Internet-Draft
LDAPExt WG
Intended Category: Standards Track

Expires: 26 December 2002

David Chadwick
University of Salford
Sean Mullan
Sun Microsystems
26 June 2002

Returning Matched Values with LDAPv3
<draft-ietf-ldapext-matchedval-06.txt>

STATUS OF THIS MEMO

Comments and suggestions on this document are encouraged. Comments on this document should be sent to the LDAPEXT working group discussion list:

ietf-ldapext@netscape.com

or directly to the authors.

ABSTRACT

This document describes a control for the Lightweight Directory Access Protocol version 3 that is used to return a subset of attribute values from an entry, specifically, only those values that match a "values return" filter. Without support for this control, a client must retrieve all of an attribute's values and search for specific values locally.

1. Introduction

When reading an attribute from an entry using the Lightweight Directory Access Protocol version 3 (LDAPv3) [2], it is normally only possible to read either the attribute type, or the attribute type and all its values. It is not possible to selectively read just a few of the attribute values. If an attribute holds many values, for example, the userCertificate attribute, or the subschema publishing operational attributes objectClasses and attributeTypes [3], then it may be desirable for the user to be able to selectively retrieve a subset of the values, specifically, those attribute values that match some user defined selection criteria. Without the control specified in this document a client must read all of the attribute's values and filter out the unwanted values, necessitating the client to implement the matching rules. It also requires the client to potentially read and process many irrelevant values, which can be inefficient if the values are large or complex, or there are many values stored per attribute.

This document specifies an LDAPv3 control to enable a user to return only those values that matched (i.e. returned TRUE to) one or more elements of a newly defined "values return" filter. This control can be especially useful when used in conjunction with extensible matching rules that match on one or more components of complex binary attribute values.

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 RFC 2119 [4].

2. The valuesReturnFilter Control

The valuesReturnFilter control is either critical or non-critical as determined by the user. It only has meaning for the Search operation, and SHOULD only be added to the Search operation by the client. If the server supports the control and it is present on a Search operation, the server MUST obey the control regardless of the value of the criticality flag.

If the control is marked as critical, and either the server does not support the control or the control is applied to an operation other than Search, then the server MUST return an unavailableCriticalExtension error. If the control is not marked as critical, and either the server does not support the control or the control is applied to an operation other than Search, then the server MUST ignore the control.

The object identifier for this control is 1.2.826.0.1.3344810.2.3.

The controlValue is an OCTET STRING, whose value is the BER encoding, as per Section 5.1 of RFC 2251 [2], of a value of the type ValuesReturnFilter.

```
        ValuesReturnFilter ::= SEQUENCE OF SimpleFilterItem

        SimpleFilterItem ::= CHOICE {
                equalityMatch   [3] AttributeValueAssertion,
                substrings      [4] SubstringFilter,
                greaterOrEqual  [5] AttributeValueAssertion,
                lessOrEqual     [6] AttributeValueAssertion,
                present         [7] AttributeDescription,
                approxMatch     [8] AttributeValueAssertion,
                extensibleMatch [9] SimpleMatchingAssertion }

         SimpleMatchingAssertion ::= SEQUENCE {
                matchingRule    [1] MatchingRuleId OPTIONAL,
                type            [2] AttributeDescription OPTIONAL,
--- at least one of the above must be present
                matchValue      [3] AssertionValue}
```

All the above data types have their standard meanings as defined in [2].

If the server supports this control, the server MUST make use of the control as follows:

(1) The Search Filter is first executed in order to determine which entries satisfy the Search criteria (these are the filtered entries). The control has no impact on this step.

(2) If the typesOnly parameter of the Search Request is TRUE, the control has no effect and the Search Request is processed as if the control had not been specified.

(3) If the attributes parameter of the Search Request consists of a list containing only the attribute with OID "1.1" (specifying that no attributes are to be returned), the control has no effect and the Search Request is processed as if the control had not been specified.

(4) For each attribute listed in the attributes parameter of the Search Request, the server MUST apply the control as follows to each entry in the set of filtered entries:

i)   Every attribute value that evaluates TRUE against one or more elements of the ValuesReturnFilter is placed in the corresponding SearchResultEntry.
ii)   Every attribute value that evaluates FALSE or undefined against all elements of the ValuesReturnFilter is not placed in the corresponding SearchResultEntry. An attribute that has no values selected is returned with an empty set of vals.

Note. If the AttributeDescriptionList is empty or comprises "*" then the control MUST be applied against every user attribute. If the AttributeDescriptionList contains a "+" then the control MUST be applied against every operational attribute.

3. Relationship to X.500

The control is a superset of the matchedValuesOnly (MVO) boolean of the X.500 Directory Access Protocol (DAP) [5] Search argument, as amended in the latest version [6]. Close examination of the matchedValuesOnly boolean by the LDAP Extensions (LDAPEXT) Working Group revealed ambiguities and complexities in the MVO boolean that could not easily be resolved. For example, it was not clear if the MVO boolean governed only those attribute values that contributed to the overall truth of the filter, or all of the attribute values even if the filter item containing the attribute evaluated to false. For this reason the LDAPEXT group decided to replace the MVO boolean with a simple filter that removes any uncertainty as to whether an attribute value has been selected or not.

4. Relationship to other LDAP Controls

The purpose of this control is to select zero, one or more attribute values from each requested attribute in a filtered entry, and to discard the remainder. Once the attribute values have been discarded by this control they MUST NOT be re-instated into the Search results by other controls.

This control acts independently of other LDAP controls such as server side sorting [10] and duplicate entries [7]. However, there might be interactions between this control and other controls so that a different set of Search Result Entries are returned, or the entries are returned in a different order, depending upon the sequencing of this control and other controls in the LDAP request. For example, with server side sorting, if sorting is done first, and value return filtering second, the set of Search Results may appear to be in the wrong order since the value filtering may remove the attribute values upon which the ordering was done. (The sorting document specifies that entries without any sort key attribute values should be treated as coming after all other attribute values.) Similarly with duplicate entries, if duplication is performed before value filtering, the set of Search Result Entries may contain identical duplicate entries, each with an empty set of attribute values, because the value filtering removed the attribute values that were used to duplicate the results.

For these reasons the ValuesReturnFilter control in a SearchRequest SHOULD precede other controls that affect the number and ordering of SearchResultEntrys.

5. Examples

All entries are provided in LDAP Data Interchange Format (LDIF)[8].

The string representation of the valuesReturnFilter in the examples below uses the following ABNF [12] notation:

```
 valuesReturnFilter = "(" 1*simpleFilterItem ")"
 simpleFilterItem = "(" item ")"
```

where item is as defined below (adapted from RFC2254 [11]).

```
          item       = simple / present / substring / extensible
        simple     = attr filtertype value
        filtertype = equal / approx / greater / less
        equal      = "="
        approx     = "~="
        greater    = ">="
        less       = "<="
        extensible = attr [":" matchingrule] ":=" value
                     / ":" 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]
```

(1) The first example shows how the control can be set to return all attribute values from one attribute type (e.g. telephoneNumber) and a subset of values from another attribute type (e.g. mail).

The entries below represent organizationalPerson object classes located somewhere beneath the distinguished name dc=ac,dc=uk.

```
dn: cn=Sean Mullan,ou=people,dc=sun,dc=ac,dc=uk
```

```
cn: Sean Mullan
sn: Mullan
objectClass: organizationalPerson
objectClass: person
objectClass: inetOrgPerson
mail: sean.mullan@hotmail.com
mail: mullan@east.sun.com
telephoneNumber: + 781 442 0926
telephoneNumber: 555-9999

dn: cn=David Chadwick,ou=isi,o=salford,dc=ac,dc=uk
cn: David Chadwick
sn: Chadwick
objectClass: organizationalPerson
objectClass: person
objectClass: inetOrgPerson
mail: d.w.chadwick@salford.ac.uk
```

An LDAP search operation is specified with a baseObject set to the DN of the search base (i.e. dc=ac,dc=uk), a subtree scope, a filter set to (sn=mullan), and the list of attributes to be returned set to "mail,telephoneNumber". In addition, a ValuesReturnFilter control is set to ((mail=*hotmail.com)(telephoneNumber=*))

The search results returned by the server would consist of the following entry:

```
dn: cn=Sean Mullan,ou=people,dc=sun,dc=ac,dc=uk
mail: sean.mullan@hotmail.com
telephoneNumber: + 781 442 0926
telephoneNumber: 555-9999
```

Note that the control has no effect on the values returned for the "telephoneNumber" attribute (all of the values are returned), since the control specified that all values should be returned.

(2) The second example shows how one might retrieve a single attribute type subschema definition for the "gunk" attribute with OID 1.2.3.4.5 from the subschema subentry

Assume the subschema subentry is held below the root entry with DN cn=subschema subentry,o=myorg and this holds an attributeTypes operational attribute holding the descriptions of the 35 attributes known to this server (each description is held as a single attribute value of the attributeTypes attribute).

```
dn: cn=subschema subentry,o=myorg
cn: subschema subentry
objectClass: subschema
attributeTypes: ( 2.5.4.3 NAME 'cn' SUP name )
attributeTypes: ( 2.5.4.6 NAME 'c' SUP name SINGLE-VALUE )
attributeTypes: ( 2.5.4.0 NAME 'objectClass' EQUALITY
 objectIdentifierMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
attributeTypes: ( 2.5.18.2 NAME 'modifyTimestamp' EQUALITY
 generalizedTimeMatch ORDERING generalizedTimeOrderingMatch
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.24 SINGLE-VALUE NO-USER-
 MODIFICATION USAGE directoryOperation )
attributeTypes: ( 2.5.21.6 NAME 'objectClasses' EQUALITY
 objectIdentifierFirstComponentMatch SYNTAX
 1.3.6.1.4.1.1466.115.121.1.37 USAGE directoryOperation )
```

```
attributeTypes: ( 1.2.3.4.5 NAME 'gunk' EQUALITY caseIgnoreMatch
 SUBSTR caseIgnoreSubstringsMatch SYNTAX
 1.3.6.1.4.1.1466.115.121.1.44{64} )
attributeTypes: ( 2.5.21.5 NAME 'attributeTypes' EQUALITY
 objectIdentifierFirstComponentMatch SYNTAX
 1.3.6.1.4.1.1466.115.121.1.3 USAGE directoryOperation )
```

plus another 28 - you get the idea.

The user creates an LDAP search operation with a baseObject set to cn=subschema subentry,o=myorg, a scope of base, a filter set to (objectClass=subschema), the list of attributes to be returned set to "attributeTypes", and the ValuesReturnFilter set to ((attributeTypes=1.2.3.4.5))

The search result returned by the server would consist of the following entry:

```
dn: cn=subschema subentry,o=myorg
attributeTypes: ( 1.2.3.4.5 NAME 'gunk' EQUALITY caseIgnoreMatch
 SUBSTR caseIgnoreSubstringsMatch SYNTAX
 1.3.6.1.4.1.1466.115.121.1.44{64} )
```

(3) The final example shows how the control can be used to match on a userCertificate attribute value. Note that this example requires the LDAP server to support the certificateExactMatch matching rule defined in [9].

The entry below represent a pkiUser object class stored in the directory.

```
dn: cn=David Chadwick+serialNumber=123456,ou=people,o=University
 of Salford,c=gb
cn: David Chadwick
serialNumber: 123456
objectClass: person
objectClass: organizationalPerson
objectClass: pkiUser
objectClass: inetOrgPerson
sn: Chadwick
mail: d.w.chadwick@salford.ac.uk
userCertificate: {binary representation of a certificate with a
serial number of 2468 issued by o=truetrust ltd, c=gb}
userCertificate: {binary representation of certificate with a serial
number of 1357 issued by o=truetrust ltd, c=gb}
userCertificate: {binary representation of certificate with a serial
number of 1234 issued by dc=certs R us, dc=com}
```

An LDAP search operation is specified with a baseObject set to o=University of Salford,c=gb, a subtree scope, a filter set to (sn=chadwick) and the list of attributes to be returned set to "userCertificate". In addition, a ValuesReturnFilter control is set to (userCertificate=1357$o=truetrust ltd, c=gb)

The search result returned by the server would consist of the following entry:

```
dn: cn=David Chadwick+serialNumber=123456,ou=people,o=University
 of Salford,c=gb
```

userCertificate;binary: {binary representation of certificate with a serial number of 1357 issued by o=truetrust ltd, c=gb}

6. Security Considerations

This document does not primarily discuss security issues.

Note however that attribute values MUST only be returned if the access controls applied by the LDAP server allow them to be returned, and in this respect the effect of the ValuesReturnFilter control is of no consequence.

Note that the ValuesReturnFilter control may have a positive effect on the deployment of public key infrastructures. Certain PKI operations, like searching for specific certificates, become more practical when combined with X.509 certificate matching rules at the server, and more scalable, since the control avoids the downloading of potentially large numbers of irrelevant certificates which would have to be processed and filtered locally (which in some cases is very difficult to perform).

7. Acknowledgements

The authors would like to thank members of the LDAPExt list for their constructive comments on earlier versions of this document, and in particular to Harald Alvestrand who first suggested having an attribute return filter and Bruce Greenblatt who first proposed a syntax for this control.

8. Copyright

9. References

Normative

[1] S. Bradner. "The Internet Standards Process -- Revision 3", RFC 2026, October 1996.
[2] M. Wahl, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3)", Dec. 1997, RFC 2251
[3] M. Wahl, A. Coulbeck, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, Dec 1997
[4] S.Bradner. "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

Informative

[5] ITU-T Rec. X.511, "The Directory: Abstract Service Definition", 1993.
[6] Draft ISO/IEC 9594 / ITU-T Rec X.511 (2001) The Directory: Abstract Service Definition.
[7] J. Sermersheim. "LDAP Control for a Duplicate Entry Representation of Search Results", Internet Draft <draft-ietf-ldapext-ldapv3-dupent-06.txt>, October 2000.
[8] G. Good. "The LDAP Data Interchange Format (LDIF) - Technical Specification". RFC 2849, June 2000.
[9] D. Chadwick, S.Legg. "Internet X.509 Public Key Infrastructure - Additional LDAP Schema for PKIs", Internet Draft <draft-pkix-ldap-pki-schema-00.txt>, June 2002
[10] T. Howes, M. Wahl, A. Anantha, "LDAP Control Extension for Server Side Sorting of Search Results", RFC 2891, August 2000
[11] T. Howes. "The String Representation of LDAP Search Filters". RFC 2254, December 1997.
[12] D. Crocker, Ed. "Augmented BNF for Syntax Specifications: ABNF." RFC 2234. November 1997.

10. Authors Addresses

David Chadwick
IS Institute
University of Salford
Salford M5 4WT
England

Email: d.w.chadwick@salford.ac.uk
Tel: +44 161 295 5351

Sean Mullan
Sun Microsystems
East Point Business Park
Dublin 3
Ireland
Tel: +353 1 853 0655
Email: sean.mullan@sun.com

11. Changes since version 2

i) Revised the examples to be more appropriate
ii) Section on interactions with other LDAP controls added
iii) Removed Editor's note concerning present filter

iv)   Tightened wording about its applicability to other operations and use of criticality field

Changes since version 3

i)   Mandated that at least one of type and matchingRule in simpleMatchingAssertion be present
ii)   Fixed LDIF mistakes in the examples
iii)   Additional minor editorials only

Changes since version 4

i)   corrected the ABNF for single items of valuesReturnFilter

Changes since version 5

i)   added some adapted BNFL from [11] into the examples (specifically the [":dn"] component was removed)
ii)   general editorial tidying up prior to Last Call