A Schema for Logging the LDAP Protocol

draft-chu-ldap-logschema-00.txt

Status of this Memo

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/ietf/1id-abstracts.txt.

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

This Internet-Draft will expire on November 4, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

In order to facilitate remote administration and auditing of LDAP server operation, it is desirable to provide the server’s operational logs themselves as a searchable LDAP directory. These logs may also be used as a persistent change log to support various replication mechanisms. This document defines a schema that may be used to represent all of the requests that have been processed by an LDAP server. It may be used by various applications for auditing, flight recorder, replication, and other purposes.
# Table of Contents

1. Introduction ............................................. 3  
2. Conventions ............................................. 4  
3. Syntaxes .................................................. 5  
3.1. Control Syntax ........................................ 5  
4. Attribute Types ......................................... 6  
4.1. General Attribute Types ............................... 6  
4.2. Request-specific Attribute Types ................... 7  
5. Object Classes .......................................... 10  
5.1. Basic Audit Object Classes ......................... 10  
5.2. Request-Specific Object Classes ................... 10  
5.3. Generic Container Class ............................. 11  
6. Discussion of Schema .................................... 12  
6.1. AuditObject ........................................... 12  
6.2. AuditContainer ........................................ 13  
6.3. Request-Specific Discussion ....................... 13  
7. Examples .................................................. 16  
7.1. Audit Trail ............................................ 16  
7.2. Add request ............................................ 17  
7.3. Modify request ........................................ 17  
7.4. Rename request ........................................ 18  
7.5. Delete request ......................................... 18  
7.6. Usage Notes ............................................ 19  
8. Security Considerations ................................. 20  
9. Normative References .................................... 20  
Appendix A. IANA Considerations ......................... 22  
Author’s Address ........................................... 23  
Intellectual Property and Copyright Statements ........ 24
1. Introduction

In a widely distributed network with multiple LDAP servers, it is desirable to be able to audit and monitor the operation of each server remotely, using the same tools that are normally used to interact with the LDAP servers. Using a standardized logging format in LDAP allows LDAP queries to be used to generate server usage statistics with little effort. This document describes a set of object classes that can be used to represent any LDAP operation. The object classes are intended to represent a complete record of all of the parameters of an operation. The log not only allows clients to see what operations were executed on a given server, but also to easily regenerate and re-issue a sequence of operations to aid in testing situations. The sequence of write operations recorded in the log can also be used by various replication mechanisms.
2. Conventions

Imperative keywords defined in [RFC2119] are used in this document, and carry the meanings described there.
3. Syntaxes

3.1. Control Syntax

A value of the Control syntax represents an LDAP Control as used by a client or server. It consists of the numeric OID of the Control, the Boolean criticality flag, and an optional OctetString containing the Control value. The definition given here merely repeats the definition of Controls in [RFC2251].

The Abstract Syntax Notation One (ASN.1 [X680]) definition of this syntax is as follows:

```
Control ::= SEQUENCE {
    controlType LDAPOID,
    criticality BOOLEAN DEFAULT FALSE,
    controlValue OCTET STRING OPTIONAL }
```

The following is an LDAP syntax description [RFC2252] suitable for publication in the subschema.

```
( LOG_SCHEMA_SYN.1 DESC 'Control' )
```
4. Attribute Types

4.1. General Attribute Types

These attributes are common to all of the LDAP request records.

( LOG_SCHEMA_AT.1 NAME 'reqDN'
  DESC 'Target DN of request'
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE )

( LOG_SCHEMA_AT .2 NAME 'reqStart'
  DESC 'Start time of request'
  EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE )

( LOG_SCHEMA_AT .3 NAME 'reqEnd'
  DESC 'End time of request'
  EQUALITY generalizedTimeMatch
  ORDERING generalizedTimeOrderingMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
  SINGLE-VALUE )

( LOG_SCHEMA_AT .4 NAME 'reqType'
  DESC 'Type of request'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( LOG_SCHEMA_AT .5 NAME 'reqSession'
  DESC 'Session ID of request'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( LOG_SCHEMA_AT .6 NAME 'reqAuthzID'
  DESC 'Authorization ID of requestor'
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE )

( LOG_SCHEMA_AT .7 NAME 'reqResult'
  DESC 'Result code of request'
  EQUALITY integerMatch
  ORDERING integerOrderingMatch)
4.2. Request-specific Attribute Types

These attributes are specific to a single type of LDAP request.

( LOG_SCHEMA_AT .12 NAME 'reqId'
DESC 'ID of Request to Abandon'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )

( LOG_SCHEMA_AT .13 NAME 'reqVersion'
DESC 'Protocol version of Bind request'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )

( LOG_SCHEMA_AT .14 NAME 'reqMethod'
DESC 'Bind method of request'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE )
( LOG_SCHEMA_AT .15 NAME 'reqAssertion'
  DESC 'Compare Assertion of request'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( LOG_SCHEMA_AT .16 NAME 'reqMod'
  DESC 'Modifications of request'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  EQUALITY octetStringMatch
  SUBSTR octetStringSubstringsMatch )

( LOG_SCHEMA_AT .17 NAME 'reqOld'
  DESC 'Old values of entry before request completed'
  EQUALITY octetStringMatch
  SUBSTR octetStringSubstringsMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( LOG_SCHEMA_AT .18 NAME 'reqNewRDN'
  DESC 'New RDN of request'
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE )

( LOG_SCHEMA_AT .19 NAME 'reqDeleteOldRDN'
  DESC 'Delete old RDN'
  EQUALITY booleanMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
  SINGLE-VALUE )

( LOG_SCHEMA_AT .20 NAME 'reqNewSuperior'
  DESC 'New superior DN of request'
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE )

( LOG_SCHEMA_AT .21 NAME 'reqScope'
  DESC 'Scope of request'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( LOG_SCHEMA_AT .22 NAME 'reqDerefAliases'
  DESC 'Disposition of Aliases in request'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )

( LOG_SCHEMA_AT .23 NAME 'reqAttrsOnly'
  DESC 'Attributes only'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  SINGLE-VALUE )
DESC 'Attributes and values of request'
EQUALITY booleanMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.7
SINGLE-VALUE )

( LOG_SCHEMA_AT .24 NAME 'reqFilter'
DESC 'Filter of request'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
SINGLE-VALUE )

( LOG_SCHEMA_AT .25 NAME 'reqAttr'
DESC 'Attributes of request'
EQUALITY caseIgnoreMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

( LOG_SCHEMA_AT .26 NAME 'reqSizeLimit'
DESC 'Size limit of request'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )

( LOG_SCHEMA_AT .27 NAME 'reqTimeLimit'
DESC 'Time limit of request'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )

( LOG_SCHEMA_AT .28 NAME 'reqEntries'
DESC 'Number of entries returned'
EQUALITY integerMatch
ORDERING integerOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE )

( LOG_SCHEMA_AT .29 NAME 'reqData'
DESC 'Data of extended request'
EQUALITY octetStringMatch
SUBSTR octetStringSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
SINGLE-VALUE )
5. Object Classes

5.1. Basic Audit Object Classes

This is the basic class containing attributes common to all of the LDAP requests. The following object classes all inherit from this class.

(LOG_SCHEMA_OC .1 NAME 'auditObject' DESC 'OpenLDAP request auditing' SUP top STRUCTURAL MUST ( reqStart $ reqType $ reqSession ) MAY ( reqDN $ reqAuthzID $ reqControls $ reqRespControls $ reqEnd $ reqResult $ reqMessage $ reqReferral )

These object classes are used to aggregate read operations and write operations under common parent classes.

(LOG_SCHEMA_OC .2 NAME 'auditReadObject' DESC 'OpenLDAP read request record' SUP auditObject STRUCTURAL MUST reqDN)

(LOG_SCHEMA_OC .3 NAME 'auditWriteObject' DESC 'OpenLDAP write request record' SUP auditObject STRUCTURAL MUST reqDN)

5.2. Request-Specific Object Classes

Each LDAP Request has its own object class containing all of the attributes needed to represent an instance of the request.

(LOG_SCHEMA_OC .4 NAME 'auditAbandon' DESC 'Abandon operation' SUP auditObject STRUCTURAL MUST reqId)

(LOG_SCHEMA_OC .5 NAME 'auditAdd' DESC 'Add operation' SUP auditWriteObject STRUCTURAL MUST reqMod)

(LOG_SCHEMA_OC .6 NAME 'auditBind' DESC 'Bind operation' SUP auditObject STRUCTURAL MUST ( reqDN $ reqMethod $ reqVersion )

(LOG_SCHEMA_OC .7 NAME 'auditCompare' DESC 'Compare operation' SUP auditReadObject STRUCTURAL MUST reqAssertion)

(LOG_SCHEMA_OC .8 NAME 'auditDelete' DESC 'Delete operation' SUP auditWriteObject STRUCTURAL MAY reqOld)

(LOG_SCHEMA_OC .9 NAME 'auditModify' DESC 'Modify operation' SUP auditWriteObject STRUCTURAL MUST reqMod MAY reqOld)

(LOG_SCHEMA_OC .10 NAME 'auditModRDN' DESC 'ModRDN operation' SUP auditWriteObject STRUCTURAL MUST ( reqNewRDN $ reqDeleteOldRDN ) MAY ( reqNewSuperior $ reqOld ))
( LOG_SCHEMA_OC .11 NAME 'auditSearch' DESC 'Search operation' SUP auditReadObject STRUCTURAL MUST ( reqScope $ reqDerefAliases $ reqAttrsonly ) MAY ( reqFilter $ reqAttr $ reqEntries $ reqSizeLimit $ reqTimeLimit ) )

( LOG_SCHEMA_OC .12 NAME 'auditExtended' DESC 'Extended operation' SUP auditObject STRUCTURAL MAY reqData )

5.3. Generic Container Class

This object class may be used for the parent entry of the log records.

( LOG_SCHEMA_OC .0 NAME 'auditContainer' DESC 'AuditLog container' SUP top STRUCTURAL MAY ( cn $ reqStart $ reqEnd ) )
6. Discussion of Schema

6.1. AuditObject

1. reqDN: the distinguished name of the entry the request applies to. In the case of a ModRDN request, the reqDN gives the DN of the entry before it was modified. In the case of a Search request, the reqDN is the base DN of the search.
   Syntax: DN

2. reqStart: the time the request began on the server. reqEnd: the time the request completed on the server. The timestamps MUST have high enough resolution to ensure that the reqStart values are unique. The values for reqEnd MUST also be unique, although overlap of reqStart and reqEnd values is allowed. Servers SHOULD use one of reqStart or reqEnd as the log records’ RDN. Either choice will allow records to be read in ascending order, although the two alternatives may produce different orders. In cases where the server clocks do not provide sufficient resolution, a simple counter may be used in the fractional seconds part to distinguish multiple events occurring within the same second.
   Syntax: GeneralizedTime

3. reqType: the type of request. One of: "abandon", "add", "bind", "compare", "delete", "modify", "modrdn", "search", or "extended{OID}". For Extended requests, the numeric objectIdentifier of the request is included in the string.
   Syntax: DirectoryString

4. reqSession: an implementation-defined value that is constant for all operations occurring within a Bind/Unbind sequence.
   Syntax: DirectoryString

5. reqAuthzID: the Authorization Identity used to perform the request. This will usually be the same as the reqDN of the Bind request with matching reqSession, but may be altered by various Controls and other processing.
   Syntax: DN

6. reqResult: the LDAP result code for a completed Request. This value is omitted for Requests which have no defined result (e.g. Abandon and Unbind) and also for Requests which were Abandoned or otherwise did not run to completion.
   Syntax: Integer

7. reqMessage: the textual error message accompanying the result, if any.
Syntax: DirectoryString

8. reqReferral: any referrals that accompanied the result. They are in the standard LDAP URI format [RFC2255].
   Syntax: DirectoryString

9. reqControls: the set of Request Controls accompanying a request. reqRespControls: the set of Response Controls accompanying a request result. Each value represents a single Control. Note that since Controls are transmitted as an ordered Sequence, the X-ORDERED 'VALUES' [XORDERED] schema extension is used here to preserve their ordering.
   Syntax: Control

6.2. AuditContainer

   reqStart: the timestamp of the first (oldest) record in the log.
   reqEnd: the timestamp of the last (newest) record in the log.
   Syntax: GeneralizedTime

6.3. Request-Specific Discussion

6.3.1. Abandon

   reqId: the ID of a request to Abandon.
   Syntax: Integer

6.3.2. Bind

   reqVersion: the protocol version of the request.
   Syntax: Integer

   reqMethod: the Bind method. Either "Simple" or "SASL/<mechanism>" where "<mechanism>" is the specific SASL [RFC2222] mechanism requested.
   Syntax: DirectoryString

6.3.3. Compare

   reqAssertion: the Attribute Value Assertion (AVA) of the request. The AVA is encoded according to the rules in [RFC2254].
   Syntax: DirectoryString

6.3.4. Rename

   reqNewRDN: the new RDN of the request.
   Syntax: DN
reqDeletedOldRDN: the deleteOldRDN value of the request.
Syntax: Boolean

reqNewSuperior: the new Superior DN of the request.
Syntax: DN

6.3.5. Add and Modify

reqMod: The modifications of the request. The encoding is defined by
the following grammar, using the ABNF notation defined in [RFC0822].

\[
\text{mod} = \text{attr } ^{::} \text{ modop} \\
\text{attr} = \text{AttributeDescription from [RFC2251]} \\
\text{modop} = \text{add} / \text{delete} / \text{replace} / \text{increment} \\
\text{add} = ^{+} \text{ sp value} \\
\text{delete} = ^{-} \text{ [ sp value ]} \\
\text{replace} = ^{=} \text{ [ sp value ]} \\
\text{increment} = ^{\#} \text{ sp value} \\
\text{sp} = ^{\ } \\
\text{value} = \text{AttributeValue from [RFC2251]}
\]

Note that Add requests will only use the add modop format.
Syntax: OctetString

reqOld: the previous values of a modified attribute. The encoding is
of the form attr ":: sp value, using the same definitions as for
reqMod above.
Syntax: OctetString

6.3.6. Delete

reqOld: the previous values of a deleted entry. The encoding is as
given above.
Syntax: OctetString

6.3.7. Search

reqScope: the scope of the Search request. The possible values are
as specified for the scope parameter in the LDAP URL format [RFC2255]
and [SUBORD]. Currently one of "base", "one", "sub", or "subord".
Syntax: DirectoryString

reqDerefAliases: the derefAliases parameter of the Search request. One of "never", "searching", "finding", or "always".
Syntax: DirectoryString

reqAttrsOnly: the typesOnly parameter of the request.
Syntax: Boolean

reqFilter: the Search filter, encoded according to [RFC2254].
Syntax: DirectoryString

reqSizeLimit: the size limit of the request.
reqTimeLimit: the time limit of the request.
Syntax: Integer

reqAttr: the specific attributes requested, if any.
Syntax: DirectoryString

reqEntries: the total number of entries returned for this request.
Syntax: Integer

6.3.8. Extended

reqData: the data accompanying the request, if any.
Syntax: OctetString
7. Examples

In the following examples the log records reside under the "cn=log" entry and are named by their "reqStart" attribute.

7.1. Audit Trail

This is the set of log records produced for a session comprising a Simple Bind request, a Search request, and an Unbind:

```plaintext
dn: reqStart=20051017081049.000000Z,cn=log
objectClass: auditBind
reqStart: 20051017081049.000000Z
reqEnd: 20051017081049.000001Z
reqType: bind
reqSession: 0
reqAuthzID:
reqDN: cn=manager,dc=example,dc=com
reqResult: 0
reqVersion: 3
reqMethod: SIMPLE


dn: reqStart=20051017081049.000002Z,cn=log
objectClass: auditSearch
reqStart: 20051017081049.000002Z
reqEnd: 20051017081049.000003Z
reqType: search
reqSession: 0
reqAuthzID: cn=Manager,dc=example,dc=com
reqDN: dc=example,dc=com
reqResult: 0
reqScope: one
reqDerefAliases: never
reqAttrsOnly: FALSE
reqFilter: (objectClass=*)
reqSizeLimit: -1
reqTimeLimit: -1
reqEntries: 3


dn: reqStart=20051017081049.000004Z,cn=log
objectClass: auditObject
reqStart: 20051017081049.000004Z
reqEnd: 20051017081049.000005Z
reqType: unbind
reqSession: 0
reqAuthzID: cn=Manager,dc=example,dc=com
```

7.2. Add request

This is a log record from adding an entry to the directory:

```text
dn: reqStart=20051017083706.000001Z,cn=log
objectClass: auditAdd
structuralObjectClass: auditAdd
reqStart: 20051017083706.000001Z
reqEnd: 20051017083706.000002Z
reqType: add
reqSession: 4
reqAuthzID: cn=Manager,dc=example,dc=com
reqDN: ou=People,dc=example,dc=com
reqResult: 0
reqMod: objectClass:+ organizationalUnit
reqMod: ou:+ People
reqMod: description:+ A bunch of people will be here
reqMod: structuralObjectClass:+ organizationalUnit
reqMod: entryUUID:+ f16734aa-d334-1029-9290-cd8deceec6b0
reqMod: creatorsName:+ cn=Manager,dc=example,dc=com
reqMod: createTimestamp:+ 20051017083706Z
reqMod: entryCSN:+ 20051017083706Z#000000#00#000000
reqMod: modifiersName:+ cn=Manager,dc=example,dc=com
reqMod: modifyTimestamp:+ 20051017083706Z
```

Note that operational attributes written with the request are included in the log record. All of the static data associated with an entry will be exposed, allowing a replication client to get a full copy of the entry.

7.3. Modify request
This is a log record from modifying an entry in the directory:

dn: reqStart=20051017083734.000010Z,cn=log
objectClass: auditModify
reqStart: 20051017083734.000010Z
reqEnd: 20051017083734.000011Z
reqType: modify
reqSession: 1
reqAuthzID: cn=Manager,dc=example,dc=com
reqDN: ou=People,dc=example,dc=com
reqResult: 0
reqMod: description:-
reqMod: entryCSN= 20051017083734Z#000003#00#000000
reqMod: modifiersName= cn=Manager,dc=example,dc=com
reqMod: modifyTimestamp= 20051017083734Z
reqOld: description: A bunch of people will be here

In this example the entire "description" attribute is deleted from the entry. Its original value is recorded in the "reqOld" attribute. Preserving the data allows the logs to be replayed both forwards and backwards. A client can run the log forward to bring a replica up to date, or run it backwards to undo a series of unintended operations.

7.4. Rename request

This is a log record from renaming an entry in the directory:

dn: reqStart=20051017083734.000018Z,cn=log
objectClass: auditModRDN
reqStart: 20051017083734.000018Z
reqEnd: 20051017083734.000019Z
reqType: modrdn
reqSession: 1
reqAuthzID: cn=Manager,dc=example,dc=com
reqDN: ou=People,dc=example,dc=com
reqResult: 0
reqNewRDN: ou=Populi
reqDeleteOldRDN: TRUE

7.5. Delete request
This is a log record from deleting an entry in the directory:

dn: reqStart=20051017083734.000020Z, cn=log
objectClass: auditDelete
reqStart: 20051017083734.000020Z
reqEnd: 20051017083734.000021Z
reqType: delete
reqSession: 1
reqAuthzID: cn=Manager, dc=example, dc=com
reqDN: ou=Populi, dc=example, dc=com
reqResult: 0
reqOld: ou: Populi
reqOld: objectClass: organizationalUnit
reqOld: structuralObjectClass: organizationalUnit
reqOld: entryUUID: f16734aa-d334-1029-9290-cd8deceec6b0
reqOld: creatorsName: cn=Manager, dc=example, dc=com
reqOld: createTimestamp: 20051017083706Z
reqOld: entryCSN: 20051017083734Z#000007#00#000000
reqOld: modifiersName: cn=Manager, dc=example, dc=com
reqOld: modifyTimestamp: 20051017083734Z

7.6. Usage Notes

More information is accommodated in this specification than may be
needed in typical use. Servers MAY implement only subsets of the
attributes, or provide configuration mechanisms to reduce the range
of operations covered in the log. Replication clients working from a
full log can use a search filter with the terms
"(&(objectClass=AuditWriteObject)(reqResult=0))" to filter out
irrelevant records. The "reqOld" attribute will often contain
redundant information; having an option to omit it from the logs may
also be more suitable for some sites.
8. Security Considerations

Servers implementing this scheme SHOULD NOT allow the logs to be generally readable. Extensive information about the existence and content of data, as well as the usage patterns associated with the data, will be present in the log and should only be made available to trusted users.

The structure of the log does not prevent fine-grained access controls from being used, although the rules will be necessarily longer than they would be in the primary database. E.g., while a single rule to deny access to the userPassword attribute would suffice in the primary database, two rules would be needed in the log - one to deny access to the reqOld attribute with values userPassword:* and one to deny access to the reqMod attribute with values userPassword:*

Servers implementing this scheme should not permit users to write directly to the log container object or any entries contained within.

9. Normative References


Appendix A.  IANA Considerations

In accordance with [RFC3383] (what needs to be done here?). Currently we are using
OpenLDAP_Experimental = 1.3.6.1.4.1.4203.666
LOG_SCHEMA = OpenLDAP_Experimental.11.5
LOG_SCHEMA_AT = LOG_SCHEMA.1
LOG_SCHEMA_OC = LOG_SCHEMA.2
LOG_SCHEMA_SYN = LOG_SCHEMA.3
Author’s Address

Howard Chu
Symas Corp.
18740 Oxnard Street, Suite 313A
Tarzana, California  91356
USA

Phone: +1 818 757-7087
Email: hyc@symas.com
Intellectual Property Statement

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.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

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