]
matchingrule = MatchingRuleId from Section 4.1.9 of [1]
value         = AttributeValue from Section 4.1.6 of [1]
```

参考文档[1]中相应的章节描述了 attr、matchingrule、和 value 的结构。

字符	ASCII 码
*	0x2a
(	0x28
)	0x29
\	0x5c
NUL	0x00

如果 value 中包含了上面字符中的任意一个，那么这个字符的编码必须以反斜杠(ASCII 0x5c)开头，后面跟着代替其 ASCII 码的两个十六进制的符号。这两个十六进制的符号的大小写无关紧要。

这种简单的转义机制消除了过滤器解析时的模糊，并且可以把任何过滤器表示成 LDAP 中以 NUL 结尾的字符串。除了这张表中的字符之外的其他字符也可以用这种机制转义，例如说非打印字符。

例如说，过滤器检查“cn”属性的值中是否包含字符“*”，它可以用“(cn=*\2a*)”来表示。

注意，尽管从语法上说 substring 和 present 的结果中也能产生出“attr=*”这样的结构，但这种结构只能被用来表示一个已存在的过滤器。

5、示例（Examples）

这一节用上述的表达法写了几个查询过滤器的示例。

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

下面的例子表示了可扩展匹配的用法。

```
(cn:1.2.3.4.5:=Fred Flintstone)
```

(sn:dn:2.4.6.8.10:=Barney Rubble)
(o:dn:=Ace Industry)
(:dn:2.4.6.8.10:=Dino)

第二个例子表示的是“:dn”的用法，指出当进行比较时，应该使用匹配规则“2.4.6.8.10”，并且在评估这个匹配时，一个条目的分辨名属性应该被考虑成这个条目的一部分。

第三个例子表示了一个相等匹配，在进行匹配时 DN 部分应当被作为条目的一部分被考虑

第四个例子是，一个过滤器应该被应用到任意一个支持给定的匹配规则的属性中（因为 attr 没有出现）。也应该考虑将支持匹配规则的属性保存在 DN 中。

下面的例子表示的是转义机制的使用。

(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)

第一个例子显示了转义机制如何用在表示括号字符时。第二个例子显示了如何在一个 value 里表示出“*”，从而防止它被认为是一个 substring 的代替。第三个例子显示了如何转义一个反斜杠。

第四个例子显示了一个 value 为“0x00000004”四字节的查询过滤器，指出如何用转义机制表示任意数据，例如说 NUL 字符。

最后一个例子显示了如何用转义机制表示各种各样的非 ASCII 码的 UTF-8 字符。

6、安全考虑（Security Considerations）

这个备忘录描述的是 LDAP 查询过滤器的字符串表示法。这个表示法本身没有已知的安全隐患，但 LDAP 查询过滤器有。它们被 LDAP 服务器解析，从重新得到的数据中选择条目。LDAP 服务器应该仔细保护它们维护的数据，不能让没有经过授权的访问获得这些数据。

7、参考书目（References）

[1] Wahl, M., Howes, T., and S. Kille, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[2] Wahl, M., Coulbeck, A., Howes, T., and S. Kille, "Lightweight Directory Access Protocol

(v3): Attribute Syntax Definitions", RFC2252, December 1997.

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

[4] Yergeau, F., "UTF-8, a transformation format of Unicode and ISO10646", RFC 2044, October 1996.

[5] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", STD 11, RFC 822, August 1982.

8、作者地址（Authors' Addresses）

Tim Howes
Netscape Communications Corp.
501 E. Middlefield Road
Mountain View, CA 94043
USA
Phone: +1 415 937-3419
EMail: mailto:howes@netscape.com

9、完整的版权声明（Full Copyright Statement）

略