Network Working Group
Request for Comment: 2657
Category: Experimental

R. Hedberg
Catalogix

August 1999

LDAPv2 Client vs. the Index Mesh

Status of this Memo

Copyright Notice

Abstract

LDAPv2 clients as implemented according to RFC 1777 [1] have no notion on referral. The integration between such a client and an Index Mesh, as defined by the Common Indexing Protocol [2], heavily depends on referrals and therefore needs to be handled in a special way. This document defines one possible way of doing this.

1. Background

During the development of the Common Indexing Protocol (CIP), one of the underlying assumptions was that the interaction between clients and the Index Mesh Servers [1] would heavily depend on the passing of referrals. Protocols like LDAPv2 [2] that lack this functionality need to compensate for it by some means. The way chosen in this memo is to add more intelligence into the client. There are two reasons behind this decision. First, this is not a major enhancement that is needed and secondly, that the intelligence when dealing with the Index Mesh, with or the knowledge about referrals, eventually has to go into the client.

2. The clients view of the Index Mesh

If a LDAPv2 client is going to be able to interact with the Index Mesh, the Mesh has to appear as something that is understandable to the client. Basically, this consists of representing the index servers and their contained indexes in a defined directory information tree (DIT) [3,4] structure and a set of object classes and attribute types that have been proven to be useful in this context.

2.1 The CIP Object Classes

Object class descriptions are written according to the BNF defined in [5].

2.1.1 cIPIndex

The cIPIndex objectClass, if present in a entry, allows it to hold one indexvalue and information connected to this value.

```
( 1.2.752.17.3.9
  NAME 'cIPIndex'
  SUP 'top'
  STRUCTURAL
  MUST ( extendedDSI $ idx )
  MAY ( indexOCAT )
)
```

2.1.2 cIPDataSet

The cIPDataSet objectClass, if present in a entry, allows it to hold information concerning one DataSet.

```
( 1.2.752.17.3.10
  NAME 'cIPDataSet'
  SUP 'top'
  STRUCTURAL
  MUST ( dSI $ searchBase )
  MAY ( indexOCAT $ description $ indexType $
        accessPoint $ protocolVersion $ polledBy $
        updateIntervall $ securityOption $
        supplierURI $ consumerURI $ baseURI $
        attributeNamespace $ consistencyBase
    )
)
```

2.2 The CIP attributeTypes

The attributes idx, indexOCAT, extendedDSI, description, cIPIndexType, baseURI, dSI are used by a client accessing the index server. The other attributes (accesspoint, protocolVersion, polledBy, updateIntervall, consumerURI, supplierURI and securityOption, attributeNamespace, consistencyBase) are all for usage in server to server interactions.

2.2.1 idx

The index value, normally used as part of the RDN.

```
( 1.2.752.17.1.20
  NAME 'idx'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
  SINGLE-VALUE
   )
```

2.2.2 dSI

DataSet Identifier, a unique identifier for one particular set of information. This should be an OID, but stored in a stringformat.

```
( 1.2.752.17.1.21
  NAME 'dSI'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
)
```

2.2.3 indexOCAT

Describes the type of data that is stored in this entry, by using objectcClasses and attributeTypes. The information is stored as a objectClass name followed by a space and then an attributeType name. A typical example when dealing with whitepages information would be "person cn".

```
( 1.2.752.17.1.28
  NAME 'indexOCAT'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
)
```

2.2.5 supplierURI

A URI describing which protocols, hostnames and ports should be used by an indexserver to interact with servers carrying indexinformation representing this dataSet.

```
  ( 1.2.752.17.1.22
   NAME 'supplierURI'
   EQUALITY caseIgnoreIA5Match
   SYNTAX IA5String
)
```

2.2.6 baseURI

The attribute value for this attribute is a LDAP URI. One can envisage other URI syntaxes, if the client knows about more access protocols besides LDAP, and the interaction between the client and the server can not use referrals for some reason.

```
( 1.2.752.17.1.26
  NAME 'baseURI'
  EQUALITY caseExactIA5Match
  SYNTAX IA5String
)
```

2.2.7 protocolVersion

At present, the Common Indexing Protocol version should be 3.

```
( 1.2.752.17.1.27
  NAME 'protocolVersion'
  EQUALITY numericStringMatch
  SYNTAX numericString
)
```

2.2.8 cIPIndexType

The type of index Object that is used to pass around index information.

```
( 1.2.752.17.1.29
  NAME 'cIPIndexType'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
)
```

2.2.10 polledBy

The Distinguished Name of Index servers that polls data from this indexserver.

```
( 1.2.752.17.1.30
  NAME 'polledBy'
  EQUALITY distinguishedNameMatch
  SYNTAX DN
)
```

2.2.11 updateIntervall

The maximum duration in seconds between the generation of two updates by the supplier server.

```
( 1.2.752.17.1.31
  Name 'updateIntervall'
  EQUALITY numericStringMatch
  SYNTAX numericString
  SINGLE-VALUE
)
```

2.2.12 securityOption

Whether and how the supplier server should sign and encrypt the update before sending it to the consumer server.

```
( 1.2.752.17.1.32
  NAME 'securityOption'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
  SINGLE-VALUE
)
```

2.2.13 extendedDSI

DataSet Identifier possibly followed by a space and a taglist, the later as specified by [6].

```
( 1.2.752.17.1.33
  NAME 'extendedDSI'
  EQUALITY caseIgnoreIA5Match
  SYNTAX IA5String
     )
```

2.2.14 consumerURI

A URI describing which means a server can accept indexinformation. An example being a mailto URI for MIME email based index transport.

```
( 1.2.752.17.1.34
  NAME 'consumerURI'
  EQUALITY caseExactIA5Match
  SYNTAX IA5String
)
```

2.2.15 attributeNamespace

Any consumer supplier pair has to agree on what attribute that should be used and also possibly the meaning of the attributenames. The value of this attribute should, for example, be a URI pointing to a document wherein the agreement is described.

```
( 1.2.752.17.1.35 NAME 'attributeNamespace' EQUALITY
  caseExactIA5Match SYNTAX IA5String
)
```

2.2.16 consistencyBase

This attribute is specifically used by consumer supplier pairs that use the tagged index object [6].

```
( 1.2.752.17.1.36
  NAME 'consistencyBase'
  EQUALITY caseExactIA5Match
  SYNTAX IA5String
)
```

3. The interaction between a client and the Index Mesh

A client interaction with the Index Mesh consists of a couple of rather well defined actions. The first being to find a suitable index to start with, then to transverse the Index Mesh and finally to query the servers holding the original data. Note when reading this text that what is discussed here is the client's perception of the DIT, how it is in fact implemented is not discussed.

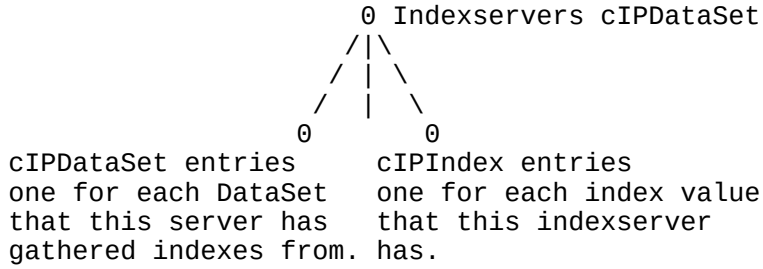
3.1 Finding a Index Mesh

This approach depends on the fact that every index server partaking in an Index Mesh is represented in the DIT by a entry of the type cIPDataSet, and has a distinguished name (DN) which most significant relative distinguished name (RDN) has the attributetype dSI. Therefore, finding a suitable indexserver to start the search from is a matter of searching the DIT at a suitable place for objects with the objectClass cIPIndexObject. Every found entry can then be evaluated by looking at the description value as well as the indexOCAT value. The description string should be a human readable and understandable text that describes what the index server is indexing. An example of such a string could be, "This index covers all employees at Swedish Universities and University Colleges that has an email account". The indexOCAT attribute supplies information about which kind of entries and which attributes within these entries that the index information has emanated from. For example, if the

indexOCAT attribute value is "person cn", one can deduce that this is an index over persons and not over roles, and that it is the attribute commonName that is indexed.

3.2 Searching the mesh

Each index server has its information represented in the DIT as a very flat tree. In fact, it is only one level deep.



A search then consists of a set of searches. The first being the search for the index entries that contains an indexvalue that matches what the user is looking for, and the second a search based on the DSI information in the extendedDSI attribute values returned from the first search. In the case of the the cIPIndexType being tagged-index, the taglists should be compared to find which DSI it might be useful to pose further queries to.

When doing these types of searches, the client should be aware of the fact that the index values disregarding their origin (attributeTypes) always are stored in the index server as values of the idx attribute.

The object of the second search is to get information on the different DataSet involved, and should normally be performed as a read. Since the DataSet information probably will remain quite stable over time, this information lends itself very well to caching. If at this stage there is more than one DataSet involved, the User interface might use the description value to aid the user in choosing which one to proceed with. The content of the searchBase value of the DataSet tells the client whether it represents another index server (the most significant part of the dn is a dSI attribute) or if it is a end server.

3.3 Querying the end server

When finally reaching the end server/servers that probably has the sought for information, the information in the indexOCAT attribute can be used to produce an appropriate filter. If a search for "Rol*" in an index having an indexOCAT attribute value of "person cn" returns an idx entry with the idx value of "Roland", then an appropriate filter to use might be "&(|(cn=* roland *)(cn=roland *)(cn=* roland))(objectclass=person)". A complete example of a search process is given in Appendix A.

4. Security Considerations

Since this memo deals with client behavior, it does not add anything that either enhances or diminishes the security features that exists in LDAPv2.

5. Internationalization

As with security, this memo neither enhances or diminishes the handling of internationalization in LDAPv2.

6. References

[1] Yeong, W., Howes, T. and S. Kille, "Lightweight Directory Access Protocol", RFC 1777, March 1995.

[2] Allen, J. and M. Mealling "The Architecture of the Common Indexing Protocol (CIP)", RFC 2651, August 1999.

[3] The Directory: Overview of Concepts, Models and Service. CCITT Recommendation X.500, 1988.

[4] Information Processing Systems -- Open Systems Interconnection -- The Directory: Overview of Concepts, Models and Service. ISO/IEC JTC 1/SC21; International Standard 9594-1, 1988.

[5] Wahl, M., Coulbeck, A., Howes, T. and S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

[6] Hedberg, R., Greenblatt, B., Moats, R. and M. Wahl, "A Tagged Index Object for use in the Common Indexing Protocol", RFC 2654, August 1999.

7. Author's Address

Roland Hedberg
Catalogix
Dalsveien 53
0387 Oslo, Norway

Phone: +47 23 08 29 96
EMail: roland@catalogix.ac.se

Appendix A - Sample Session

Below is a sample of a session between a LDAPv2 client and an index server mesh as specified in this memo.

The original question of the session is to find the email address of a person by the name, "Roland Hedberg", who is working at "Umea University" in Sweden.

Step 1.

A singlelevel search with the baseaddress "c=SE" and the filter "(objectclass=cipDataset)" was issued.

The following results were received:

```
DN: dSI=1.2.752.17.5.0,c=SE
dsi= 1.2.752.17.5.0
description= "index over employees with emailaddresses within Swedish
higher education"
indexOCAT= "cn person"
cIPIndexType= "x-tagged-index-1" ;
searchBase= "dsi=1.2.752.17.5.0,c=SE"
protocolVersion = 3

DN: dSI=1.2.752.23.1.3,c=SE
dsi= 1.2.752.23.1.3
description= "index over Swedish lawyers"
indexOCAT= "cn person"
cIPIndexType= "x-tagged-index-1" ;
searchBase= "dsi=1.2.752.23.1.3,c=SE"
protocolVersion = 3
```

Step 2.

Since the first index seemed to cover the interesting population, a single level search with the baseaddress "dsi=1.2.752.17.5.0,c=SE" and the filter "(|(idx=roland)(idx=hedberg))" was issued.

The following results were received:

```
DN: idx=Roland,dSI=1.2.752.17.5.0,c=SE
idx= Roland
extendedDSI= 1.2.752.17.5.10 1,473,612,879,1024
extendedDSI= 1.2.752.17.5.14 35,78,150,200
extendedDSI= 1.2.752.17.5.16 187,2031,3167,5284,6034-6040
extendedDSI= 1.2.752.17.5.17 17
```

```
DN: idx=Hedberg,dSI=1.2.752.17.5.0,c=SE
idx= Hedberg
extendedDSI= 1.2.752.17.5.8  24,548-552,1066
extendedDSI= 1.2.752.17.5.10 473,512,636,777,1350
extendedDSI= 1.2.752.17.5.14 84,112,143,200
extendedDSI= 1.2.752.17.5.15 1890-1912
extendedDSI= 1.2.752.17.5.17 44
```

A comparison between the two sets of extendedDSIs shows that two datasets 1.2.752.17.5.10 and 1.2.752.17.5.14 contains persons named "Roland" and "Hedberg". Therefore, the next step would be to see what the datasets represent. A comparison like this should normally not be left to the user.

Step. 3

Two baselevel searches, one for "dsi=1.2.752.17.5.10,dsi=1.2.752.17.5.0,c=SE" and the other for "dsi=1.2.752.17.5.14,dsi=1.2.752.17.5.0,c=SE" with the filter "(objectclass=cipdataset)" were issued.

The following results were received:

```
DN: dSI=1.2.752.17.5.10,dSI=1.2.752.17.5.0,c=SE
dsi= 1.2.752.17.5.10
description= "Employees at Umea University,Sweden"
indexOCAT= "person cn"
searchBase= "o=Umea Universitet,c=SE"
```

respectively

```
DN: dSI=1.2.752.17.5.14,dSI=1.2.752.17.5.0,c=SE
dsi= 1.2.752.17.5.14
description= "Employees at Lund University,Sweden"
indexOCAT= "person cn"
searchBase= "o=Lunds Universitet,c=SE"
```

Step 4

Based on the descriptions for the two datasets, "1.2.752.17.5.10" was chosen as the best to proceed with. From the searchbase attribute value, it was clear that this was a base server. The query now has to be somewhat modified. One possibility would be to issue a query with the baseobject "o=Umea Universitet,c=SE" and the filter "(&(cn=Roland Hedberg)(objectclass=person))"

Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.