Status of Memo

This document is intended to be, after appropriate review and revision, submitted to the RFC Editor for publication as an Experimental RFC. Distribution of this memo is unlimited. Technical discussion of this document will take place on the IETF LDAP Extensions mailing list <ldapext@ietf.org>. Please send editorial comments directly to the author <Kurt.Zeilenga@Isode.COM>.

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and 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/1id-abstracts.html.

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

Copyright (c) 2008 IETF Trust and the persons identified as the document authors.

Please see the Full Copyright section near the end of this document for more information.
Abstract

Lightweight Directory Access Protocol (LDAP) update operations, such as Add, Delete, and Modify operations, have atomic, consistency, isolation, durability (ACID) properties. Each of these update operations act upon an entry. It is often desirable to update two or more entries in a single unit of interaction, a transaction. Transactions are necessary to support a number of applications including resource provisioning. This document extends LDAP to support transactions.

1. Overview

This document extends the Lightweight Directory Access Protocol (LDAP) [RFC4510] to allow clients to relate a number of update operations [RFC4511] and have them performed as one unit of interaction, a transaction. As with distinct update operations, each transaction has atomic, consistency, isolation, and durability (ACID) properties [ACID].

This extension consists of two extended operations, one control, and one unsolicited notification message. The Start Transaction operation is used to obtain a transaction identifier. This identifier is then attached to multiple update operations to indicate that they belong to the transaction using the Transaction Specification control. The End Transaction is used to settle (commit or abort) the transaction. The Aborted Transaction Notice is provided by the server to notify the client that the server is no longer willing or able to process an outstanding transaction.

1.1. Conventions and Terminology

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 [RFC2119].

Protocol elements are described using ASN.1 [X.680] with implicit tags. The term "BER-encoded" means the element is to be encoded using the Basic Encoding Rules [X.690] under the restrictions detailed in Section 5.1 of [RFC4511].

DSA stands for "Directory System Agent" (a server). DSE stands for "DSA-specific entry".

2. Elements of an LDAP Transaction
2.1. Start Transaction Request and Response

A Start Transaction Request is an LDAPMessage of CHOICE extendedReq where the requestName is IANA-ASSIGNED-OID.1 and the requestValue is absent.

A Start Transaction Response is an LDAPMessage of CHOICE extendedRes sent in response to a Start Transaction Request. Its responseName is absent. When the resultCode is success (0), responseValue is present and contains a transaction identifier. Otherwise, the responseValue is absent.

2.2. Transaction Specification Control

A Transaction Specification control is an LDAPControl where the controlType is IANA-ASSIGNED-OID.2, the criticality is TRUE, and the controlValue is a transaction identifier. The control is appropriate for update requests including Add, Delete, Modify, and ModifyDN (Rename) requests [RFC4511], as well as the Password Modify requests [RFC3062].

As discussed in Section 4, the Transaction Specification control can be used in conjunction with request controls appropriate for the update request.

2.3. End Transactions Request and Response

An End Transaction Request is an LDAPMessage of CHOICE extendedReq where the requestName is IANA-ASSIGNED-OID.3 and the requestValue is present and contains a BER-encoded txnEndReq.

\[
\text{txnEndReq ::= SEQUENCE } \\
\quad \text{commit BOOLEAN DEFAULT TRUE,} \\
\quad \text{identifier OCTET STRING}
\]

A commit value of TRUE indicates a request to commit the transaction identified by the identifier. A commit value of FALSE indicates a request to abort the identified transaction.

An End Transaction Response is an LDAPMessage sent in response to a End Transaction Request. Its response name is absent. The responseValue when present contains a BER-encoded txnEndRes.

\[
\text{txnEndRes ::= SEQUENCE } \\
\quad \text{messageID MessageID OPTIONAL,} \\
\quad -- \text{msgid associated with non-success resultCode}
\]
updatesControls SEQUENCE OF updateControls SEQUENCE {
  messageID MessageID,
    -- msgid associated with controls
  controls  Controls
} OPTIONAL

-- where MessageID and Controls are as specified in RFC 4511

The txnEndRes.messageID provides the message id of the update request associated with a non-success response. txnEndRes.messageID is absent when resultCode of the End Transaction Response is success (0).

The txnEndRes.updatesControls provides a facility for returning response controls that normally (i.e., in absence of transactions) would be returned in an update response. The updateControls.messageID provides the message id of the update request associated with the response controls provided in updateControls.controls.

The txnEndRes.updatesControls is absent when there are no update response controls to return.

If both txnEndRes.messageID and txnEndRes.updatesControl are absent, the responseValue of the End Transaction Response is absent.

2.4. Aborted Transaction Notice

The Aborted Transaction Notice is an Unsolicited Notification message where the responseName is IANA-ASSIGNED-OID.4 and responseValue is present and contains a transaction identifier.

3. An LDAP Transaction

3.1. Extension Discovery

To allow clients to discover support for this extension, servers implementing this specification SHOULD publish IANA-ASSIGNED-OID.1 and IANA-ASSIGNED-OID.3 as values of the 'supportedExtension' attribute [RFC4512] within the Root DSE, and publish the IANA-ASSIGNED-OID.2 as a value of the 'supportedControl' attribute [RFC4512] of the Root DSE.

A server MAY choose to advertise this extension only when the client is authorized to use it.

3.2. Starting a Transaction
A client wishing to perform a sequence of directory updates as an transaction issues a Start Transaction Request. A server which is willing and able to support transactions responds to this request with a Start Transaction Response providing a transaction identifier and with a resultCode of success (0). Otherwise, the server responds with a Start Transaction Response with a result code other than success indicating the nature of the failure.

The transaction identifier provided upon successful start of a transaction is used in subsequent protocol messages to identify this transaction.

3.3. Specification of a Transaction

The client then can issue one or more update requests, each with a Transaction Specification control containing the transaction identifier indicating the updates are to processed as part of the transaction. Each of these update request MUST have a different MessageID value. If the server is unwilling or unable to attempt to process the requested update operation as part of the transaction, the server immediately returns the appropriate response to the request with a resultCode indicating the nature of the failure. Otherwise, the server immediately returns success (0) and the defers further processing of the operation is then deferred until settlement.

If the server becomes unwilling or unable to continue the specification of a transaction, the server issues an Aborted Transaction Notice with a non-success resultCode indicating the nature of the failure. All operations that were to be processed as part of the transaction are implicitly abandoned. Upon receipt of an Aborted Transaction Notice, the client is to discontinue all use of the transaction identifier as the transaction is null and void. Any future use of identifier by the client will result in a response containing a non-success resultCode.

3.4. Transaction Settlement

A client requests settlement of transaction by issuing an End Transaction request for the transaction indicating whether it desires the transaction to be committed or aborted.

Upon receipt of a request to abort the transaction, the server is to abort the identified transaction (abandoning all operations which are part of the transaction) and indicate that it has done so by returning an End Transaction Response with a resultCode of success (0).
Upon receipt of a request to commit the transaction, the server processes all update operations of the transaction as one atomic, durable, isolated, and consistent action with each requested update being processed in turn. Either all of the requested updates are to be successfully applied or none of the requested are to be applied. The server returns an End Transaction Response with a resultCode of success (0) and no responseValue to indicate all the requested updates were applied. Otherwise, the server returns an End Transaction with an non-success resultCode indicating the nature of the failure. If the failure is associated with a particular update request, the txnEndRes.messageID in the responseValue is the messageID of this update request. If the failure was not associated with any particular update request, no txnEnd.messageID is provided.

There is no requirement that a server serialize transactions, or updates requested outside of a transaction. That is, a server MAY process multiple commit requests (from one or more clients) acting upon different sets of entries concurrently. A server MUST avoid deadlock.

3.5. Miscellaneous Issues

Transactions cannot be nested.

Each LDAP transaction should be initiated, specified, and settled within a stable security context. Between the Start request and the End response, the peers SHOULD avoid negotiating new security associations and/or layers.

Upon receipt of a Bind or Unbind request, the server SHALL abort any and all outstanding transactions without notice and nullify their identifiers.

4. Interaction with Other Extensions

The LDAP Transaction extension may be used with many but not all LDAP control extensions designed to extend Update (and possibly other) operations. The remainder of this subsection discusses interaction with a number of control extensions. Interaction with other control extensions may be discussed in other documents, in particular in control extension specifications.

4.1. Assertion Control

The Assertion [RFC4528] control is appropriate for use with update
requests specified as part of a transaction. The evaluation of the assertion is performed as part of the transaction.

The Assertion control is inappropriate for use with either the Transaction Start or End extended operations.

4.2. ManageDsaIT Control

The ManageDsaIT [RFC3296] control is appropriate for use with update requests specified as part of a transaction.

The ManageDsaIT control is inappropriate for use with either the Transaction Start or End extended operations.

4.4. Proxied Authorization Control

The Proxied Authorization [RFC4370] control is appropriate for use with the Transaction Start extended operation, but not the Transaction End extended operation or any update request specified as part of a transaction.

To request that a transaction be performed under a different authorization, the client provides a Proxied Authorization control with the Transaction Start request. If the client is not authorized to assume the requested authorization identity, the server is to return the authorizationDenied (123) resultCode in its response. Otherwise, further processing of the request and transaction is performed under the requested authorization identity.

Any proxied authorization request attached to an update request specified as part of a transaction, or attached to a Transaction end request, is to be regarded as a protocol error.

4.5. Read Entry Controls

The Pre- and Post-Read Entry [RFC4527] request control are appropriate for use with update requests specified as part of a transaction.

The response control produced in response to a Pre- or Post-Read Entry request control is returned in the txnEndRes.updatesControls field of responseValue of the End Transaction Response.

The Pre- and Post-Read Entry controls are inappropriate for use in the LDAPMessage.controls field of the Transaction Start and End request and response messages.
5. Distributed Directory Considerations

The LDAP/X.500 models provide for distributed directory operations, including server-side chaining and client-side chasing of referrals.

This document does not preclude servers from chaining operations which are part of a transaction. However, if a server does attempt such chaining, it MUST ensure that transaction semantics are provided.

This mechanism defined by this document does not support client-side chasing. Transaction identifiers are specific to a particular LDAP association (as established via the LDAP Bind operation).

The LDAP/X.500 models provide for a single-master/multiple-shadow replication architecture. There is no requirement that changes made to the directory based upon processing a transaction be replicated as one atomic action. Hence, clients SHOULD NOT assume tight data consistency nor fast data convergence of shadow copies unless they have prior knowledge that these properties are provided. Note that DONTUseCopy control [DONTUSECOPY] control may be used in conjunction with the LDAP search request to ask for the return of the authoritative copy of the entry.

6. Security Considerations

Transactions mechanisms may be the target of denial-of-service attacks, especially where implementation lock shared resources for the duration of a transaction.

General security considerations [RFC4510], especially those associated with update operations [RFC4511], apply to this extension.

7. IANA Considerations

It is requested that Internet Assigned Numbers Authority (IANA) make the following assignments.

7.1. Object Identifier

Assignment of an LDAP Object Identifier [RFC4520] to identify the protocol elements specified in this document this document is requested.

Subject: Request for LDAP Object Identifier Registration
Person & email address to contact for further information:
7.2. LDAP Protocol Mechanism

Registration of the protocol mechanisms [RFC4520] specified in this
document is requested.

Subject: Request for LDAP Protocol Mechanism Registration
Object Identifier: see table
Description: see table
Person & email address to contact for further information:
    Kurt Zeilenga <Kurt.Zeilenga@Isode.COM>
Specification: RFC XXXX
Author/Change Controller: Kurt Zeilenga <Kurt.Zeilenga@Isode.COM>
Comments:

<table>
<thead>
<tr>
<th>Object Identifier</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IANA-ASSIGNED-OID.1 E</td>
<td>E</td>
<td>Start Transaction Extended Request</td>
</tr>
<tr>
<td>IANA-ASSIGNED-OID.2 C</td>
<td>C</td>
<td>Transaction Specification Control</td>
</tr>
<tr>
<td>IANA-ASSIGNED-OID.3 E</td>
<td>E</td>
<td>End Transaction Extended Request</td>
</tr>
<tr>
<td>IANA-ASSIGNED-OID.4 N</td>
<td>N</td>
<td>Aborted Transaction Notice</td>
</tr>
</tbody>
</table>

Legend
------------------------
C => supportedControl
E => supportedExtension
N => Unsolicited Notice

8. Acknowledgments

The author gratefully acknowledges the contributions made by Internet
Engineering Task Force participants.

9. Author’s Address

Kurt D. Zeilenga
Isode Limited

Email: Kurt.Zeilenga@Isode.COM
10. References

[[Note to the RFC Editor: please replace the citation tags used in referencing Internet-Drafts with tags of the form RFCnnnn where possible.]]

10.1. Normative References

[ RFC 2119 ] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14 (also RFC 2119), March 1997.


10.2. Informative References


Full Copyright

Copyright (c) 2008 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.